云容器引擎
Autopilot
API 参考

文档版本 01

发布日期 2024-10-15

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 使用前必读

1.1 概述

欢迎使用云容器引擎（Cloud Container Engine，简称CCE）。云容器引擎提供高度可扩展的、高性能的企业级Kubernetes集群，支持运行Docker容器。借助云容器引擎，您可以在云上轻松部署、管理和扩展容器化应用程序。

您可以使用本文档提供API对云容器引擎进行相关操作，如创建、删除、变更规格、添加网卡等。

在调用云容器引擎API之前，请确保已经充分了解云容器引擎相关概念，详细信息请参见**产品介绍**。

另外，云容器引擎所提供的接口分为CCE接口与Kubernetes原生接口。通过配合使用，您可以完整的使用云容器引擎的所有功能。

- CCE接口：CCE服务通过API网关开放的接口，支持操作云服务层面的基础设施（如创建集群）。同时也支持调用集群层面的资源（如**创建工作负载**）。
- Kubernetes原生接口：直接通过Kubernetes原生API Server来调用集群层面的资源（如**创建工作负载**），但不支持操作云服务层面的基础设施（如创建集群）。

  Kubernetes原生接口版本级别的相关概念请参见**https://kubernetes.io/docs/concepts/overview/kubernetes-api/**。

  说明

  - 当前版本调用Kubernetes接口不支持HTTP长链接。
  - 当前版本调用的Kubernetes接口包含Beta级别的接口，即版本名称包含了beta（例如:v1beta1）的接口。此类接口会根据Kubernetes原生接口的变化而变化，因此推荐在非重要的情况下使用，例如短期测试集群等。

1.2 调用说明

云容器引擎提供了REST（Representational State Transfer）风格API，支持您通过HTTPS请求调用，调用方法请参见**3 如何调用API**。

1.3 终端节点（Endpoint）

终端节点（Endpoint）即调用API的**请求地址**，不同服务不同区域的终端节点不同，您可以从**地区和终端节点**查询服务的终端节点。

请您根据业务需要选择对应区域的终端节点。

- 集群管理、配额管理的URL格式为：**https://Endpoint/uri**。其中**uri**为资源路径，也即API访问的路径。
- Kubernetes API、存储管理、插件管理的URL格式为：**https://{clusterid}.Endpoint/uri**。其中**{clusterid}**为集群ID，**uri**为资源路径，也即API访问的路径。

  说明

  - 插件管理接口调用的URL格式为：**https://{clusterid}.Endpoint/uri**，但**{clusterid}**参数仅用于域名，不会被接口校验和使用。插件管理实际使用的**{clusterid}**参数请参考插件管理，填写在query或body体中。
  - **{clusterid}**参数对Kubernetes API、存储管理生效，对应需要调用接口访问的集群。

表 1-1 URL 中的参数说明

参数	描述
{clusterid}	集群ID，创建集群后，调用**获取指定项目下的集群**接口获取。
Endpoint	Web服务入口点的URL，不同服务不同区域的终端节点不同。
uri	资源路径，也即API访问路径。从具体接口的URI模块获取，例如“**获取用户Token**”API的resource-path为“v3/auth/tokens”。

1.4 约束与限制

- 云容器引擎对单个用户的资源数量和容量限定了配额，默认情况下，您最多可以创建5个集群（每个Region下），每个集群中最多可以添加 50 个节点。如果您需要创建更多的集群或添加更多的节点，请**提交工单**申请。配额的详细信息请参见**关于配额**。
- 更详细的限制请参见具体API的说明。

1.5 基本概念

- 账号

  用户注册时的账号，账号对其所拥有的资源及云服务具有完全的访问权限，可以重置用户密码、分配用户权限等。由于账号是付费主体，为了确保账号安全，建议您不要直接使用账号进行日常管理工作，而是创建用户并使用用户进行日常管理工作。
- 用户

由账号在IAM中创建的用户，是云服务的使用人员，具有身份凭证（密码和访问密钥）。

在**我的凭证**下，您可以查看账号ID和IAM用户ID。通常在调用API的鉴权过程中，您需要用到账号、用户和密码等信息。

- 区域（Region）

  从地理位置和网络时延维度划分，同一个Region内共享弹性计算、块存储、对象存储、VPC网络、弹性公网IP、镜像等公共服务。Region分为通用Region和专属Region，通用Region指面向公共租户提供通用云服务的Region；专属Region指只承载同一类业务或只面向特定租户提供业务服务的专用Region。

  详情请参见**区域和可用区**。

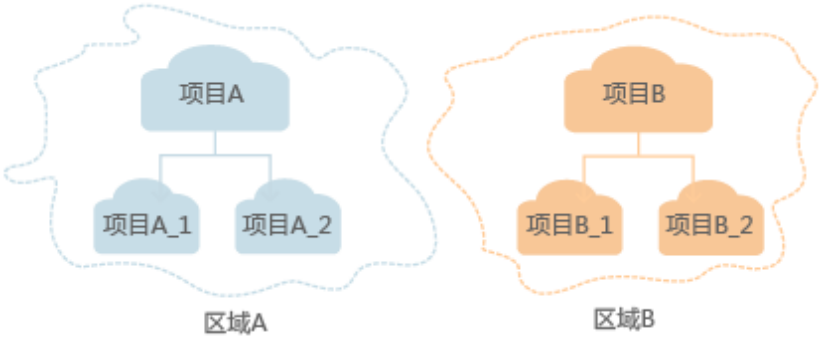
- 可用区（AZ，Availability Zone）

  一个可用区是一个或多个物理数据中心的集合，有独立的风火水电，AZ内逻辑上再将计算、网络、存储等资源划分成多个集群。一个Region中的多个AZ间通过高速光纤相连，以满足用户跨AZ构建高可用性系统的需求。

- 项目

  区域默认对应一个项目，这个项目由系统预置，用来隔离物理区域间的资源（计算资源、存储资源和网络资源），以默认项目为单位进行授权，用户可以访问您账号中该区域的所有资源。如果您希望进行更加精细的权限控制，可以在区域默认的项目中创建子项目，并在子项目中创建资源，然后以子项目为单位进行授权，使得用户仅能访问特定子项目中的资源，使得资源的权限控制更加精确。

  图 1-1 项目隔离模型

  同样在**我的凭证**下，您可以查看项目ID。

- 企业项目

  企业项目是项目的升级版，针对企业不同项目间的资源进行分组和管理，是逻辑隔离。企业项目中可以包含多个区域的资源，且项目中的资源可以迁入迁出。

  关于企业项目ID的获取及企业项目特性的详细信息，请参见《**企业管理用户指南**》。

2 API 概览

云容器引擎所提供的接口分为CCE接口与Kubernetes原生接口。通过配合使用CCE接口和Kubernetes原生接口，您可以完整的使用云容器引擎的所有功能，包括创建集群，使用Kubernetes接口创建容器工作负载，使用CCE接口监控工作负载的使用数据等。

类型	子类型	说明
CCE接口	集群管理	集群管理接口，包括创建、删除集群的接口等。 通过这些接口，您可以创建集群、获取已创建集群的信息。
	插件管理	插件管理接口，包括AddonTemplates的查询，AddonInstance的创建、更新、删除和获取。
	集群升级	集群升级接口，包括集群升级、升级前检查、集群备份等相关接口。
	配额管理	配额管理接口，支持查询CCE服务下资源配额。
	标签管理	标签管理接口，支持添加、删除集群的资源标签。
	模板管理	模板管理接口，支持模板和模板实例的创建、删除、更新、获取等操作。
Kubernetes原生接口	-	Kubernetes原生接口，关于如何调用请参见使用Kubernetes API。

Kubernetes API

API	功能	URI
Node	获取指定的Node	GET /api/v1/nodes/{name}
	列出所有的Node	GET /api/v1/nodes
	更新指定的Node	PATCH /api/v1/nodes/{name}

API	功能	URI
Namespace	创建Namespace	POST /api/v1/namespaces
	删除Namespace	DELETE /api/v1/namespaces/{name}
	获取指定的Namespace	GET /api/v1/namespaces/{name}
	替换指定的Namespace	PUT /api/v1/namespaces/{name}
	替换指定的Namespace的状态	PUT /api/v1/namespaces/{name}/status
	替换指定的Namespace的Finalize值	PUT /api/v1/namespaces/{name}/finalize
	列出Namespace	GET /api/v1/namespaces
	更新指定的Namespace	PATCH /api/v1/namespaces/{name}
Resourcequotas	获取Resourcequotas	GET /api/v1/resourcequotas
	创建Resourcequota	POST /api/v1/namespaces/{namespace}/resourcequotas
	更新Resourcequota	PUT /api/v1/namespaces/{namespace}/resourcequotas/{name}
	删除Resourcequota	DELETE /api/v1/namespaces/{namespace}/resourcequotas/{name}
Pod	创建Pod	POST /api/v1/namespaces/{namespace}/pods
	删除Pod	DELETE /api/v1/namespaces/{namespace}/pods/{name}
	删除所有的Pod	DELETE /api/v1/namespaces/{namespace}/pods
	获取指定的Pod	GET /api/v1/namespaces/{namespace}/pods/{name}
	替换指定的Pod	PUT /api/v1/namespaces/{namespace}/pods/{name}
	替换指定的Pod的状态	PUT /api/v1/namespaces/{namespace}/pods/{name}/status
	列出指定Namespaces下的所有Pod	GET /api/v1/namespaces/{namespace}/pods
	列出Pod	GET /api/v1/pods
	更新指定的Pod	PATCH /api/v1/namespaces/{namespace}/pods/{name}
Deployment	创建Deployment	POST /apis/apps/v1/namespaces/{namespace}/deployments

API	功能	URI
	创建Deployment的回滚操作	PATCH /apis/apps/v1/namespaces/{namespace}/deployments/{name}
	删除Deployment	DELETE /apis/apps/v1/namespaces/{namespace}/deployments/{name}
	删除所有的Deployment	DELETE /apis/apps/v1/namespaces/{namespace}/deployments
	获取指定的Deployment	GET /apis/apps/v1/namespaces/{namespace}/deployments/{name}
	获取指定的Deployment的状态	GET /apis/apps/v1/namespaces/{namespace}/deployments/{name}/status
	获取指定的Deployment的伸缩操作	GET /apis/apps/v1/namespaces/{namespace}/deployments/{name}/scale
	替换指定的Deployment	PUT /apis/apps/v1/namespaces/{namespace}/deployments/{name}
	替换指定的Deployment的状态	PUT /apis/apps/v1/namespaces/{namespace}/deployments/{name}/status
	替换指定的Deployment的伸缩操作	PUT /apis/apps/v1/namespaces/{namespace}/deployments/{name}/scale
	列出指定Namespace下的Deployment	GET /apis/apps/v1/namespaces/{namespace}/deployments
	列出所有的Deployment	GET /apis/apps/v1/deployments
	更新指定的Deployment	PATCH /apis/apps/v1/namespaces/{namespace}/deployments/{name}
	更新指定的Deployment的状态	PATCH /apis/apps/v1/namespaces/{namespace}/deployments/{name}/status
	更新指定的Deployment的伸缩操作	PATCH /apis/apps/v1/namespaces/{namespace}/deployments/{name}/scale
Statefulset	创建StatefulSet	POST /apis/apps/v1/namespaces/{namespace}/statefulsets
	删除指定的StatefulSet	DELETE /apis/apps/v1/namespaces/{namespace}/statefulsets/{name}
	删除所有的StatefulSet	DELETE /apis/apps/v1/namespaces/{namespace}/statefulsets
	获取指定的StatefulSet	GET /apis/apps/v1/namespaces/{namespace}/statefulsets/{name}
	获取指定的StatefulSet的状态	GET /apis/apps/v1/namespaces/{namespace}/statefulsets/{name}/status

API	功能	URI
	替换指定的StatefulSet	PUT /apis/apps/v1/namespaces/{namespace}/statefulsets/{name}
	替换指定的StatefulSet的状态	PUT /apis/apps/v1/namespaces/{namespace}/statefulsets/{name}/status
	列出指定Namespace下的StatefulSet	GET /apis/apps/v1/namespaces/{namespace}/statefulsets
	列出所有的StatefulSet	GET /apis/apps/v1/statefulsets
	更新指定的StatefulSet	PATCH /apis/apps/v1/namespaces/{namespace}/statefulsets/{name}
	更新指定的StatefulSet的状态	PATCH /apis/apps/v1/namespaces/{namespace}/statefulsets/{name}/status
Job	创建Job	POST /apis/batch/v1/namespaces/{namespace}/jobs
	删除Job	DELETE /apis/batch/v1/namespaces/{namespace}/jobs/{name}
	删除所有的Job	DELETE /apis/batch/v1/namespaces/{namespace}/jobs
	获取指定的Job	GET /apis/batch/v1/namespaces/{namespace}/jobs/{name}
	获取指定的Job的状态	GET /apis/batch/v1/namespaces/{namespace}/jobs/{name}/status
	替换指定的Job	PUT /apis/batch/v1/namespaces/{namespace}/jobs/{name}
	替换指定的Job的状态	PUT /apis/batch/v1/namespaces/{namespace}/jobs/{name}/status
	列出指定Namespace下的Job	GET /apis/batch/v1/namespaces/{namespace}/jobs
	列出所有Job	GET /apis/batch/v1/jobs
	更新指定的Job的状态	PATCH /apis/batch/v1/namespaces/{namespace}/jobs/{name}/status
	更新指定的Job	PATCH /apis/batch/v1/namespaces/{namespace}/jobs/{name}
CronJob	创建CronJob	POST /apis/batch/v1/namespaces/{namespace}/cronjobs
	删除CronJob	DELETE /apis/batch/v1/namespaces/{namespace}/cronjobs/{name}

API	功能	URI
	删除所有的CronJob	DELETE /apis/batch/v1/namespaces/{namespace}/cronjobs
	获取指定的CronJob	GET /apis/batch/v1/namespaces/{namespace}/cronjobs/{name}
	获取指定的CronJob的状态	GET /apis/batch/v1/namespaces/{namespace}/cronjobs/{name}/status
	替换指定的CronJob	PUT /apis/batch/v1/namespaces/{namespace}/cronjobs/{name}
	替换指定的CronJob的状态	PUT /apis/batch/v1/namespaces/{namespace}/cronjobs/{name}/status
	列出指定Namespace下的CronJob	GET /apis/batch/v1/namespaces/{namespace}/cronjobs
	列出所有的CronJob	GET /apis/batch/v1/cronjobs
	更新指定的CronJob的状态	PATCH /apis/batch/v1/namespaces/{namespace}/cronjobs/{name}/status
	更新指定的CronJob	PATCH /apis/batch/v1/namespaces/{namespace}/cronjobs/{name}
ReplicaSet	列出指定的ReplicaSet	GET /apis/apps/v1/namespaces/{namespace}/replicasets
	获取指定的ReplicaSet	GET /apis/apps/v1/namespaces/{namespace}/replicasets/{name}
	获取Replicasets	GET /apis/apps/v1/replicasets
ReplicationController	创建ReplicationController	POST /api/v1/namespaces/{namespace}/replicationcontrollers
	删除ReplicationController	DELETE /api/v1/namespaces/{namespace}/replicationcontrollers/{name}
	删除所有的ReplicationController	DELETE /api/v1/namespaces/{namespace}/replicationcontrollers
	获取指定Namespace下的ReplicationController	GET /api/v1/namespaces/{namespace}/replicationcontrollers/{name}
	替换指定Namespace下的ReplicationController	PUT /api/v1/namespaces/{namespace}/replicationcontrollers/{name}
	替换指定Namespace下的ReplicationController状态	PUT /api/v1/namespaces/{namespace}/replicationcontrollers/{name}/status
	列出指定Namespace下的ReplicationController	GET /api/v1/namespaces/{namespace}/replicationcontrollers

API	功能	URI
	列出ReplicationController	GET /api/v1/replicationcontrollers
	更新指定的ReplicationController	PATCH /api/v1/namespaces/{namespace}/replicationcontrollers/{name}
Endpoints	创建Endpoints	POST /api/v1/namespaces/{namespace}/endpoints
	删除Endpoints	DELETE /api/v1/namespaces/{namespace}/endpoints/{name}
	删除所有的Endpoints	DELETE /api/v1/namespaces/{namespace}/endpoints
	获取指定的Endpoints	GET /api/v1/namespaces/{namespace}/endpoints/{name}
	替换指定的Endpoints	PUT /api/v1/namespaces/{namespace}/endpoints/{name}
	列出Endpoints	GET /api/v1/endpoints
	列出指定Namespace下的Endpoints	GET /api/v1/namespaces/{namespace}/endpoints
	更新指定的Endpoints	PATCH /api/v1/namespaces/{namespace}/endpoints/{name}
Service	创建Service	POST /api/v1/namespaces/{namespace}/services
	删除指定的Service	DELETE /api/v1/namespaces/{namespace}/services/{name}
	获取指定的Service	GET /api/v1/namespaces/{namespace}/services/{name}
	替换指定的Service	PUT /api/v1/namespaces/{namespace}/services/{name}
	列出指定Namespace下的Service	GET /api/v1/namespaces/{namespace}/services
	列出Service	GET /api/v1/services
	更新指定的Service	PATCH /api/v1/namespaces/{namespace}/services/{name}
Ingress	创建Ingress	POST /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses
	更新指定的Ingress	PATCH /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}

API	功能	URI
	替换指定的Ingress	PUT /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}
	删除Ingress	DELETE /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}
	删除所有的Ingress	DELETE /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses
	获取指定的Ingress	GET /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}
	列出指定Namespace下的Ingress	GET /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses
	获取Ingress列表	GET /apis/networking.k8s.io/v1/ingresses
	获取指定Namespace下的某个Ingress对象的状态	GET /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}/status
	替换指定Namespace下的某个Ingress对象的状态	PUT /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}/status
	更新指定Namespace下的某个Ingress对象的状态	PATCH /apis/networking.k8s.io/v1/namespaces/{namespace}/ingresses/{name}/status
NetworkPolicy	创建networkpolicy	POST /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies
	更新指定的networkpolicy	PATCH /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies/{name}
	替换指定的networkpolicy	PUT /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies/{name}
	删除networkpolicy	DELETE /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies/{name}
	批量删除networkpolicy	DELETE /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies
	获取指定的networkpolicy	GET /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies/{name}

API	功能	URI
	列出指定namespace下的networkpolicy	GET /apis/networking.k8s.io/v1/namespaces/{namespace}/networkpolicies
	获取networkpolicy列表	GET /apis/networking.k8s.io/v1/networkpolicies
PersistentVolume	创建PersistentVolume	POST /api/v1/persistentvolumes
	删除指定的PersistentVolume	DELETE /api/v1/persistentvolumes/{name}
	删除所有的PersistentVolume	DELETE /api/v1/persistentvolumes
	获取指定的PersistentVolume	GET /api/v1/persistentvolumes/{name}
	替换指定的PersistentVolume	PUT /api/v1/persistentvolumes/{name}
	替换指定的PersistentVolume的状态	PUT /api/v1/persistentvolumes/{name}/status
	列出所有的PersistentVolume	GET /api/v1/persistentvolumes
	更新指定的PersistentVolume	PATCH /api/v1/persistentvolumes/{name}
PersistentVolumeClaim	创建PersistentVolumeClaim	POST /api/v1/namespaces/{namespace}/persistentvolumeclaims
	删除指定的PersistentVolumeClaim	DELETE /api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}
	删除所有的PersistentVolumeClaim	DELETE /api/v1/namespaces/{namespace}/persistentvolumeclaims
	获取指定的PersistentVolumeClaim	GET /api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}
	替换指定的PersistentVolumeClaim	PUT /api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}
	替换指定的PersistentVolumeClaim的状态	PUT /api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}/status
	列出指定的Namespace下的PersistentVolumeClaim	GET /api/v1/namespaces/{namespace}/persistentvolumeclaims
	列出所有的PersistentVolumeClaim	GET /api/v1/persistentvolumeclaims

API	功能	URI
	更新指定的PersistentVolumeClaim	PATCH /api/v1/namespaces/{namespace}/persistentvolumeclaims/{name}
ConfigMap	创建ConfigMap	POST /api/v1/namespaces/{namespace}/configmaps
	删除ConfigMap	DELETE /api/v1/namespaces/{namespace}/configmaps/{name}
	删除所有的ConfigMap	DELETE /api/v1/namespaces/{namespace}/configmaps
	获取指定的ConfigMap	GET /api/v1/namespaces/{namespace}/configmaps/{name}
	替换指定ConfigMap	PUT /api/v1/namespaces/{namespace}/configmaps/{name}
	列出指定Namespace下的ConfigMap	GET /api/v1/namespaces/{namespace}/configmaps
	列出所有的ConfigMap	GET /api/v1/configmaps
	更新指定的ConfigMap	PATCH /api/v1/namespaces/{namespace}/configmaps/{name}
Secret	创建Secret	POST /api/v1/namespaces/{namespace}/secrets
	删除Secret	DELETE /api/v1/namespaces/{namespace}/secrets/{name}
	删除指定命名空间下所有的Secret	DELETE /api/v1/namespaces/{namespace}/secrets
	获取Secret信息	GET /api/v1/namespaces/{namespace}/secrets/{name}
	替换指定的Secret	PUT /api/v1/namespaces/{namespace}/secrets/{name}
	列出指定Namespace下的Secret	GET /api/v1/namespaces/{namespace}/secrets
	列出集群下的Secret	GET /api/v1/secrets
RBAC/ClusterRole	创建ClusterRole	POST /apis/rbac.authorization.k8s.io/v1/clusterroles
	更新指定的ClusterRole	PATCH /apis/rbac.authorization.k8s.io/v1/clusterroles/{name}
	替换指定的ClusterRole	PUT /apis/rbac.authorization.k8s.io/v1/clusterroles/{name}

API	功能	URI
	删除指定的ClusterRole	DELETE /apis/rbac.authorization.k8s.io/v1/clusterroles/{name}
	批量删除ClusterRole	DELETE /apis/rbac.authorization.k8s.io/v1/clusterroles
	获取指定的ClusterRole	GET /apis/rbac.authorization.k8s.io/v1/clusterroles/{name}
	获取ClusterRole列表	GET /apis/rbac.authorization.k8s.io/v1/clusterroles
RBAC/ClusterRoleBinding	创建ClusterRoleBinding	POST /apis/rbac.authorization.k8s.io/v1/clusterrolebindings
	更新指定的ClusterRoleBinding	PATCH /apis/rbac.authorization.k8s.io/v1/clusterrolebindings/{name}
	替换指定的ClusterRoleBinding	PUT /apis/rbac.authorization.k8s.io/v1/clusterrolebindings/{name}
	删除指定的ClusterRoleBinding	DELETE /apis/rbac.authorization.k8s.io/v1/clusterrolebindings/{name}
	批量删除ClusterRoleBinding	DELETE /apis/rbac.authorization.k8s.io/v1/clusterrolebindings
	获取指定的ClusterRoleBinding	GET /apis/rbac.authorization.k8s.io/v1/clusterrolebindings/{name}
	获取ClusterRoleBinding列表	GET /apis/rbac.authorization.k8s.io/v1/clusterrolebindings
RBAC/Role	创建Role	POST /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/roles
	更新指定的Role	PATCH /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/roles/{name}
	替换指定的Role	PUT /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/roles/{name}
	删除指定的Role	DELETE /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/roles/{name}
	批量删除Role	DELETE /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/roles
	获取指定的Role	GET /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/roles/{name}
	获取指定namespace下的Role列表	GET /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/roles
	获取Role列表	GET /apis/rbac.authorization.k8s.io/v1/roles

API	功能	URI
RBAC/RoleBinding	创建RoleBinding	POST /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/rolebindings
	更新指定的RoleBinding	PATCH /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/rolebindings/{name}
	替换指定的RoleBinding	PUT /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/rolebindings/{name}
	删除指定的RoleBinding	DELETE /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/rolebindings/{name}
	批量删除RoleBinding	DELETE /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/rolebindings
	获取指定的RoleBinding	GET /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/rolebindings/{name}
	获取指定namespace下RoleBinding列表	GET /apis/rbac.authorization.k8s.io/v1/namespaces/{namespace}/rolebindings
	获取RoleBinding列表	GET /apis/rbac.authorization.k8s.io/v1/rolebindings
API groups	列出APIVersions	GET /api
	列出APIGroups	GET /apis
	listing APIResources of GroupVersion apiregistration.k8s.io/v1beta1	GET /apis/apiregistration.k8s.io/v1beta1
	listing APIResources of GroupVersion extensions/v1beta1	GET /apis/extensions/v1beta1
	listing APIResources of GroupVersion apps/v1&apps/v1beta1	GET /apis/apps/v1
	listing APIResources of GroupVersion authentication.k8s.io/v1	GET /apis/authentication.k8s.io/v1
	listing APIResources of GroupVersion authentication.k8s.io/v1beta1	GET /apis/authentication.k8s.io/v1beta1

API	功能	URI
	listing APIResources of GroupVersion authorization.k8s.io/v1	GET /apis/authorization.k8s.io/v1
	listing APIResources of GroupVersion authorization.k8s.io/v1beta1	GET /apis/authorization.k8s.io/v1beta1
	listing APIResources of GroupVersion autoscaling/v1	GET /apis/autoscaling/v1
	listing APIResources of GroupVersion batch/v1	GET /apis/batch/v1
	listing APIResources of GroupVersion certificates.k8s.io/v1beta1	GET /apis/certificates.k8s.io/v1beta1
	listing APIResources of GroupVersion networking.k8s.io/v1	GET /apis/networking.k8s.io/v1
	listing APIResources of GroupVersion policy/v1beta1	GET /apis/policy/v1beta1
	listing APIResources of GroupVersion rbac.authorization.k8s.io/v1beta1	GET /apis/rbac.authorization.k8s.io/v1beta1
	listing APIResources of GroupVersion storage.k8s.io/v1	GET /apis/storage.k8s.io/v1
	listing APIResources of GroupVersion storage.k8s.io/v1beta1	GET /apis/storage.k8s.io/v1beta1
	listing APIResources of GroupVersion apiextensions.k8s.io/v1beta1	GET /apis/apiextensions.k8s.io/v1beta1
	listing APIResources of GroupVersion v1	GET /api/v1
Event	获取Event	GET /api/v1/events
	列出指定命名空间下的Event	GET /api/v1/namespaces/{namespace}/events

3 如何调用 API

3.1 构造请求

本节介绍REST API请求的组成，并以调用IAM服务的**获取用户Token**说明如何调用API，该API获取用户的Token，Token可以用于调用其他API时鉴权。

您还可以通过这个视频教程了解如何构造请求调用API：**https://bbs.huaweicloud.com/videos/102987**。

请求 URI

请求URI由如下部分组成：

{URI-scheme}://{Endpoint}/{resource-path}?{query-string}

尽管请求URI包含在请求消息头中，但大多数语言或框架都要求您从请求消息中单独传递它，所以在此单独强调。

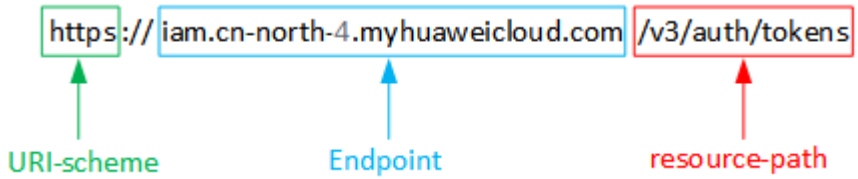
表 3-1 URI 中的参数说明

参数	描述
URI-scheme	表示用于传输请求的协议，当前所有API均采用**HTTPS**协议。
Endpoint	指定承载REST服务端点的服务器域名或IP，不同服务不同区域的Endpoint不同，您可以从**地区和终端节点**获取。 例如IAM服务在“华北-北京四”区域的Endpoint为“iam.cn-north-4.myhuaweicloud.com”。
resource-path	资源路径，也即API访问路径。从具体API的URI模块获取，例如“获取用户Token”API的resource-path为“/v3/auth/tokens”。
query-string	查询参数，是可选部分，并不是每个API都有查询参数。查询参数前面需要带一个“?”，形式为“参数名=参数取值”，例如“?limit=10”，表示查询不超过10条数据。

例如您需要获取IAM在“华北-北京四”区域的Token，则需使用“华北-北京四”区域的Endpoint（iam.cn-north-4.myhuaweicloud.com），并在**获取用户Token**的URI部分找到resource-path（/v3/auth/tokens），拼接起来如下所示。

```
https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens
```

图 3-1 URI 示意图



说明

为查看方便，在每个具体API的URI部分，只给出resource-path部分，并将请求方法写在一起。这是因为URI-scheme都是HTTPS，而Endpoint在同一个区域也相同，所以简洁起见将这两部分省略。

请求方法

HTTP请求方法（也称为操作或动词），它告诉服务你正在请求什么类型的操作。

表 3-2 HTTP 方法

方法	说明
GET	请求服务器返回指定资源。
PUT	请求服务器更新指定资源。
POST	请求服务器新增资源或执行特殊操作。
DELETE	请求服务器删除指定资源，如删除对象等。
HEAD	请求服务器资源头部。
PATCH	请求服务器更新资源的部分内容。 当资源不存在的时候，PATCH可能会去创建一个新的资源。

在**获取用户Token**的URI部分，您可以看到其请求方法为“POST”，则其请求为：

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens
```

请求消息头

附加请求头字段，如指定的URI和HTTP方法所要求的字段。例如定义消息体类型的请求头“Content-Type”，请求鉴权信息等。

详细的公共请求消息头字段请参见**表3-3**。

表 3-3 公共请求消息头

名称	描述	是否必选	示例
Host	请求的服务器信息，从服务API的URL中获取。值为hostname[:port]。端口缺省时使用默认的端口，https的默认端口为443。	否 使用AK/SK认证时该字段必选。	code.test.com or code.test.com:443
Content-Type	消息体的类型（格式）。推荐用户使用默认值application/json，有其他取值时会在具体接口中专门说明。	是	application/json
Content-Length	请求body长度，单位为Byte。	否	3495
X-Project-Id	project id，项目编号。请参考**获取项目ID**章节获取项目编号。	否 如果是专属云场景采用AK/SK认证方式的接口请求或者多project场景采用AK/SK认证的接口请求，则该字段必选。	e9993fc787d94b6c886cbaa340f9c0f4
X-Auth-Token	用户Token。 用户Token也就是调用**获取用户Token**接口的响应值，该接口是唯一不需要认证的接口。 请求响应成功后在响应消息头（Headers）中包含的“X-Subject-Token”的值即为Token值。	否 使用Token认证时该字段必选。	注：以下仅为Token示例片段 MIIPAgYJKoZIhvcNAQcCo...ggg1BBIINPXsidG9rZ

说明

API同时支持使用AK/SK认证，AK/SK认证是使用SDK对请求进行签名，签名过程会自动往请求中添加Authorization（签名认证信息）和X-Sdk-Date（请求发送的时间）请求头。

AK/SK认证的详细说明请参见**认证鉴权**的“AK/SK认证”。

对于**获取用户Token**接口，由于不需要认证，所以只添加“Content-Type”即可，添加消息头后的请求如下所示。

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

请求消息体（可选）

该部分可选。请求消息体通常以结构化格式（如JSON或XML）发出，与请求消息头中Content-Type对应，传递除请求消息头之外的内容。若请求消息体中的参数支持中文，则中文字符必须为UTF-8编码。

每个接口的请求消息体内容不同，也并不是每个接口都需要有请求消息体（或者说消息体为空），GET、DELETE操作类型的接口就不需要消息体，消息体具体内容需要根据具体接口而定。

对于**获取用户Token**接口，您可以从接口的请求部分看到所需的请求参数及参数说明。将消息体加入后的请求如下所示，加粗的斜体字段需要根据实际值填写，其中***username***为用户名，***domainname***为用户所属的账号名称，*********为用户登录密码，***xxxxxxxxxxxxxxxxxx***为project的名称，您可以从**地区和终端节点**获取。

说明

scope参数定义了Token的作用域，下面示例中获取的Token仅能访问project下的资源。您还可以设置Token的作用域为某个账号下所有资源或账号的某个project下的资源，详细定义请参见**获取用户Token**。

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json

{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "name": "xxxxxxxxxxxxxxxxxx"
            }
        }
    }
}
```

到这里为止这个请求需要的内容就具备齐全了，您可以使用**curl**、**Postman**或直接编写代码等方式发送请求调用API。对于获取用户Token接口，返回的响应消息头中“x-subject-token”就是需要获取的用户Token。有了Token之后，您就可以使用Token认证调用其他API。

3.2 认证鉴权

调用接口有如下两种认证方式，您可以选择其中一种进行认证鉴权。

- Token认证：通过Token认证调用请求。
- AK/SK认证：通过AK（Access Key ID）/SK（Secret Access Key）加密调用请求。推荐使用AK/SK认证，其安全性比Token认证要高。

Token 认证

说明

Token的有效期为24小时，需要使用一个Token鉴权时，可以先缓存起来，避免频繁调用。

Token在计算机系统中代表令牌（临时）的意思，拥有Token就代表拥有某种权限。Token认证就是在调用API的时候将Token加到请求消息头，从而通过身份认证，获得操作API的权限。

Token可通过调用**获取用户Token**接口获取，调用本服务API需要project级别的Token，即调用**获取用户Token**接口时，请求body中auth.scope的取值需要选择project，如下所示。

```
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "name": "xxxxxxxx"
            }
        }
    }
}
```

获取Token后，再调用其他接口时，您需要在请求消息头中添加“X-Auth-Token”，其值即为Token。例如Token值为“ABCDEFJ....”，则调用接口时将“X-Auth-Token: ABCDEFJ....”加到请求消息头即可，如下所示。

```
POST https://iam.cn-north-1.myhuaweicloud.com/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

您还可以通过这个视频教程了解如何使用Token认证：**https://bbs.huaweicloud.com/videos/101333**。

AK/SK 认证

说明

AK/SK签名认证方式仅支持消息体大小在12MB以内，12MB以上的请求请使用Token认证。

AK/SK认证就是使用AK/SK对请求进行签名，在请求时将签名信息添加到消息头，从而通过身份认证。

- AK（Access Key ID）：访问密钥ID。与私有访问密钥关联的唯一标识符；访问密钥ID和私有访问密钥一起使用，对请求进行加密签名。
- SK（Secret Access Key）：私有访问密钥。与访问密钥ID结合使用，对请求进行加密签名，可标识发送方，并防止请求被修改。

使用AK/SK认证时，您可以基于签名算法使用AK/SK对请求进行签名，也可以使用专门的签名SDK对请求进行签名。详细的签名方法和SDK使用方法请参见**API签名指南**。

说明

签名SDK只提供签名功能，与服务提供的SDK不同，使用时请注意。

3.3 返回结果

状态码

请求发送以后，您会收到响应，包含状态码、响应消息头和消息体。

状态码是一组从1xx到5xx的数字代码，状态码表示了请求响应的状态，完整的状态码列表请参见**状态码**。

对于**获取用户Token**接口，如果调用后返回状态码为“201”，则表示请求成功。

响应消息头

对应请求消息头，响应同样也有消息头，如“Content-type”。

对于**获取用户Token**接口，返回如**图3-2**所示的消息头，其中“x-subject-token”就是需要获取的用户Token。有了Token之后，您就可以使用Token认证调用其他API。

图 3-2 获取用户 Token 响应消息头

响应消息体

响应消息体通常以结构化格式返回，与响应消息头中Content-type对应，传递除响应消息头之外的内容。

对于**获取用户Token**接口，返回如下消息体。为篇幅起见，这里只展示部分内容。

```
{
    "token": {
        "expires_at": "2019-02-13T06:52:13.855000Z",
        "methods": [
            "password"
        ],
        "catalog": [
            {
                "endpoints": [
                    {
                        "region_id": "cn-north-4",
.....
```

当接口调用出错时，会返回错误码及错误信息说明，错误响应的Body体格式如下所示。

```
{
    "error_msg": "The format of message is error",
    "error_code": "AS.0001"
}
```

其中，error_code表示错误码 ，error_msg表示错误描述信息。

4 API

4.1 集群管理（Autopilot）

4.1.1 创建集群

功能介绍

该API用于创建一个空集群（即只有控制节点Master，没有工作节点Node）。

说明

- 集群管理的URL格式为：https://Endpoint/uri。其中uri为资源路径，也即API访问的路径。

接口约束

调用CCE接口创建集群之前，请检查是否已满足如下条件：

- 创建集群之前，您必须先确保已存在**虚拟私有云**，否则无法创建集群。若您已有虚拟私有云，可重复使用，无需重复创建。虚拟私有云为CCE集群提供一个隔离的、用户自主配置和管理的虚拟网络环境。若您没有虚拟私有云，请先进行创建，详情请参见**创建VPC**
- 创建集群之前，请提前规划好服务网段。容器隧道网络模式的集群在创建之后，无法修改网段参数；vpc网络模式/云原生网络模式的集群在创建后可以新增网段参数/子网参数，不可修改已有网段参数/子网参数，需要重新创建集群才能调整，请谨慎选择。
- 请确保已正确创建委托，并确保委托未被删除，委托校验失败将导致集群创建失败。建议登录CCE控制台，如没有创建委托，会提示您创建，如已经创建则无提示。
- 默认情况下，一个账户只能创建5个集群（每个Region下），如果您需要创建更多的集群，请申请增加配额。详情请参见**如何申请扩大配额**

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/projects/{project_id}/clusters

表 4-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-2 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-3 请求 Body 参数

参数	是否必选	参数类型	描述
kind	是	String	API类型，固定值“Cluster”或“cluster”，该值不可修改。
apiVersion	是	String	API版本，固定值“v3”，该值不可修改。
metadata	是	**AutopilotClusterMetadata** object	集群的基本信息，为集合类的元素类型，包含一组由不同名称定义的属性。
spec	是	**AutopilotClusterSpec** object	spec是集合类的元素类型，您对需要管理的集群对象进行详细描述的主体部分都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	否	**AutopilotClusterStatus** object	集合类的元素类型，用于记录对象在系统中的当前状态信息，包含了集群状态和本次创建集群作业的jobID

表 4-4 AutopilotClusterMetadata

参数	是否必选	参数类型	描述
name	是	String	集群名称。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。
uid	否	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效。在创建包周期集群时，响应体不返回集群ID。
alias	否	String	集群显示名，用于在 CCE 界面显示，该名称创建后可修改。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。 显示名和其他集群的名称、显示名不可以重复。 在创建集群、更新集群请求体中，集群显示名alias未指定或取值为空，表示与集群名称name一致。在创建集群等响应体中，集群显示名alias未配置时将不返回。
annotations	否	Map<String,String>	集群注解，由key/value组成： `"annotations": { "key1" : "value1", "key2" : "value2" }` **说明** • Annotations不用于标识和选择对象。Annotations中的元数据可以是small或large，structured或unstructured，并且可以包括标签不允许使用的字符。 • 该字段不会被数据库保存，当前仅用于指定集群待安装插件。
labels	否	Map<String,String>	集群标签，key/value对格式。 **说明** 该字段值由系统自动生成，用于升级时前端识别集群支持的特性开关，用户指定无效。
creationTimestamp	否	String	集群创建时间

参数	是否必选	参数类型	描述
updateTimestamp	否	String	集群更新时间

表 4-5 AutopilotClusterSpec

参数	是否必选	参数类型	描述
category	否	String	集群类别。Autopilot集群仅支持Turbo类型。
type	否	String	集群Master节点架构： ● VirtualMachine：Master节点为x86架构服务器
flavor	是	String	集群规格，cce.autopilot.cluster
version	否	String	集群版本，与Kubernetes社区基线版本保持一致，建议选择最新版本。 在CCE控制台支持创建三种最新版本的集群。可登录CCE控制台创建集群，在“版本”处获取到集群版本。 其它集群版本，当前仍可通过api创建，但后续会逐渐下线，具体下线策略请关注CCE官方公告。 **说明** ● 若不配置，默认创建最新版本的集群。

参数	是否必选	参数类型	描述
platformVersion	否	String	CCE集群平台版本号，表示集群版本(version)下的内部版本。用于跟踪某一集群版本内的迭代，集群版本内唯一，跨集群版本重新计数。不支持用户指定，集群创建时自动选择对应集群版本的最新平台版本。 platformVersion格式为：cce.X.Y • X: 表示内部特性版本。集群版本中特性或者补丁修复，或者OS支持等变更场景。其值从1开始单调递增。 • Y: 表示内部特性版本的补丁版本。仅用于特性版本上线后的软件包更新，不涉及其他修改。其值从0开始单调递增。
description	否	String	集群描述，对于集群使用目的的描述，可根据实际情况自定义，默认为空。集群创建成功后可通过接口**更新指定的集群**来做出修改，也可在CCE控制台中对应集群的“集群详情”下的“描述”处进行修改。仅支持utf-8编码。
customSan	否	Array of strings	集群的API Server服务端证书中的自定义SAN（Subject Alternative Name）字段，遵从SSL标准X509定义的格式规范。Autopilot集群暂不支持。 1. 不允许出现同名重复。 2. 格式符合IP和域名格式。 示例: SAN 1: DNS Name=example.com SAN 2: DNS Name=www.example.com SAN 3: DNS Name=example.net SAN 4: IP Address=93.184.216.34
enableSnat	否	Boolean	集群是否配置SNAT，仅Autopilot集群创建接口使用和返回。开启后您的集群可以通过NAT网关访问公网，默认使用所选的VPC中已有的NAT网关，否则系统将会为您自动创建一个默认规格的NAT网关并绑定弹性公网IP，自动配置SNAT规则。

参数	是否必选	参数类型	描述
enableSWRImageAccess	否	Boolean	集群是否配置镜像访问，仅Autopilot集群创建接口使用和返回。为确保您的集群节点可以从容器镜像服务中拉取镜像，默认使用所选VPC中已有的SWR和OBS终端节点，否则将会为您自动新建SWR和OBS终端节点。
enableAutopilot	否	Boolean	是否为Autopilot集群。
ipv6enable	否	Boolean	集群是否使用IPv6模式。Autopilot集群暂不支持。
hostNetwork	是	**AutopilotHostNetwork** object	节点网络参数，包含了虚拟私有云VPC和子网的ID信息，而VPC是集群内节点之间的通信依赖，所以是必选的参数集。
containerNetwork	是	**AutopilotContainerNetwork** object	容器网络参数，包含了容器网络类型和容器网段的信息。
eniNetwork	否	**AutopilotEniNetwork** object	云原生网络2.0网络配置。
serviceNetwork	否	**AutopilotServiceNetwork** object	服务网段参数，包含IPv4 CIDR。
authentication	否	**AutopilotAuthentication** object	集群认证方式相关配置。
billingMode	否	Integer	集群的计费方式。 • 0: 按需计费 默认为“按需计费”。
kubernetesSvcIpRange	否	String	服务网段参数，kubernetes clusterIP取值范围。创建集群时如若未传参，默认为"10.247.0.0/16"。该参数废弃中，推荐使用新字段serviceNetwork，包含IPv4服务网段。
clusterTags	否	Array of **AutopilotResourceTag** objects	集群资源标签

参数	是否必选	参数类型	描述
kubeProxyMode	否	String	服务转发模式： • iptables：社区传统的kube-proxy模式，完全以iptables规则的方式来实现service负载均衡。该方式最主要的问题是在服务多的时候产生太多的iptables规则，非增量式更新会引入一定的时延，大规模情况下有明显的性能问题。 **说明** 默认使用iptables转发模式。
az	否	String	可用区（仅查询返回字段）。 CCE支持的可用区请参考**地区和终端节点**
extendParam	否	**AutopilotClusterExtendParam** object	集群扩展字段，可配置多可用区集群、专属CCE集群，以及将集群创建在特定的企业项目下等。
configurationsOverride	否	Array of **AutopilotPackageConfiguration** objects	覆盖集群默认组件配置。Autopilot集群暂不支持。

表 4-6 AutopilotHostNetwork

参数	是否必选	参数类型	描述
vpc	是	String	用于创建控制节点的VPC的ID。 获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，在虚拟私有云的详情页面查找VPC ID。 • 方法2：通过虚拟私有云服务的API接口查询。 链接请参见**查询VPC列表**

参数	是否必选	参数类型	描述
subnet	是	String	用于创建控制节点的subnet的网络ID。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找网络ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-7 AutopilotContainerNetwork

参数	是否必选	参数类型	描述
mode	是	String	容器网络类型 • eni：云原生网络2.0，深度整合VPC原生ENI弹性网卡能力，采用VPC网段分配容器地址，支持ELB直通容器，享有高性能，创建集群时指定。

表 4-8 AutopilotEniNetwork

参数	是否必选	参数类型	描述
subnets	是	Array of **AutopilotNetworkSubnet** objects	ENI所在子网的IPv4子网ID列表。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-9 AutopilotNetworkSubnet

参数	是否必选	参数类型	描述
subnetID	是	String	用于创建控制节点和容器的subnet的IPv4子网ID(暂不支持IPv6)。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-10 AutopilotServiceNetwork

参数	是否必选	参数类型	描述
IPv4CIDR	否	String	kubernetes clusterIP IPv4 CIDR取值范围。创建集群时若未传参，默认为"10.247.0.0/16"。

表 4-11 AutopilotAuthentication

参数	是否必选	参数类型	描述
mode	否	String	集群认证模式。默认取值为“rbac”。

表 4-12 AutopilotResourceTag

参数	是否必选	参数类型	描述
key	否	String	Key值。 • 不能为空，最多支持128个字符 • 可用UTF-8格式表示的汉字、字母、数字和空格 • 支持部分特殊字符：_.:/=+-@ • 不能以"_sys_"开头

参数	是否必选	参数类型	描述
value	否	String	Value值。 • 可以为空但不能缺省，最多支持255个字符 • 可用UTF-8格式表示的汉字、字母、数字和空格 • 支持部分特殊字符：_.:/=+-@

表 4-13 AutopilotClusterExtendParam

参数	是否必选	参数类型	描述
enterpriseProjectId	否	String	集群所属的企业项目ID。 **说明** • 需要开通企业项目功能后才可配置企业项目。
upgradefrom	否	String	记录集群通过何种升级方式升级到当前版本。

表 4-14 AutopilotPackageConfiguration

参数	是否必选	参数类型	描述
name	否	String	组件名称
configurations	否	Array of **AutopilotConfigurationItem** objects	组件配置项

表 4-15 AutopilotConfigurationItem

参数	是否必选	参数类型	描述
name	否	String	组件配置项名称
value	否	Object	组件配置项值

表 4-16 AutopilotClusterStatus

参数	是否必选	参数类型	描述
phase	否	String	集群状态，取值如下 ● Available：可用，表示集群处于正常状态。 ● Unavailable：不可用，表示集群异常，需手动删除。 ● ScalingUp：扩容中，表示集群正处于扩容过程中。 ● ScalingDown：缩容中，表示集群正处于缩容过程中。 ● Creating：创建中，表示集群正处于创建过程中。 ● Deleting：删除中，表示集群正处于删除过程中。 ● Upgrading：升级中，表示集群正处于升级过程中。 ● Resizing：规格变更中，表示集群正处于变更规格中。 ● ResizeFailed：规格变更异常，表示集群变更规格异常。 ● RollingBack：回滚中，表示集群正处于回滚过程中。 ● RollbackFailed：回滚异常，表示集群回滚异常。 ● Hibernating：休眠中，表示集群正处于休眠过程中。 ● Hibernation：已休眠，表示集群正处于休眠状态。 ● Freezing：冻结中，表示集群正处于冻结过程中。 ● Frozen：已冻结，表示集群正处于冻结状态。 ● UnFreezing：解冻中，表示集群正处于解冻过程中。 ● Awaking：唤醒中，表示集群正处于从休眠状态唤醒的过程中。 ● Empty：集群无任何资源（已废弃） ● Error：错误，表示集群资源异常，可尝试手动删除。

参数	是否必选	参数类型	描述
jobID	否	String	任务ID,集群当前状态关联的任务ID。当前支持: • 创建集群时返回关联的任务ID，可通过任务ID查询创建集群的附属任务信息； • 删除集群或者删除集群失败时返回关联的任务ID，此字段非空时，可通过任务ID查询删除集群的附属任务信息。 **说明** 任务信息具有一定时效性，仅用于短期跟踪任务进度，请勿用于集群状态判断等额外场景。
reason	否	String	集群变为当前状态的原因，在集群在非“Available”状态下时，会返回此参数。
message	否	String	集群变为当前状态的原因的详细信息，在集群在非“Available”状态下时，会返回此参数。
endpoints	否	Array of **AutopilotClusterEndpoints** objects	集群中 kube-apiserver 的访问地址。
isLocked	否	Boolean	CBC资源锁定
lockScene	否	String	CBC资源锁定场景
lockSource	否	String	锁定资源
lockSourceId	否	String	锁定的资源ID
deleteOption	否	Object	删除配置状态（仅删除请求响应包含）
deleteStatus	否	Object	删除状态信息（仅删除请求响应包含）

表 4-17 AutopilotClusterEndpoints

参数	是否必选	参数类型	描述
url	否	String	集群中 kube-apiserver 的访问地址

参数	是否必选	参数类型	描述
type	否	String	集群访问地址的类型 • Internal：用户子网内访问的地址 • External：公网访问的地址

响应参数

状态码： 201

表 4-18 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Cluster”或“cluster”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**AutopilotClusterMetadata** object	集群的基本信息，为集合类的元素类型，包含一组由不同名称定义的属性。
spec	**AutopilotClusterSpec** object	spec是集合类的元素类型，您对需要管理的集群对象进行详细描述的主体部分都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	**AutopilotClusterStatus** object	集合类的元素类型，用于记录对象在系统中的当前状态信息，包含了集群状态和本次创建集群作业的jobID

表 4-19 AutopilotClusterMetadata

参数	参数类型	描述
name	String	集群名称。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。
uid	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效。在创建包周期集群时，响应体不返回集群ID。

参数	参数类型	描述
alias	String	集群显示名，用于在 CCE 界面显示，该名称创建后可修改。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。 显示名和其他集群的名称、显示名不可以重复。 在创建集群、更新集群请求体中，集群显示名alias未指定或取值为空，表示与集群名称name一致。在创建集群等响应体中，集群显示名alias未配置时将不返回。

annotations	Map<String,String>	集群注解，由key/value组成：
		`"annotations": {`
		`"key1" : "value1",`
		`"key2" : "value2"`
		`}`
		说明
		• Annotations不用于标识和选择对象。Annotations中的元数据可以是small或large，structured或unstructured，并且可以包括标签不允许使用的字符。
		• 该字段不会被数据库保存，当前仅用于指定集群待安装插件。

labels	Map<String,String>	集群标签，key/value对格式。 **说明** 该字段值由系统自动生成，用于升级时前端识别集群支持的特性开关，用户指定无效。
creationTimestamp	String	集群创建时间
updateTimestamp	String	集群更新时间

表 4-20 AutopilotClusterSpec

参数	参数类型	描述
category	String	集群类别。Autopilot集群仅支持Turbo类型。
type	String	集群Master节点架构： • VirtualMachine：Master节点为x86架构服务器
flavor	String	集群规格，cce.autopilot.cluster

参数	参数类型	描述
version	String	集群版本，与Kubernetes社区基线版本保持一致，建议选择最新版本。 在CCE控制台支持创建三种最新版本的集群。可登录CCE控制台创建集群，在“版本”处获取到集群版本。 其它集群版本，当前仍可通过api创建，但后续会逐渐下线，具体下线策略请关注CCE官方公告。 **说明** • 若不配置，默认创建最新版本的集群。
platformVersion	String	CCE集群平台版本号，表示集群版本(version)下的内部版本。用于跟踪某一集群版本内的迭代，集群版本内唯一，跨集群版本重新计数。不支持用户指定，集群创建时自动选择对应集群版本的最新平台版本。 platformVersion格式为：cce.X.Y • X: 表示内部特性版本。集群版本中特性或者补丁修复，或者OS支持等变更场景。其值从1开始单调递增。 • Y: 表示内部特性版本的补丁版本。仅用于特性版本上线后的软件包更新，不涉及其他修改。其值从0开始单调递增。
description	String	集群描述，对于集群使用目的的描述，可根据实际情况自定义，默认为空。集群创建成功后可通过接口**更新指定的集群**来做出修改，也可在CCE控制台中对应集群的“集群详情”下的“描述”处进行修改。仅支持utf-8编码。
customSan	Array of strings	集群的API Server服务端证书中的自定义SAN（Subject Alternative Name）字段，遵从SSL标准X509定义的格式规范。Autopilot集群暂不支持。 1. 不允许出现同名重复。 2. 格式符合IP和域名格式。 示例: SAN 1: DNS Name=example.com SAN 2: DNS Name=www.example.com SAN 3: DNS Name=example.net SAN 4: IP Address=93.184.216.34
enableSnat	Boolean	集群是否配置SNAT，仅Autopilot集群创建接口使用和返回。开启后您的集群可以通过NAT网关访问公网，默认使用所选的VPC中已有的NAT网关，否则系统将会为您自动创建一个默认规格的NAT网关并绑定弹性公网IP，自动配置SNAT规则。

参数	参数类型	描述
enableSWRImageAccess	Boolean	集群是否配置镜像访问，仅Autopilot集群创建接口使用和返回。为确保您的集群节点可以从容器镜像服务中拉取镜像，默认使用所选VPC中已有的SWR和OBS终端节点，否则将会为您自动新建SWR和OBS终端节点。
enableAutopilot	Boolean	是否为Autopilot集群。
ipv6enable	Boolean	集群是否使用IPv6模式。Autopilot集群暂不支持。
hostNetwork	**AutopilotHostNetwork** object	节点网络参数，包含了虚拟私有云VPC和子网的ID信息，而VPC是集群内节点之间的通信依赖，所以是必选的参数集。
containerNetwork	**AutopilotContainerNetwork** object	容器网络参数，包含了容器网络类型和容器网段的信息。
eniNetwork	**AutopilotEniNetwork** object	云原生网络2.0网络配置。
serviceNetwork	**AutopilotServiceNetwork** object	服务网段参数，包含IPv4 CIDR。
authentication	**AutopilotAuthentication** object	集群认证方式相关配置。
billingMode	Integer	集群的计费方式。 • 0: 按需计费 默认为“按需计费”。
kubernetesSvcIpRange	String	服务网段参数，kubernetes clusterIP取值范围。创建集群时如若未传参，默认为"10.247.0.0/16"。该参数废弃中，推荐使用新字段serviceNetwork，包含IPv4服务网段。
clusterTags	Array of **AutopilotResourceTag** objects	集群资源标签

参数	参数类型	描述
kubeProxyMode	String	服务转发模式： • iptables：社区传统的kube-proxy模式，完全以iptables规则的方式来实现service负载均衡。该方式最主要的问题是在服务多的时候产生太多的iptables规则，非增量式更新会引入一定的时延，大规模情况下有明显的性能问题。 **说明** 默认使用iptables转发模式。
az	String	可用区（仅查询返回字段）。 CCE支持的可用区请参考**地区和终端节点**
extendParam	**AutopilotClusterExtendParam** object	集群扩展字段，可配置多可用区集群、专属CCE集群，以及将集群创建在特定的企业项目下等。
configurationsOverride	Array of **AutopilotPackageConfiguration** objects	覆盖集群默认组件配置。Autopilot集群暂不支持。

表 4-21 AutopilotHostNetwork

参数	参数类型	描述
vpc	String	用于创建控制节点的VPC的ID。 获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，在虚拟私有云的详情页面查找VPC ID。 • 方法2：通过虚拟私有云服务的API接口查询。 链接请参见**查询VPC列表**
subnet	String	用于创建控制节点的subnet的网络ID。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找网络ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-22 AutopilotContainerNetwork

参数	参数类型	描述
mode	String	容器网络类型 • eni：云原生网络2.0，深度整合VPC原生ENI弹性网卡能力，采用VPC网段分配容器地址，支持ELB直通容器，享有高性能，创建集群时指定。

表 4-23 AutopilotEniNetwork

参数	参数类型	描述
subnets	Array of **AutopilotNetworkSubnet** objects	ENI所在子网的IPv4子网ID列表。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-24 AutopilotNetworkSubnet

参数	参数类型	描述
subnetID	String	用于创建控制节点和容器的subnet的IPv4子网ID(暂不支持IPv6)。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-25 AutopilotServiceNetwork

参数	参数类型	描述
IPv4CIDR	String	kubernetes clusterIP IPv4 CIDR取值范围。创建集群时若未传参，默认为"10.247.0.0/16"。

表 4-26 AutopilotAuthentication

参数	参数类型	描述
mode	String	集群认证模式。默认取值为“rbac”。

表 4-27 AutopilotResourceTag

参数	参数类型	描述
key	String	Key值。 ● 不能为空，最多支持128个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@ ● 不能以"_sys_"开头
value	String	Value值。 ● 可以为空但不能缺省，最多支持255个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@

表 4-28 AutopilotClusterExtendParam

参数	参数类型	描述
enterpriseProjectId	String	集群所属的企业项目ID。 **说明** ● 需要开通企业项目功能后才可配置企业项目。
upgradefrom	String	记录集群通过何种升级方式升级到当前版本。

表 4-29 AutopilotPackageConfiguration

参数	参数类型	描述
name	String	组件名称
configurations	Array of **AutopilotConfigurationItem** objects	组件配置项

表 4-30 AutopilotConfigurationItem

参数	参数类型	描述
name	String	组件配置项名称
value	Object	组件配置项值

表 4-31 AutopilotClusterStatus

参数	参数类型	描述
phase	String	集群状态，取值如下 ● Available：可用，表示集群处于正常状态。 ● Unavailable：不可用，表示集群异常，需手动删除。 ● ScalingUp：扩容中，表示集群正处于扩容过程中。 ● ScalingDown：缩容中，表示集群正处于缩容过程中。 ● Creating：创建中，表示集群正处于创建过程中。 ● Deleting：删除中，表示集群正处于删除过程中。 ● Upgrading：升级中，表示集群正处于升级过程中。 ● Resizing：规格变更中，表示集群正处于变更规格中。 ● ResizeFailed：规格变更异常，表示集群变更规格异常。 ● RollingBack：回滚中，表示集群正处于回滚过程中。 ● RollbackFailed：回滚异常，表示集群回滚异常。 ● Hibernating：休眠中，表示集群正处于休眠过程中。 ● Hibernation：已休眠，表示集群正处于休眠状态。 ● Freezing：冻结中，表示集群正处于冻结过程中。 ● Frozen：已冻结，表示集群正处于冻结状态。 ● UnFreezing：解冻中，表示集群正处于解冻过程中。 ● Awaking：唤醒中，表示集群正处于从休眠状态唤醒的过程中。 ● Empty：集群无任何资源（已废弃） ● Error：错误，表示集群资源异常，可尝试手动删除。

参数	参数类型	描述
jobID	String	任务ID,集群当前状态关联的任务ID。当前支持: ● 创建集群时返回关联的任务ID，可通过任务ID查询创建集群的附属任务信息； ● 删除集群或者删除集群失败时返回关联的任务ID，此字段非空时，可通过任务ID查询删除集群的附属任务信息。 **说明** 任务信息具有一定时效性，仅用于短期跟踪任务进度，请勿用于集群状态判断等额外场景。
reason	String	集群变为当前状态的原因，在集群在非“Available”状态下时，会返回此参数。
message	String	集群变为当前状态的原因的详细信息，在集群在非“Available”状态下时，会返回此参数。
endpoints	Array of **AutopilotClusterEndpoints** objects	集群中 kube-apiserver 的访问地址。
isLocked	Boolean	CBC资源锁定
lockScene	String	CBC资源锁定场景
lockSource	String	锁定资源
lockSourceId	String	锁定的资源ID
deleteOption	Object	删除配置状态（仅删除请求响应包含）
deleteStatus	Object	删除状态信息（仅删除请求响应包含）

表 4-32 AutopilotClusterEndpoints

参数	参数类型	描述
url	String	集群中 kube-apiserver 的访问地址
type	String	集群访问地址的类型 ● Internal：用户子网内访问的地址 ● External：公网访问的地址

请求示例

创建一个v1.28版本的Autopilot集群，计费模式为按需计费，。

```
/autopilot/v3/projects/{project_id}/clusters

{
  "kind" : "Cluster",
```

```
  "apiVersion" : "v3",
  "metadata" : {
    "name" : "test-cluster-autopilot",
    "annotations" : {
      "cluster.install.addons/install" : "[{\"addonTemplateName\":\"coredns\",\"values\":{\"flavor\":{\"category
\":[\"Autopilot\"],\"is_default\":true,\"name\":\"autopilot-flavor1\",\"replicas\":2,\"resources\":[{\"limitsCpu
\":1,\"limitsMem\":\"2Gi\",\"name\":\"coredns\",\"requestsCpu\":1,\"requestsMem\":\"2Gi\"}]}}},
{\"addonTemplateName\":\"metrics-server\",\"values\":{\"flavor\":{\"category\":[\"Autopilot\"],\"description
\":\"custom resources in autopilot cluster\",\"is_default\":true,\"name\":\"autopilot-flavor1\",\"replicas
\":2,\"resources\":[{\"limitsCpu\":1,\"limitsMem\":\"2Gi\",\"name\":\"metrics-server\",\"requestsCpu
\":1,\"requestsMem\":\"2Gi\"}]}}}]"
    }
  },
  "spec" : {
    "category" : "Turbo",
    "flavor" : "cce.autopilot.cluster",
    "type" : "VirtualMachine",
    "version" : "v1.28",
    "hostNetwork" : {
      "vpc" : "c6549063-d459-4ae1-9550-b5fec6741b0f"
    },
    "extendParam" : {
      "enterpriseProjectId" : "0"
    },
    "containerNetwork" : {
      "mode" : "eni"
    },
    "description" : "",
    "billingMode" : 0,
    "eniNetwork" : {
      "subnets" : [ {
        "subnetID" : "186f9322-50c5-4e5a-91e3-47da86959afc"
      } ]
    },
    "enableSWRImageAccess" : true,
    "enableSnat" : true,
    "serviceNetwork" : {
      "IPv4CIDR" : "10.247.0.0/16"
    }
  }
}
```


响应示例

状态码： 201

表示创建集群作业下发成功。


```
{
  "kind" : "Cluster",
  "apiVersion" : "v3",
  "metadata" : {
    "name" : "test-cluster-autopilot",
    "uid" : "e18f8b25-2270-11ef-a160-0255ac100100",
    "creationTimestamp" : "2024-06-04 12:49:28.773718231 +0000 UTC",
    "updateTimestamp" : "2024-06-04 12:49:28.773718305 +0000 UTC",
    "annotations" : {
      "jobid" : "e1c49157-2270-11ef-a160-0255ac100100",
      "resourceJobId" : "e18fa26f-2270-11ef-a160-0255ac100100"
    }
  },
  "spec" : {
    "category" : "Turbo",
    "type" : "VirtualMachine",
    "flavor" : "cce.autopilot.cluster",
    "version" : "v1.28",
    "platformVersion" : "cce.4.0",
    "hostNetwork" : {
      "vpc" : "c6549063-d459-4ae1-9550-b5fec6741b0f",
```

```
    "subnet" : "3b18c2d5-b352-4f59-b421-c2d2d48a1333"
  },
  "containerNetwork" : {
    "mode" : "eni"
  },
  "eniNetwork" : {
    "subnets" : [ {
      "subnetID" : "186f9322-50c5-4e5a-91e3-47da86959afc"
    } ]
  },
  "serviceNetwork" : {
    "IPv4CIDR" : "10.247.0.0/16"
  },
  "authentication" : {
    "mode" : "rbac"
  },
  "billingMode" : 0,
  "kubernetesSvcIpRange" : "10.247.0.0/16",
  "kubeProxyMode" : "iptables",
  "extendParam" : {
    "enterpriseProjectId" : "0"
  },
  "enableSWRImageAccess" : true,
  "enableSnat" : true
 },
 "status" : {
  "phase" : "Creating",
  "jobID" : "e1c49157-2270-11ef-a160-0255ac100100"
 }
}
```


SDK 代码示例

SDK代码示例如下。

Java

创建一个v1.28版本的Autopilot集群，计费模式为按需计费，。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;

import java.util.List;
import java.util.ArrayList;
import java.util.Map;
import java.util.HashMap;

public class CreateAutopilotClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";
```



```
        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAutopilotClusterRequest request = new CreateAutopilotClusterRequest();
        AutopilotCluster body = new AutopilotCluster();
        AutopilotClusterExtendParam extendParamSpec = new AutopilotClusterExtendParam();
        extendParamSpec.withEnterpriseProjectId("0");
        AutopilotServiceNetwork serviceNetworkSpec = new AutopilotServiceNetwork();
        serviceNetworkSpec.withIpv4CIDR("10.247.0.0/16");
        List<AutopilotNetworkSubnet> listEniNetworkSubnets = new ArrayList<>();
        listEniNetworkSubnets.add(
            new AutopilotNetworkSubnet()
                .withSubnetID("186f9322-50c5-4e5a-91e3-47da86959afc")
        );
        AutopilotEniNetwork eniNetworkSpec = new AutopilotEniNetwork();
        eniNetworkSpec.withSubnets(listEniNetworkSubnets);
        AutopilotContainerNetwork containerNetworkSpec = new AutopilotContainerNetwork();
        containerNetworkSpec.withMode(AutopilotContainerNetwork.ModeEnum.fromValue("eni"));
        AutopilotHostNetwork hostNetworkSpec = new AutopilotHostNetwork();
        hostNetworkSpec.withVpc("c6549063-d459-4ae1-9550-b5fec6741b0f");
        AutopilotClusterSpec specbody = new AutopilotClusterSpec();
        specbody.withCategory(AutopilotClusterSpec.CategoryEnum.fromValue("Turbo"))
            .withType(AutopilotClusterSpec.TypeEnum.fromValue("VirtualMachine"))
            .withFlavor("cce.autopilot.cluster")
            .withVersion("v1.28")
            .withDescription("")
            .withEnableSnat(true)
            .withEnableSWRImageAccess(true)
            .withHostNetwork(hostNetworkSpec)
            .withContainerNetwork(containerNetworkSpec)
            .withEniNetwork(eniNetworkSpec)
            .withServiceNetwork(serviceNetworkSpec)
            .withBillingMode(0)
            .withExtendParam(extendParamSpec);
        Map<String, String> listMetadataAnnotations = new HashMap<>();
        listMetadataAnnotations.put("cluster.install.addons/install",
"[{"addonTemplateName":"coredns","values":{"flavor":{"category":
["Autopilot"],"is_default":true,"name":"autopilot-flavor1","replicas":2,"resources":
[{"limitsCpu":1,"limitsMem":"2Gi","name":"coredns","requestsCpu":1,"requestsMem":"2Gi"}]}}},
{"addonTemplateName":"metrics-server","values":{"flavor":{"category":["Autopilot"],"description":"custom
resources in autopilot cluster","is_default":true,"name":"autopilot-flavor1","replicas":2,"resources":
[{"limitsCpu":1,"limitsMem":"2Gi","name":"metrics-server","requestsCpu":1,"requestsMem":"2Gi"}]}}}]");
        AutopilotClusterMetadata metadatabody = new AutopilotClusterMetadata();
        metadatabody.withName("test-cluster-autopilot")
            .withAnnotations(listMetadataAnnotations);
        body.withSpec(specbody);
        body.withMetadata(metadatabody);
        body.withApiVersion("v3");
        body.withKind("Cluster");
        request.withBody(body);
        try {
            CreateAutopilotClusterResponse response = client.createAutopilotCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
```

```
        }
    }
}
```

Python

创建一个v1.28版本的Autopilot集群，计费模式为按需计费，。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAutopilotClusterRequest()
        extendParamSpec = AutopilotClusterExtendParam(
            enterprise_project_id="0"
        )
        serviceNetworkSpec = AutopilotServiceNetwork(
            i_pv4_cidr="10.247.0.0/16"
        )
        listSubnetsEniNetwork = [
            AutopilotNetworkSubnet(
                subnet_id="186f9322-50c5-4e5a-91e3-47da86959afc"
            )
        ]
        eniNetworkSpec = AutopilotEniNetwork(
            subnets=listSubnetsEniNetwork
        )
        containerNetworkSpec = AutopilotContainerNetwork(
            mode="eni"
        )
        hostNetworkSpec = AutopilotHostNetwork(
            vpc="c6549063-d459-4ae1-9550-b5fec6741b0f"
        )
        specbody = AutopilotClusterSpec(
            category="Turbo",
            type="VirtualMachine",
            flavor="cce.autopilot.cluster",
            version="v1.28",
            description="",
            enable_snat=True,
            enable_swr_image_access=True,
            host_network=hostNetworkSpec,
            container_network=containerNetworkSpec,
            eni_network=eniNetworkSpec,
            service_network=serviceNetworkSpec,
            billing_mode=0,
            extend_param=extendParamSpec
```

```
        )
        listAnnotationsMetadata = {
            "cluster.install.addons/install": "[{"addonTemplateName":"coredns","values":{"flavor":{"category":
["Autopilot"],"is_default":true,"name":"autopilot-flavor1","replicas":2,"resources":
[{"limitsCpu":1,"limitsMem":"2Gi","name":"coredns","requestsCpu":1,"requestsMem":"2Gi"}]}}},
{"addonTemplateName":"metrics-server","values":{"flavor":{"category":["Autopilot"],"description":"custom
resources in autopilot cluster","is_default":true,"name":"autopilot-flavor1","replicas":2,"resources":
[{"limitsCpu":1,"limitsMem":"2Gi","name":"metrics-server","requestsCpu":1,"requestsMem":"2Gi"}]}}}]"
        }
        metadatabody = AutopilotClusterMetadata(
            name="test-cluster-autopilot",
            annotations=listAnnotationsMetadata
        )
        request.body = AutopilotCluster(
            spec=specbody,
            metadata=metadatabody,
            api_version="v3",
            kind="Cluster"
        )
        response = client.create_autopilot_cluster(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建一个v1.28版本的Autopilot集群，计费模式为按需计费，。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAutopilotClusterRequest{}
    enterpriseProjectIdExtendParam:= "0"
    extendParamSpec := &model.AutopilotClusterExtendParam{
        EnterpriseProjectId: &enterpriseProjectIdExtendParam,
    }
    iPv4CIDRServiceNetwork:= "10.247.0.0/16"
```

```
    serviceNetworkSpec := &model.AutopilotServiceNetwork{
        IPv4CIDR: &iPv4CIDRServiceNetwork,
    }
    var listSubnetsEniNetwork = []model.AutopilotNetworkSubnet{
        {
            SubnetID: "186f9322-50c5-4e5a-91e3-47da86959afc",
        },
    }
    eniNetworkSpec := &model.AutopilotEniNetwork{
        Subnets: listSubnetsEniNetwork,
    }
    containerNetworkSpec := &model.AutopilotContainerNetwork{
        Mode: model.GetAutopilotContainerNetworkModeEnum().ENI,
    }
    hostNetworkSpec := &model.AutopilotHostNetwork{
        Vpc: "c6549063-d459-4ae1-9550-b5fec6741b0f",
    }
    categorySpec:= model.GetAutopilotClusterSpecCategoryEnum().TURBO
    typeSpec:= model.GetAutopilotClusterSpecTypeEnum().VIRTUAL_MACHINE
    versionSpec:= "v1.28"
    descriptionSpec:= ""
    enableSnatSpec:= true
    enableSWRImageAccessSpec:= true
    billingModeSpec:= int32(0)
    specbody := &model.AutopilotClusterSpec{
        Category: &categorySpec,
        Type: &typeSpec,
        Flavor: "cce.autopilot.cluster",
        Version: &versionSpec,
        Description: &descriptionSpec,
        EnableSnat: &enableSnatSpec,
        EnableSWRImageAccess: &enableSWRImageAccessSpec,
        HostNetwork: hostNetworkSpec,
        ContainerNetwork: containerNetworkSpec,
        EniNetwork: eniNetworkSpec,
        ServiceNetwork: serviceNetworkSpec,
        BillingMode: &billingModeSpec,
        ExtendParam: extendParamSpec,
    }
    var listAnnotationsMetadata = map[string]string{
        "cluster.install.addons/install": "[{"addonTemplateName":"coredns","values":{"flavor":{"category":
["Autopilot"],"is_default":true,"name":"autopilot-flavor1","replicas":2,"resources":
[{"limitsCpu":1,"limitsMem":"2Gi","name":"coredns","requestsCpu":1,"requestsMem":"2Gi"}]}}},
{"addonTemplateName":"metrics-server","values":{"flavor":{"category":["Autopilot"],"description":"custom
resources in autopilot cluster","is_default":true,"name":"autopilot-flavor1","replicas":2,"resources":
[{"limitsCpu":1,"limitsMem":"2Gi","name":"metrics-server","requestsCpu":1,"requestsMem":"2Gi"}]}}}]",
    }
    metadatabody := &model.AutopilotClusterMetadata{
        Name: "test-cluster-autopilot",
        Annotations: listAnnotationsMetadata,
    }
    request.Body = &model.AutopilotCluster{
        Spec: specbody,
        Metadata: metadatabody,
        ApiVersion: "v3",
        Kind: "Cluster",
    }
    response, err := client.CreateAutopilotCluster(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	表示创建集群作业下发成功。

错误码

请参见**错误码**。

4.1.2 获取指定的集群

功能介绍

该API用于获取指定集群的详细信息。

说明

集群管理的URL格式为：https://Endpoint/uri。其中uri为资源路径，也即API访问的路径。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters/{cluster_id}

表 4-33 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

表 4-34 Query 参数

参数	是否必选	参数类型	描述
detail	否	String	查询集群详细信息。 若设置为true，获取集群下已安装插件列表中包含名称(addonTemplateName)、版本号(version)、插件的状态信息(status)，放入到annotation中。

请求参数

表 4-35 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-36 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Cluster”或“cluster”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**AutopilotClusterMetadata** object	集群的基本信息，为集合类的元素类型，包含一组由不同名称定义的属性。
spec	**AutopilotClusterSpec** object	spec是集合类的元素类型，您对需要管理的集群对象进行详细描述的主体部分都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	**AutopilotClusterStatus** object	集群状态信息

表 4-37 AutopilotClusterMetadata

参数	参数类型	描述
name	String	集群名称。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。
uid	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效。在创建包周期集群时，响应体不返回集群ID。
alias	String	集群显示名，用于在 CCE 界面显示，该名称创建后可修改。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。 显示名和其他集群的名称、显示名不可以重复。 在创建集群、更新集群请求体中，集群显示名alias未指定或取值为空，表示与集群名称name一致。在创建集群等响应体中，集群显示名alias未配置时将不返回。

annotations	Map<String,String>	集群注解，由key/value组成：
		`"annotations": {`
		`  "key1" : "value1",`
		`  "key2" : "value2"`
		`}`
		说明
		• Annotations不用于标识和选择对象。Annotations中的元数据可以是small或large，structured或unstructured，并且可以包括标签不允许使用的字符。
		• 该字段不会被数据库保存，当前仅用于指定集群待安装插件。

labels	Map<String,String>	集群标签，key/value对格式。 **说明** 该字段值由系统自动生成，用于升级时前端识别集群支持的特性开关，用户指定无效。
creationTimestamp	String	集群创建时间
updateTimestamp	String	集群更新时间

表 4-38 AutopilotClusterSpec

参数	参数类型	描述
category	String	集群类别。Autopilot集群仅支持Turbo类型。

参数	参数类型	描述
type	String	集群Master节点架构： ● VirtualMachine：Master节点为x86架构服务器
flavor	String	集群规格，cce.autopilot.cluster
version	String	集群版本，与Kubernetes社区基线版本保持一致，建议选择最新版本。 在CCE控制台支持创建三种最新版本的集群。可登录CCE控制台创建集群，在“版本”处获取到集群版本。 其它集群版本，当前仍可通过api创建，但后续会逐渐下线，具体下线策略请关注CCE官方公告。 **说明** ● 若不配置，默认创建最新版本的集群。
platformVersion	String	CCE集群平台版本号，表示集群版本(version)下的内部版本。用于跟踪某一集群版本内的迭代，集群版本内唯一，跨集群版本重新计数。不支持用户指定，集群创建时自动选择对应集群版本的最新平台版本。 platformVersion格式为：cce.X.Y ● X: 表示内部特性版本。集群版本中特性或者补丁修复，或者OS支持等变更场景。其值从1开始单调递增。 ● Y: 表示内部特性版本的补丁版本。仅用于特性版本上线后的软件包更新，不涉及其他修改。其值从0开始单调递增。
description	String	集群描述，对于集群使用目的的描述，可根据实际情况自定义，默认为空。集群创建成功后可通过接口**更新指定的集群**来做出修改，也可在CCE控制台中对应集群的“集群详情”下的“描述”处进行修改。仅支持utf-8编码。
customSan	Array of strings	集群的API Server服务端证书中的自定义SAN（Subject Alternative Name）字段，遵从SSL标准X509定义的格式规范。Autopilot集群暂不支持。 1. 不允许出现同名重复。 2. 格式符合IP和域名格式。 示例: SAN 1: DNS Name=example.com SAN 2: DNS Name=www.example.com SAN 3: DNS Name=example.net SAN 4: IP Address=93.184.216.34

参数	参数类型	描述
enableSnat	Boolean	集群是否配置SNAT，仅Autopilot集群创建接口使用和返回。开启后您的集群可以通过NAT网关访问公网，默认使用所选的VPC中已有的NAT网关，否则系统将会为您自动创建一个默认规格的NAT网关并绑定弹性公网IP，自动配置SNAT规则。
enableSWRImageAccess	Boolean	集群是否配置镜像访问，仅Autopilot集群创建接口使用和返回。为确保您的集群节点可以从容器镜像服务中拉取镜像，默认使用所选VPC中已有的SWR和OBS终端节点，否则将会为您自动新建SWR和OBS终端节点。
enableAutopilot	Boolean	是否为Autopilot集群。
ipv6enable	Boolean	集群是否使用IPv6模式。Autopilot集群暂不支持。
hostNetwork	**AutopilotHostNetwork** object	节点网络参数，包含了虚拟私有云VPC和子网的ID信息，而VPC是集群内节点之间的通信依赖，所以是必选的参数集。
containerNetwork	**AutopilotContainerNetwork** object	容器网络参数，包含了容器网络类型和容器网段的信息。
eniNetwork	**AutopilotEniNetwork** object	云原生网络2.0网络配置。
serviceNetwork	**AutopilotServiceNetwork** object	服务网段参数，包含IPv4 CIDR。
authentication	**AutopilotAuthentication** object	集群认证方式相关配置。
billingMode	Integer	集群的计费方式。 • 0: 按需计费 默认为“按需计费”。
kubernetesSvcIpRange	String	服务网段参数，kubernetes clusterIP取值范围。创建集群时如若未传参，默认为"10.247.0.0/16"。该参数废弃中，推荐使用新字段serviceNetwork，包含IPv4服务网段。
clusterTags	Array of **AutopilotResourceTag** objects	集群资源标签

参数	参数类型	描述
kubeProxyMode	String	服务转发模式： • iptables：社区传统的kube-proxy模式，完全以iptables规则的方式来实现service负载均衡。该方式最主要的问题是在服务多的时候产生太多的iptables规则，非增量式更新会引入一定的时延，大规模情况下有明显的性能问题。 **说明** 默认使用iptables转发模式。
az	String	可用区（仅查询返回字段）。 CCE支持的可用区请参考**地区和终端节点**
extendParam	**AutopilotClusterExtendParam** object	集群扩展字段，可配置多可用区集群、专属CCE集群，以及将集群创建在特定的企业项目下等。
configurationsOverride	Array of **AutopilotPackageConfiguration** objects	覆盖集群默认组件配置。Autopilot集群暂不支持。

表 4-39 AutopilotHostNetwork

参数	参数类型	描述
vpc	String	用于创建控制节点的VPC的ID。 获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，在虚拟私有云的详情页面查找VPC ID。 • 方法2：通过虚拟私有云服务的API接口查询。 链接请参见**查询VPC列表**
subnet	String	用于创建控制节点的subnet的网络ID。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找网络ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-40 AutopilotContainerNetwork

参数	参数类型	描述
mode	String	容器网络类型 • eni：云原生网络2.0，深度整合VPC原生ENI弹性网卡能力，采用VPC网段分配容器地址，支持ELB直通容器，享有高性能，创建集群时指定。

表 4-41 AutopilotEniNetwork

参数	参数类型	描述
subnets	Array of **AutopilotNetworkSubnet** objects	ENI所在子网的IPv4子网ID列表。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-42 AutopilotNetworkSubnet

参数	参数类型	描述
subnetID	String	用于创建控制节点和容器的subnet的IPv4子网ID(暂不支持IPv6)。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-43 AutopilotServiceNetwork

参数	参数类型	描述
IPv4CIDR	String	kubernetes clusterIP IPv4 CIDR取值范围。创建集群时若未传参，默认为"10.247.0.0/16"。

表 4-44 AutopilotAuthentication

参数	参数类型	描述
mode	String	集群认证模式。默认取值为“rbac”。

表 4-45 AutopilotResourceTag

参数	参数类型	描述
key	String	Key值。 ● 不能为空，最多支持128个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@ ● 不能以"_sys_"开头
value	String	Value值。 ● 可以为空但不能缺省，最多支持255个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@

表 4-46 AutopilotClusterExtendParam

参数	参数类型	描述
enterpriseProjectId	String	集群所属的企业项目ID。 **说明** ● 需要开通企业项目功能后才可配置企业项目。
upgradefrom	String	记录集群通过何种升级方式升级到当前版本。

表 4-47 AutopilotPackageConfiguration

参数	参数类型	描述
name	String	组件名称
configurations	Array of **AutopilotConfigurationItem** objects	组件配置项

表 4-48 AutopilotConfigurationItem

参数	参数类型	描述
name	String	组件配置项名称
value	Object	组件配置项值

表 4-49 AutopilotClusterStatus

参数	参数类型	描述
phase	String	集群状态，取值如下 ● Available：可用，表示集群处于正常状态。 ● Unavailable：不可用，表示集群异常，需手动删除。 ● ScalingUp：扩容中，表示集群正处于扩容过程中。 ● ScalingDown：缩容中，表示集群正处于缩容过程中。 ● Creating：创建中，表示集群正处于创建过程中。 ● Deleting：删除中，表示集群正处于删除过程中。 ● Upgrading：升级中，表示集群正处于升级过程中。 ● Resizing：规格变更中，表示集群正处于变更规格中。 ● ResizeFailed：规格变更异常，表示集群变更规格异常。 ● RollingBack：回滚中，表示集群正处于回滚过程中。 ● RollbackFailed：回滚异常，表示集群回滚异常。 ● Hibernating：休眠中，表示集群正处于休眠过程中。 ● Hibernation：已休眠，表示集群正处于休眠状态。 ● Freezing：冻结中，表示集群正处于冻结过程中。 ● Frozen：已冻结，表示集群正处于冻结状态。 ● UnFreezing：解冻中，表示集群正处于解冻过程中。 ● Awaking：唤醒中，表示集群正处于从休眠状态唤醒的过程中。 ● Empty：集群无任何资源（已废弃） ● Error：错误，表示集群资源异常，可尝试手动删除。

参数	参数类型	描述
jobID	String	任务ID,集群当前状态关联的任务ID。当前支持: • 创建集群时返回关联的任务ID，可通过任务ID查询创建集群的附属任务信息； • 删除集群或者删除集群失败时返回关联的任务ID，此字段非空时，可通过任务ID查询删除集群的附属任务信息。 **说明** 任务信息具有一定时效性，仅用于短期跟踪任务进度，请勿用于集群状态判断等额外场景。
reason	String	集群变为当前状态的原因，在集群在非“Available”状态下时，会返回此参数。
message	String	集群变为当前状态的原因的详细信息，在集群在非“Available”状态下时，会返回此参数。
endpoints	Array of **AutopilotClusterEndpoints** objects	集群中 kube-apiserver 的访问地址。
isLocked	Boolean	CBC资源锁定
lockScene	String	CBC资源锁定场景
lockSource	String	锁定资源
lockSourceId	String	锁定的资源ID
deleteOption	Object	删除配置状态（仅删除请求响应包含）
deleteStatus	Object	删除状态信息（仅删除请求响应包含）

表 4-50 AutopilotClusterEndpoints

参数	参数类型	描述
url	String	集群中 kube-apiserver 的访问地址
type	String	集群访问地址的类型 • Internal：用户子网内访问的地址 • External：公网访问的地址

请求示例

无

响应示例

状态码： 200

表示获取指定集群成功。


```
{
  "kind" : "Cluster",
  "apiVersion" : "v3",
  "metadata" : {
    "name" : "autopilot-test-v1274",
    "uid" : "087dc72a-1ff6-11ef-af74-0255ac10010b",
    "alias" : "autopilot-test-v1274",
    "annotations" : {
      "enableAutopilot" : "true"
    },
    "labels" : {
      "FeatureGates" : "arpOptimization,elbv3,xGPU"
    },
    "creationTimestamp" : "2024-06-01 09:05:03.665134 +0000 UTC",
    "updateTimestamp" : "2024-06-04 12:54:27.803532 +0000 UTC"
  },
  "spec" : {
    "category" : "Turbo",
    "type" : "VirtualMachine",
    "flavor" : "cce.autopilot.cluster",
    "version" : "v1.27",
    "platformVersion" : "cce.6.0",
    "hostNetwork" : {
      "vpc" : "13cd773f-7f9f-4821-b9b8-dba5b351e1ec",
      "subnet" : "19e3960d-b9ae-4d2d-b3a4-92ff56ae0301"
    },
    "containerNetwork" : {
      "mode" : "eni"
    },
    "eniNetwork" : {
      "subnets" : [ {
        "subnetID" : "f061c486-a190-4e2e-993c-5bc3dc9d65a8"
      } ]
    },
    "serviceNetwork" : {
      "IPv4CIDR" : "10.247.0.0/16"
    },
    "authentication" : {
      "mode" : "rbac"
    },
    "billingMode" : 0,
    "kubernetesSvcIpRange" : "10.247.0.0/16",
    "kubeProxyMode" : "iptables",
    "az" : "cn-north-7c",
    "extendParam" : {
      "enterpriseProjectId" : "0",
      "upgradefrom" : ""
    }
  },
  "status" : {
    "phase" : "Available",
    "endpoints" : [ {
      "url" : "https://087dc72a-1ff6-11ef-af74-0255ac10010b.cluster.cce.cn-north-7.myhuaweicloud.com:5443",
      "type" : "Internal"
    } ]
  }
}
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;
```

```
import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotClusterRequest request = new ShowAutopilotClusterRequest();
        request.withClusterId("{cluster_id}");
        try {
            ShowAutopilotClusterResponse response = client.showAutopilotCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"
```



```
credentials = BasicCredentials(ak, sk, projectId)

client = CceClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(CceRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = ShowAutopilotClusterRequest()
    request.cluster_id = "{cluster_id}"
    response = client.show_autopilot_cluster(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotClusterRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ShowAutopilotCluster(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取指定集群成功。

错误码

请参见**错误码**。

4.1.3 获取指定项目下的集群

功能介绍

该API用于获取指定项目下所有集群的详细信息。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters

表 4-51 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。

表 4-52 Query 参数

参数	是否必选	参数类型	描述
detail	否	String	查询集群详细信息。 若设置为true，获取集群下已安装插件列表中包含名称(addonTemplateName)、版本号(version)、插件的状态信息(status)，放入到annotation中。

参数	是否必选	参数类型	描述
status	否	String	集群状态，取值如下 ● Available：可用，表示集群处于正常状态。 ● Unavailable：不可用，表示集群异常，需手动删除。 ● ScalingUp：扩容中，表示集群正处于扩容过程中。 ● ScalingDown：缩容中，表示集群正处于缩容过程中。 ● Creating：创建中，表示集群正处于创建过程中。 ● Deleting：删除中，表示集群正处于删除过程中。 ● Upgrading：升级中，表示集群正处于升级过程中。 ● Resizing：规格变更中，表示集群正处于变更规格中。 ● ResizeFailed：规格变更异常，表示集群变更规格异常。 ● RollingBack：回滚中，表示集群正处于回滚过程中。 ● RollbackFailed：回滚异常，表示集群回滚异常。 ● Hibernating：休眠中，表示集群正处于休眠过程中。 ● Hibernation：已休眠，表示集群正处于休眠状态。 ● Freezing：冻结中，表示集群正处于冻结过程中。 ● Frozen：已冻结，表示集群正处于冻结状态。 ● UnFreezing：解冻中，表示集群正处于解冻过程中。 ● Awaking：唤醒中，表示集群正处于从休眠状态唤醒的过程中。 ● Empty：集群无任何资源（已废弃） ● Error：错误，表示集群资源异常，可尝试手动删除。

参数	是否必选	参数类型	描述
type	否	String	集群类型： • VirtualMachine：CCE集群
version	否	String	集群版本过滤

请求参数

表 4-53 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-54 响应 Body 参数

参数	参数类型	描述
kind	String	Api type
apiVersion	String	API version
items	Array of **AutopilotCluster** objects	集群对象列表，包含了当前项目下所有集群的详细信息。您可通过items.metadata.name下的值来找到对应的集群。

表 4-55 AutopilotCluster

参数	参数类型	描述
kind	String	API类型，固定值“Cluster”或“cluster”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。

参数	参数类型	描述
metadata	**AutopilotClusterMetadata** object	集群的基本信息，为集合类的元素类型，包含一组由不同名称定义的属性。
spec	**AutopilotClusterSpec** object	spec是集合类的元素类型，您对需要管理的集群对象进行详细描述的主体部分都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	**AutopilotClusterStatus** object	集合类的元素类型，用于记录对象在系统中的当前状态信息，包含了集群状态和本次创建集群作业的jobID

表 4-56 AutopilotClusterMetadata

参数	参数类型	描述
name	String	集群名称。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。
uid	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效。在创建包周期集群时，响应体不返回集群ID。
alias	String	集群显示名，用于在 CCE 界面显示，该名称创建后可修改。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。 显示名和其他集群的名称、显示名不可以重复。 在创建集群、更新集群请求体中，集群显示名alias未指定或取值为空，表示与集群名称name一致。在创建集群等响应体中，集群显示名alias未配置时将不返回。

annotations	Map<String,String>	集群注解，由key/value组成：
		`"annotations": {`
		`  "key1" : "value1",`
		`  "key2" : "value2"`
		`}`
		说明
		• Annotations不用于标识和选择对象。Annotations中的元数据可以是small或large，structured或unstructured，并且可以包括标签不允许使用的字符。
		• 该字段不会被数据库保存，当前仅用于指定集群待安装插件。

参数	参数类型	描述
labels	Map<String,String>	集群标签，key/value对格式。 **说明** 该字段值由系统自动生成，用于升级时前端识别集群支持的特性开关，用户指定无效。
creationTimestamp	String	集群创建时间
updateTimestamp	String	集群更新时间

表 4-57 AutopilotClusterSpec

参数	参数类型	描述
category	String	集群类别。Autopilot集群仅支持Turbo类型。
type	String	集群Master节点架构： ● VirtualMachine：Master节点为x86架构服务器
flavor	String	集群规格，cce.autopilot.cluster
version	String	集群版本，与Kubernetes社区基线版本保持一致，建议选择最新版本。 在CCE控制台支持创建三种最新版本的集群。可登录CCE控制台创建集群，在“版本”处获取到集群版本。 其它集群版本，当前仍可通过api创建，但后续会逐渐下线，具体下线策略请关注CCE官方公告。 **说明** ● 若不配置，默认创建最新版本的集群。
platformVersion	String	CCE集群平台版本号，表示集群版本(version)下的内部版本。用于跟踪某一集群版本内的迭代，集群版本内唯一，跨集群版本重新计数。不支持用户指定，集群创建时自动选择对应集群版本的最新平台版本。 platformVersion格式为：cce.X.Y ● X: 表示内部特性版本。集群版本中特性或者补丁修复，或者OS支持等变更场景。其值从1开始单调递增。 ● Y: 表示内部特性版本的补丁版本。仅用于特性版本上线后的软件包更新，不涉及其他修改。其值从0开始单调递增。

参数	参数类型	描述
description	String	集群描述，对于集群使用目的的描述，可根据实际情况自定义，默认为空。集群创建成功后可通过接口**更新指定的集群**来做出修改，也可在CCE控制台中对应集群的“集群详情”下的“描述”处进行修改。仅支持utf-8编码。
customSan	Array of strings	集群的API Server服务端证书中的自定义SAN（Subject Alternative Name）字段，遵从SSL标准X509定义的格式规范。Autopilot集群暂不支持。 1. 不允许出现同名重复。 2. 格式符合IP和域名格式。 示例: SAN 1: DNS Name=example.com SAN 2: DNS Name=www.example.com SAN 3: DNS Name=example.net SAN 4: IP Address=93.184.216.34
enableSnat	Boolean	集群是否配置SNAT，仅Autopilot集群创建接口使用和返回。开启后您的集群可以通过NAT网关访问公网，默认使用所选的VPC中已有的NAT网关，否则系统将会为您自动创建一个默认规格的NAT网关并绑定弹性公网IP，自动配置SNAT规则。
enableSWRImageAccess	Boolean	集群是否配置镜像访问，仅Autopilot集群创建接口使用和返回。为确保您的集群节点可以从容器镜像服务中拉取镜像，默认使用所选VPC中已有的SWR和OBS终端节点，否则将会为您自动新建SWR和OBS终端节点。
enableAutopilot	Boolean	是否为Autopilot集群。
ipv6enable	Boolean	集群是否使用IPv6模式。Autopilot集群暂不支持。
hostNetwork	**AutopilotHostNetwork** object	节点网络参数，包含了虚拟私有云VPC和子网的ID信息，而VPC是集群内节点之间的通信依赖，所以是必选的参数集。
containerNetwork	**AutopilotContainerNetwork** object	容器网络参数，包含了容器网络类型和容器网段的信息。
eniNetwork	**AutopilotEniNetwork** object	云原生网络2.0网络配置。
serviceNetwork	**AutopilotServiceNetwork** object	服务网段参数，包含IPv4 CIDR。

参数	参数类型	描述
authentication	**AutopilotAuthentication** object	集群认证方式相关配置。
billingMode	Integer	集群的计费方式。 ● 0: 按需计费 默认为“按需计费”。
kubernetesSvcIpRange	String	服务网段参数，kubernetes clusterIP取值范围。创建集群时如若未传参，默认为"10.247.0.0/16"。该参数废弃中，推荐使用新字段serviceNetwork，包含IPv4服务网段。
clusterTags	Array of **AutopilotResourceTag** objects	集群资源标签
kubeProxyMode	String	服务转发模式： ● iptables：社区传统的kube-proxy模式，完全以iptables规则的方式来实现service负载均衡。该方式最主要的问题是在服务多的时候产生太多的iptables规则，非增量式更新会引入一定的时延，大规模情况下有明显的性能问题。 **说明** 默认使用iptables转发模式。
az	String	可用区（仅查询返回字段）。 CCE支持的可用区请参考**地区和终端节点**
extendParam	**AutopilotClusterExtendParam** object	集群扩展字段，可配置多可用区集群、专属CCE集群，以及将集群创建在特定的企业项目下等。
configurationsOverride	Array of **AutopilotPackageConfiguration** objects	覆盖集群默认组件配置。Autopilot集群暂不支持。

表 4-58 AutopilotHostNetwork

参数	参数类型	描述
vpc	String	用于创建控制节点的VPC的ID。 获取方法如下： ● 方法1：登录虚拟私有云服务的控制台界面，在虚拟私有云的详情页面查找VPC ID。 ● 方法2：通过虚拟私有云服务的API接口查询。 链接请参见**查询VPC列表**
subnet	String	用于创建控制节点的subnet的网络ID。获取方法如下： ● 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找网络ID。 ● 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-59 AutopilotContainerNetwork

参数	参数类型	描述
mode	String	容器网络类型 ● eni：云原生网络2.0，深度整合VPC原生ENI弹性网卡能力，采用VPC网段分配容器地址，支持ELB直通容器，享有高性能，创建集群时指定。

表 4-60 AutopilotEniNetwork

参数	参数类型	描述
subnets	Array of **AutopilotNetworkSubnet** objects	ENI所在子网的IPv4子网ID列表。获取方法如下： ● 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 ● 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-61 AutopilotNetworkSubnet

参数	参数类型	描述
subnetID	String	用于创建控制节点和容器的subnet的IPv4子网ID(暂不支持IPv6)。获取方法如下： ● 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 ● 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-62 AutopilotServiceNetwork

参数	参数类型	描述
IPv4CIDR	String	kubernetes clusterIP IPv4 CIDR取值范围。创建集群时若未传参，默认为"10.247.0.0/16"。

表 4-63 AutopilotAuthentication

参数	参数类型	描述
mode	String	集群认证模式。默认取值为“rbac”。

表 4-64 AutopilotResourceTag

参数	参数类型	描述
key	String	Key值。 ● 不能为空，最多支持128个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@ ● 不能以"_sys_"开头
value	String	Value值。 ● 可以为空但不能缺省，最多支持255个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@

表 4-65 AutopilotClusterExtendParam

参数	参数类型	描述
enterpriseProjectId	String	集群所属的企业项目ID。 **说明** • 需要开通企业项目功能后才可配置企业项目。
upgradefrom	String	记录集群通过何种升级方式升级到当前版本。

表 4-66 AutopilotPackageConfiguration

参数	参数类型	描述
name	String	组件名称
configurations	Array of **AutopilotConfigurationItem** objects	组件配置项

表 4-67 AutopilotConfigurationItem

参数	参数类型	描述
name	String	组件配置项名称
value	Object	组件配置项值

表 4-68 AutopilotClusterStatus

参数	参数类型	描述
phase	String	集群状态，取值如下 ● Available：可用，表示集群处于正常状态。 ● Unavailable：不可用，表示集群异常，需手动删除。 ● ScalingUp：扩容中，表示集群正处于扩容过程中。 ● ScalingDown：缩容中，表示集群正处于缩容过程中。 ● Creating：创建中，表示集群正处于创建过程中。 ● Deleting：删除中，表示集群正处于删除过程中。 ● Upgrading：升级中，表示集群正处于升级过程中。 ● Resizing：规格变更中，表示集群正处于变更规格中。 ● ResizeFailed：规格变更异常，表示集群变更规格异常。 ● RollingBack：回滚中，表示集群正处于回滚过程中。 ● RollbackFailed：回滚异常，表示集群回滚异常。 ● Hibernating：休眠中，表示集群正处于休眠过程中。 ● Hibernation：已休眠，表示集群正处于休眠状态。 ● Freezing：冻结中，表示集群正处于冻结过程中。 ● Frozen：已冻结，表示集群正处于冻结状态。 ● UnFreezing：解冻中，表示集群正处于解冻过程中。 ● Awaking：唤醒中，表示集群正处于从休眠状态唤醒的过程中。 ● Empty：集群无任何资源（已废弃） ● Error：错误，表示集群资源异常，可尝试手动删除。

参数	参数类型	描述
jobID	String	任务ID,集群当前状态关联的任务ID。当前支持: • 创建集群时返回关联的任务ID，可通过任务ID查询创建集群的附属任务信息； • 删除集群或者删除集群失败时返回关联的任务ID，此字段非空时，可通过任务ID查询删除集群的附属任务信息。 **说明** 任务信息具有一定时效性，仅用于短期跟踪任务进度，请勿用于集群状态判断等额外场景。
reason	String	集群变为当前状态的原因，在集群在非“Available”状态下时，会返回此参数。
message	String	集群变为当前状态的原因的详细信息，在集群在非“Available”状态下时，会返回此参数。
endpoints	Array of **AutopilotClusterEndpoints** objects	集群中 kube-apiserver 的访问地址。
isLocked	Boolean	CBC资源锁定
lockScene	String	CBC资源锁定场景
lockSource	String	锁定资源
lockSourceId	String	锁定的资源ID
deleteOption	Object	删除配置状态（仅删除请求响应包含）
deleteStatus	Object	删除状态信息（仅删除请求响应包含）

表 4-69 AutopilotClusterEndpoints

参数	参数类型	描述
url	String	集群中 kube-apiserver 的访问地址
type	String	集群访问地址的类型 • Internal：用户子网内访问的地址 • External：公网访问的地址

请求示例

无

响应示例

状态码： 200

表示获取集群列表成功。


```
{
  "kind" : "Cluster",
  "apiVersion" : "v3",
  "items" : [ {
    "kind" : "Cluster",
    "apiVersion" : "v3",
    "metadata" : {
      "name" : "test",
      "uid" : "de63df79-1de1-11ef-95a4-0255ac1007fa",
      "creationTimestamp" : "2024-05-29 17:35:40.770923 +0000 UTC",
      "updateTimestamp" : "2024-06-04 08:14:13.881756 +0000 UTC",
      "labels" : {
        "FeatureGates" : "arpOptimization,elbv3,xGPU"
      },
      "alias" : "s00648239-b003-v127"
    },
    "spec" : {
      "category" : "Turbo",
      "type" : "VirtualMachine",
      "flavor" : "cce.autopilot.cluster",
      "version" : "v1.27",
      "hostNetwork" : {
        "vpc" : "26958bf6-9ce3-4184-9e19-d793880a162b",
        "subnet" : "df93b82f-5196-43a9-a3f1-78deeab504eb"
      },
      "containerNetwork" : {
        "mode" : "eni"
      },
      "eniNetwork" : {
        "subnets" : [ {
          "subnetID" : "97205694-3537-45b6-9459-c98e9704574a"
        } ]
      },
      "serviceNetwork" : {
        "IPv4CIDR" : "10.247.0.0/16"
      },
      "authentication" : {
        "mode" : "rbac"
      },
      "billingMode" : 0,
      "kubernetesSvcIpRange" : "10.247.0.0/16",
      "kubeProxyMode" : "iptables",
      "az" : "cn-north-7c",
      "extendParam" : {
        "enterpriseProjectId" : "0",
        "upgradefrom" : ""
      }
    },
    "status" : {
      "phase" : "Available",
      "endpoints" : [ {
        "url" : "https://de63df79-1de1-11ef-95a4-0255ac1007fa.cluster.cce.cn-
north-7.myhuaweicloud.com:5443",
        "type" : "Internal"
      } ]
    }
  } ]
}
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;
```

```
import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotClustersSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotClustersRequest request = new ListAutopilotClustersRequest();
        try {
            ListAutopilotClustersResponse response = client.listAutopilotClusters(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"
```



```
    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotClustersRequest()
        response = client.list_autopilot_clusters(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListAutopilotClustersRequest{}
    response, err := client.ListAutopilotClusters(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群列表成功。

错误码

请参见**错误码**。

4.1.4 更新指定的集群

功能介绍

该API用于更新指定的集群。

说明

集群管理的URL格式为：https://Endpoint/uri。其中uri为资源路径，也即API访问的路径。

调用方法

请参见**如何调用API**。

URI

PUT /autopilot/v3/projects/{project_id}/clusters/{cluster_id}

表 4-70 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-71 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-72 请求 Body 参数

参数	是否必选	参数类型	描述
spec	是	**AutopilotClusterInformationSpec** object	具体集群参数
metadata	否	**AutopilotClusterMetadataForUpdate** object	集群基本信息，包含与名称相关的字段

表 4-73 AutopilotClusterInformationSpec

参数	是否必选	参数类型	描述
description	否	String	集群的描述信息。 1. 字符取值范围[0,200]。不包含~$%^&*<>[]{}()'"#\等特殊字符。 2. 仅运行（Available）的集群允许修改。
customSan	否	Array of strings	集群的API Server服务端证书中的自定义SAN（Subject Alternative Name）字段，遵从SSL标准X509定义的格式规范。Autopilot集群暂不支持。 1. 不允许出现同名重复。 2. 格式符合IP和域名格式。 示例: SAN 1: DNS Name=example.com SAN 2: DNS Name=www.example.com SAN 3: DNS Name=example.net SAN 4: IP Address=93.184.216.34
eniNetwork	否	**AutopilotEniNetworkUpdate** object	云原生网络2.0网络配置，包含集群的容器子网信息。

表 4-74 AutopilotEniNetworkUpdate

参数	是否必选	参数类型	描述
subnets	否	Array of **AutopilotNetworkSubnet** objects	IPv4子网ID列表。 只允许新增子网，不可删除已有子网，请谨慎选择。 请求体中需包含所有已经存在的subnet。

表 4-75 AutopilotNetworkSubnet

参数	是否必选	参数类型	描述
subnetID	是	String	用于创建控制节点和容器的subnet的IPv4子网ID(暂不支持IPv6)。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-76 AutopilotClusterMetadataForUpdate

参数	是否必选	参数类型	描述
alias	否	String	集群显示名。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。 显示名和其他集群的名称、显示名不可以重复。 为空时表示不进行修改。

响应参数

状态码： 200

表 4-77 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Cluster”或“cluster”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**AutopilotClusterMetadata** object	集群的基本信息，为集合类的元素类型，包含一组由不同名称定义的属性。
spec	**AutopilotClusterSpec** object	spec是集合类的元素类型，您对需要管理的集群对象进行详细描述的主体部分都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	**AutopilotClusterStatus** object	集合类的元素类型，用于记录对象在系统中的当前状态信息，包含了集群状态和本次创建集群作业的jobID

表 4-78 AutopilotClusterMetadata

参数	参数类型	描述
name	String	集群名称。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。
uid	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效。在创建包周期集群时，响应体不返回集群ID。
alias	String	集群显示名，用于在 CCE 界面显示，该名称创建后可修改。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。 显示名和其他集群的名称、显示名不可以重复。 在创建集群、更新集群请求体中，集群显示名alias未指定或取值为空，表示与集群名称name一致。在创建集群等响应体中，集群显示名alias未配置时将不返回。

参数	参数类型	描述

annotations	Map<String,String>	集群注解，由key/value组成：
		"annotations": {
		"key1" : "value1",
		"key2" : "value2"
		}
		说明
		● Annotations不用于标识和选择对象。Annotations中的元数据可以是small或large，structured或unstructured，并且可以包括标签不允许使用的字符。
		● 该字段不会被数据库保存，当前仅用于指定集群待安装插件。

labels	Map<String,String>	集群标签，key/value对格式。 **说明** 该字段值由系统自动生成，用于升级时前端识别集群支持的特性开关，用户指定无效。
creationTimestamp	String	集群创建时间
updateTimestamp	String	集群更新时间

表 4-79 AutopilotClusterSpec

参数	参数类型	描述
category	String	集群类别。Autopilot集群仅支持Turbo类型。
type	String	集群Master节点架构： ● VirtualMachine：Master节点为x86架构服务器
flavor	String	集群规格，cce.autopilot.cluster
version	String	集群版本，与Kubernetes社区基线版本保持一致，建议选择最新版本。 在CCE控制台支持创建三种最新版本的集群。可登录CCE控制台创建集群，在“版本”处获取到集群版本。 其它集群版本，当前仍可通过api创建，但后续会逐渐下线，具体下线策略请关注CCE官方公告。 **说明** ● 若不配置，默认创建最新版本的集群。

参数	参数类型	描述
platformVersion	String	CCE集群平台版本号，表示集群版本(version)下的内部版本。用于跟踪某一集群版本内的迭代，集群版本内唯一，跨集群版本重新计数。不支持用户指定，集群创建时自动选择对应集群版本的最新平台版本。 platformVersion格式为：cce.X.Y • X: 表示内部特性版本。集群版本中特性或者补丁修复，或者OS支持等变更场景。其值从1开始单调递增。 • Y: 表示内部特性版本的补丁版本。仅用于特性版本上线后的软件包更新，不涉及其他修改。其值从0开始单调递增。
description	String	集群描述，对于集群使用目的的描述，可根据实际情况自定义，默认为空。集群创建成功后可通过接口**更新指定的集群**来做出修改，也可在CCE控制台中对应集群的“集群详情”下的“描述”处进行修改。仅支持utf-8编码。
customSan	Array of strings	集群的API Server服务端证书中的自定义SAN（Subject Alternative Name）字段，遵从SSL标准X509定义的格式规范。Autopilot集群暂不支持。 1. 不允许出现同名重复。 2. 格式符合IP和域名格式。 示例: SAN 1: DNS Name=example.com SAN 2: DNS Name=www.example.com SAN 3: DNS Name=example.net SAN 4: IP Address=93.184.216.34
enableSnat	Boolean	集群是否配置SNAT，仅Autopilot集群创建接口使用和返回。开启后您的集群可以通过NAT网关访问公网，默认使用所选的VPC中已有的NAT网关，否则系统将会为您自动创建一个默认规格的NAT网关并绑定弹性公网IP，自动配置SNAT规则。
enableSWRImageAccess	Boolean	集群是否配置镜像访问，仅Autopilot集群创建接口使用和返回。为确保您的集群节点可以从容器镜像服务中拉取镜像，默认使用所选VPC中已有的SWR和OBS终端节点，否则将会为您自动新建SWR和OBS终端节点。
enableAutopilot	Boolean	是否为Autopilot集群。
ipv6enable	Boolean	集群是否使用IPv6模式。Autopilot集群暂不支持。

参数	参数类型	描述
hostNetwork	**AutopilotHostNetwork** object	节点网络参数，包含了虚拟私有云VPC和子网的ID信息，而VPC是集群内节点之间的通信依赖，所以是必选的参数集。
containerNetwork	**AutopilotContainerNetwork** object	容器网络参数，包含了容器网络类型和容器网段的信息。
eniNetwork	**AutopilotEniNetwork** object	云原生网络2.0网络配置。
serviceNetwork	**AutopilotServiceNetwork** object	服务网段参数，包含IPv4 CIDR。
authentication	**AutopilotAuthentication** object	集群认证方式相关配置。
billingMode	Integer	集群的计费方式。 ● 0: 按需计费 默认为“按需计费”。
kubernetesSvcIpRange	String	服务网段参数，kubernetes clusterIP取值范围。创建集群时如若未传参，默认为"10.247.0.0/16"。该参数废弃中，推荐使用新字段serviceNetwork，包含IPv4服务网段。
clusterTags	Array of **AutopilotResourceTag** objects	集群资源标签
kubeProxyMode	String	服务转发模式： ● iptables：社区传统的kube-proxy模式，完全以iptables规则的方式来实现service负载均衡。该方式最主要的问题是在服务多的时候产生太多的iptables规则，非增量式更新会引入一定的时延，大规模情况下有明显的性能问题。 **说明** 默认使用iptables转发模式。
az	String	可用区（仅查询返回字段）。 CCE支持的可用区请参考**地区和终端节点**
extendParam	**AutopilotClusterExtendParam** object	集群扩展字段，可配置多可用区集群、专属CCE集群，以及将集群创建在特定的企业项目下等。

参数	参数类型	描述
configurationsOverride	Array of **AutopilotPackageConfiguration** objects	覆盖集群默认组件配置。Autopilot集群暂不支持。

表 4-80 AutopilotHostNetwork

参数	参数类型	描述
vpc	String	用于创建控制节点的VPC的ID。 获取方法如下： ● 方法1：登录虚拟私有云服务的控制台界面，在虚拟私有云的详情页面查找VPC ID。 ● 方法2：通过虚拟私有云服务的API接口查询。 链接请参见**查询VPC列表**
subnet	String	用于创建控制节点的subnet的网络ID。获取方法如下： ● 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找网络ID。 ● 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-81 AutopilotContainerNetwork

参数	参数类型	描述
mode	String	容器网络类型 ● eni：云原生网络2.0，深度整合VPC原生ENI弹性网卡能力，采用VPC网段分配容器地址，支持ELB直通容器，享有高性能，创建集群时指定。

表 4-82 AutopilotEniNetwork

参数	参数类型	描述
subnets	Array of **AutopilotNetworkSubnet** objects	ENI所在子网的IPv4子网ID列表。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-83 AutopilotNetworkSubnet

参数	参数类型	描述
subnetID	String	用于创建控制节点和容器的subnet的IPv4子网ID(暂不支持IPv6)。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-84 AutopilotServiceNetwork

参数	参数类型	描述
IPv4CIDR	String	kubernetes clusterIP IPv4 CIDR取值范围。创建集群时若未传参，默认为"10.247.0.0/16"。

表 4-85 AutopilotAuthentication

参数	参数类型	描述
mode	String	集群认证模式。默认取值为“rbac”。

表 4-86 AutopilotResourceTag

参数	参数类型	描述
key	String	Key值。 ● 不能为空，最多支持128个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@ ● 不能以"_sys_"开头
value	String	Value值。 ● 可以为空但不能缺省，最多支持255个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@

表 4-87 AutopilotClusterExtendParam

参数	参数类型	描述
enterpriseProjectId	String	集群所属的企业项目ID。 **说明** ● 需要开通企业项目功能后才可配置企业项目。
upgradefrom	String	记录集群通过何种升级方式升级到当前版本。

表 4-88 AutopilotPackageConfiguration

参数	参数类型	描述
name	String	组件名称
configurations	Array of **AutopilotConfigurationItem** objects	组件配置项

表 4-89 AutopilotConfigurationItem

参数	参数类型	描述
name	String	组件配置项名称
value	Object	组件配置项值

表 4-90 AutopilotClusterStatus

参数	参数类型	描述
phase	String	集群状态，取值如下 ● Available：可用，表示集群处于正常状态。 ● Unavailable：不可用，表示集群异常，需手动删除。 ● ScalingUp：扩容中，表示集群正处于扩容过程中。 ● ScalingDown：缩容中，表示集群正处于缩容过程中。 ● Creating：创建中，表示集群正处于创建过程中。 ● Deleting：删除中，表示集群正处于删除过程中。 ● Upgrading：升级中，表示集群正处于升级过程中。 ● Resizing：规格变更中，表示集群正处于变更规格中。 ● ResizeFailed：规格变更异常，表示集群变更规格异常。 ● RollingBack：回滚中，表示集群正处于回滚过程中。 ● RollbackFailed：回滚异常，表示集群回滚异常。 ● Hibernating：休眠中，表示集群正处于休眠过程中。 ● Hibernation：已休眠，表示集群正处于休眠状态。 ● Freezing：冻结中，表示集群正处于冻结过程中。 ● Frozen：已冻结，表示集群正处于冻结状态。 ● UnFreezing：解冻中，表示集群正处于解冻过程中。 ● Awaking：唤醒中，表示集群正处于从休眠状态唤醒的过程中。 ● Empty：集群无任何资源（已废弃） ● Error：错误，表示集群资源异常，可尝试手动删除。

参数	参数类型	描述
jobID	String	任务ID,集群当前状态关联的任务ID。当前支持: • 创建集群时返回关联的任务ID，可通过任务ID查询创建集群的附属任务信息； • 删除集群或者删除集群失败时返回关联的任务ID，此字段非空时，可通过任务ID查询删除集群的附属任务信息。 **说明** 任务信息具有一定时效性，仅用于短期跟踪任务进度，请勿用于集群状态判断等额外场景。
reason	String	集群变为当前状态的原因，在集群在非“Available”状态下时，会返回此参数。
message	String	集群变为当前状态的原因的详细信息，在集群在非“Available”状态下时，会返回此参数。
endpoints	Array of **AutopilotClusterEndpoints** objects	集群中 kube-apiserver 的访问地址。
isLocked	Boolean	CBC资源锁定
lockScene	String	CBC资源锁定场景
lockSource	String	锁定资源
lockSourceId	String	锁定的资源ID
deleteOption	Object	删除配置状态（仅删除请求响应包含）
deleteStatus	Object	删除状态信息（仅删除请求响应包含）

表 4-91 AutopilotClusterEndpoints

参数	参数类型	描述
url	String	集群中 kube-apiserver 的访问地址
type	String	集群访问地址的类型 • Internal：用户子网内访问的地址 • External：公网访问的地址

请求示例

仅更新集群描述

```
{
  "spec" : {
    "description" : "new description"
```

```
  }
}
```

响应示例

状态码： 200

表示更新指定集群成功。

```
{
  "kind" : "Cluster",
  "apiVersion" : "v3",
  "metadata" : {
    "name" : "s00842745-128-3-r0",
    "uid" : "c82a7d44-1cc4-11ef-8460-0255ac101780",
    "creationTimestamp" : "2024-05-28 07:34:56.917029 +0000 UTC",
    "updateTimestamp" : "2024-06-04 13:12:35.749294 +0000 UTC",
    "labels" : {
      "FeatureGates" : "arpOptimization,elbv3,xGPU"
    },
    "annotations" : {
      "enableAutopilot" : "true",
      "feature:supportNodePoolScaleGroup" : "true"
    },
    "alias" : "s00842745-128-3-r0"
  },
  "spec" : {
    "category" : "Turbo",
    "type" : "VirtualMachine",
    "flavor" : "cce.autopilot.cluster",
    "version" : "v1.28",
    "platformVersion" : "cce.3.0",
    "description" : "new description",
    "hostNetwork" : {
      "vpc" : "f9122377-7b2e-49c9-ab9e-bf0bfdd807b4",
      "subnet" : "6b757878-c428-4e76-a7e9-5e3853778d5d"
    },
    "containerNetwork" : {
      "mode" : "eni"
    },
    "eniNetwork" : {
      "eniSubnetId" : "b04a4b46-9f99-44a1-9a98-de52e549e68b",
      "subnets" : [ {
        "subnetID" : "b04a4b46-9f99-44a1-9a98-de52e549e68b"
      } ]
    },
    "serviceNetwork" : {
      "IPv4CIDR" : "10.247.0.0/16"
    },
    "authentication" : {
      "mode" : "rbac"
    },
    "billingMode" : 0,
    "kubernetesSvcIpRange" : "10.247.0.0/16",
    "kubeProxyMode" : "iptables",
    "az" : "cn-north-7c",
    "extendParam" : {
      "enterpriseProjectId" : "5ebc44c1-617b-4d93-8d49-895b8a457a1f",
      "upgradefrom" : ""
    }
  },
  "status" : {
    "phase" : "Available",
    "endpoints" : [ {
      "url" : "https://c82a7d44-1cc4-11ef-8460-0255ac101780.cluster.cce.cn-
north-7.myhuaweicloud.com:5443",
      "type" : "Internal"
    } ]
```

```
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

仅更新集群描述

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class UpdateAutopilotClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateAutopilotClusterRequest request = new UpdateAutopilotClusterRequest();
        request.withClusterId("{cluster_id}");
        AutopilotClusterInformation body = new AutopilotClusterInformation();
        AutopilotClusterInformationSpec specbody = new AutopilotClusterInformationSpec();
        specbody.withDescription("new description");
        body.withSpec(specbody);
        request.withBody(body);
        try {
            UpdateAutopilotClusterResponse response = client.updateAutopilotCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

仅更新集群描述

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateAutopilotClusterRequest()
        request.cluster_id = "{cluster_id}"
        specbody = AutopilotClusterInformationSpec(
            description="new description"
        )
        request.body = AutopilotClusterInformation(
            spec=specbody
        )
        response = client.update_autopilot_cluster(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

仅更新集群描述

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
```

```
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateAutopilotClusterRequest{}
    request.ClusterId = "{cluster_id}"
    descriptionSpec:= "new description"
    specbody := &model.AutopilotClusterInformationSpec{
        Description: &descriptionSpec,
    }
    request.Body = &model.AutopilotClusterInformation{
        Spec: specbody,
    }
    response, err := client.UpdateAutopilotCluster(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示更新指定集群成功。

错误码

请参见**错误码**。

4.1.5 删除集群

功能介绍

该API用于删除一个指定的集群。

说明

集群管理的URL格式为：https://Endpoint/uri。其中uri为资源路径，也即API访问的路径。

调用方法

请参见**如何调用API**。

URI

DELETE /autopilot/v3/projects/{project_id}/clusters/{cluster_id}

表 4-92 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

表 4-93 Query 参数

参数	是否必选	参数类型	描述
delete_efs	否	String	是否删除SFS Turbo（极速文件存储卷）， 枚举取值： • true或block (执行删除流程，失败则阻塞后续流程) • try (执行删除流程，失败则忽略，并继续执行后续流程) • false或skip (跳过删除流程，默认选项)
delete_eni	否	String	是否删除eni ports（原生弹性网卡）。 枚举取值： • true或block：执行删除流程，失败则阻塞后续流程，默认选项。 • try：执行删除流程，失败则忽略，并继续执行后续流程。 • false或skip：跳过删除流程。

参数	是否必选	参数类型	描述
delete_net	否	String	是否删除elb（弹性负载均衡）等集群Service/Ingress相关资源。 枚举取值： ● true或block：执行删除流程，失败则阻塞后续流程，默认选项。 ● try：执行删除流程，失败则忽略，并继续执行后续流程。 ● false或skip：跳过删除流程。
delete_obs	否	String	是否删除obs（对象存储卷）。 枚举取值： ● true或block：执行删除流程，失败则阻塞后续流程。 ● try：执行删除流程，失败则忽略，并继续执行后续流程。 ● false或skip：跳过删除流程，默认选项。
delete_sfs30	否	String	是否删除sfs3.0（文件存储卷3.0）。 枚举取值： ● true或block：执行删除流程，失败则阻塞后续流程。 ● try：执行删除流程，失败则忽略，并继续执行后续流程。 ● false或skip：跳过删除流程，默认选项。
lts_reclaim_policy	否	String	是否删除LTS资源（日志组/日志流）。 枚举取值： ● Delete_Log_Group：删除日志组，失败则忽略，并继续执行后续流程。 ● Delete_Master_Log_Stream：删除Master接入日志流，失败则忽略，并继续执行后续流程，默认选项。 ● Retain：跳过删除流程。

请求参数

表 4-94 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-95 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Cluster”或“cluster”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**AutopilotClusterMetadata** object	集群的基本信息，为集合类的元素类型，包含一组由不同名称定义的属性。
spec	**AutopilotClusterSpec** object	spec是集合类的元素类型，您对需要管理的集群对象进行详细描述的主体部分都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	**AutopilotClusterStatus** object	集合类的元素类型，用于记录对象在系统中的当前状态信息，包含了集群状态和本次创建集群作业的jobID

表 4-96 AutopilotClusterMetadata

参数	参数类型	描述
name	String	集群名称。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。

参数	参数类型	描述
uid	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效。在创建包周期集群时，响应体不返回集群ID。
alias	String	集群显示名，用于在 CCE 界面显示，该名称创建后可修改。 命名规则：以小写字母开头，由小写字母、数字、中划线(-)组成，长度范围4-128位，且不能以中划线(-)结尾。 显示名和其他集群的名称、显示名不可以重复。 在创建集群、更新集群请求体中，集群显示名alias未指定或取值为空，表示与集群名称name一致。在创建集群等响应体中，集群显示名alias未配置时将不返回。

annotations	Map<String,String>	集群注解，由key/value组成：
		`"annotations": {`
		`"key1" : "value1",`
		`"key2" : "value2"`
		`}`
		说明
		● Annotations不用于标识和选择对象。Annotations中的元数据可以是small或large，structured或unstructured，并且可以包括标签不允许使用的字符。
		● 该字段不会被数据库保存，当前仅用于指定集群待安装插件。

labels	Map<String,String>	集群标签，key/value对格式。 **说明** 该字段值由系统自动生成，用于升级时前端识别集群支持的特性开关，用户指定无效。
creationTimestamp	String	集群创建时间
updateTimestamp	String	集群更新时间

表 4-97 AutopilotClusterSpec

参数	参数类型	描述
category	String	集群类别。Autopilot集群仅支持Turbo类型。
type	String	集群Master节点架构： ● VirtualMachine：Master节点为x86架构服务器
flavor	String	集群规格，cce.autopilot.cluster

参数	参数类型	描述
version	String	集群版本，与Kubernetes社区基线版本保持一致，建议选择最新版本。 在CCE控制台支持创建三种最新版本的集群。可登录CCE控制台创建集群，在“版本”处获取到集群版本。 其它集群版本，当前仍可通过api创建，但后续会逐渐下线，具体下线策略请关注CCE官方公告。 **说明** • 若不配置，默认创建最新版本的集群。
platformVersion	String	CCE集群平台版本号，表示集群版本(version)下的内部版本。用于跟踪某一集群版本内的迭代，集群版本内唯一，跨集群版本重新计数。不支持用户指定，集群创建时自动选择对应集群版本的最新平台版本。 platformVersion格式为：cce.X.Y • X: 表示内部特性版本。集群版本中特性或者补丁修复，或者OS支持等变更场景。其值从1开始单调递增。 • Y: 表示内部特性版本的补丁版本。仅用于特性版本上线后的软件包更新，不涉及其他修改。其值从0开始单调递增。
description	String	集群描述，对于集群使用目的的描述，可根据实际情况自定义，默认为空。集群创建成功后可通过接口**更新指定的集群**来做出修改，也可在CCE控制台中对应集群的“集群详情”下的“描述”处进行修改。仅支持utf-8编码。
customSan	Array of strings	集群的API Server服务端证书中的自定义SAN（Subject Alternative Name）字段，遵从SSL标准X509定义的格式规范。Autopilot集群暂不支持。 1. 不允许出现同名重复。 2. 格式符合IP和域名格式。 示例: SAN 1: DNS Name=example.com SAN 2: DNS Name=www.example.com SAN 3: DNS Name=example.net SAN 4: IP Address=93.184.216.34
enableSnat	Boolean	集群是否配置SNAT，仅Autopilot集群创建接口使用和返回。开启后您的集群可以通过NAT网关访问公网，默认使用所选的VPC中已有的NAT网关，否则系统将会为您自动创建一个默认规格的NAT网关并绑定弹性公网IP，自动配置SNAT规则。

参数	参数类型	描述
enableSWRImageAccess	Boolean	集群是否配置镜像访问，仅Autopilot集群创建接口使用和返回。为确保您的集群节点可以从容器镜像服务中拉取镜像，默认使用所选VPC中已有的SWR和OBS终端节点，否则将会为您自动新建SWR和OBS终端节点。
enableAutopilot	Boolean	是否为Autopilot集群。
ipv6enable	Boolean	集群是否使用IPv6模式。Autopilot集群暂不支持。
hostNetwork	**AutopilotHostNetwork** object	节点网络参数，包含了虚拟私有云VPC和子网的ID信息，而VPC是集群内节点之间的通信依赖，所以是必选的参数集。
containerNetwork	**AutopilotContainerNetwork** object	容器网络参数，包含了容器网络类型和容器网段的信息。
eniNetwork	**AutopilotEniNetwork** object	云原生网络2.0网络配置。
serviceNetwork	**AutopilotServiceNetwork** object	服务网段参数，包含IPv4 CIDR。
authentication	**AutopilotAuthentication** object	集群认证方式相关配置。
billingMode	Integer	集群的计费方式。 • 0: 按需计费 默认为“按需计费”。
kubernetesSvcIpRange	String	服务网段参数，kubernetes clusterIP取值范围。创建集群时如若未传参，默认为"10.247.0.0/16"。该参数废弃中，推荐使用新字段serviceNetwork，包含IPv4服务网段。
clusterTags	Array of **AutopilotResourceTag** objects	集群资源标签

参数	参数类型	描述
kubeProxyMode	String	服务转发模式： • iptables：社区传统的kube-proxy模式，完全以iptables规则的方式来实现service负载均衡。该方式最主要的问题是在服务多的时候产生太多的iptables规则，非增量式更新会引入一定的时延，大规模情况下有明显的性能问题。 **说明** 默认使用iptables转发模式。
az	String	可用区（仅查询返回字段）。 CCE支持的可用区请参考**地区和终端节点**
extendParam	**AutopilotClusterExtendParam** object	集群扩展字段，可配置多可用区集群、专属CCE集群，以及将集群创建在特定的企业项目下等。
configurationsOverride	Array of **AutopilotPackageConfiguration** objects	覆盖集群默认组件配置。Autopilot集群暂不支持。

表 4-98 AutopilotHostNetwork

参数	参数类型	描述
vpc	String	用于创建控制节点的VPC的ID。 获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，在虚拟私有云的详情页面查找VPC ID。 • 方法2：通过虚拟私有云服务的API接口查询。 链接请参见**查询VPC列表**
subnet	String	用于创建控制节点的subnet的网络ID。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找网络ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-99 AutopilotContainerNetwork

参数	参数类型	描述
mode	String	容器网络类型 • eni：云原生网络2.0，深度整合VPC原生ENI弹性网卡能力，采用VPC网段分配容器地址，支持ELB直通容器，享有高性能，创建集群时指定。

表 4-100 AutopilotEniNetwork

参数	参数类型	描述
subnets	Array of **AutopilotNetworkSubnet** objects	ENI所在子网的IPv4子网ID列表。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-101 AutopilotNetworkSubnet

参数	参数类型	描述
subnetID	String	用于创建控制节点和容器的subnet的IPv4子网ID(暂不支持IPv6)。获取方法如下： • 方法1：登录虚拟私有云服务的控制台界面，单击VPC下的子网，进入子网详情页面，查找IPv4子网ID。 • 方法2：通过虚拟私有云服务的查询子网列表接口查询。 链接请参见**查询子网列表**

表 4-102 AutopilotServiceNetwork

参数	参数类型	描述
IPv4CIDR	String	kubernetes clusterIP IPv4 CIDR取值范围。创建集群时若未传参，默认为"10.247.0.0/16"。

表 4-103 AutopilotAuthentication

参数	参数类型	描述
mode	String	集群认证模式。默认取值为“rbac”。

表 4-104 AutopilotResourceTag

参数	参数类型	描述
key	String	Key值。 ● 不能为空，最多支持128个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@ ● 不能以"_sys_"开头
value	String	Value值。 ● 可以为空但不能缺省，最多支持255个字符 ● 可用UTF-8格式表示的汉字、字母、数字和空格 ● 支持部分特殊字符：_.:/=+-@

表 4-105 AutopilotClusterExtendParam

参数	参数类型	描述
enterpriseProjectId	String	集群所属的企业项目ID。 **说明** ● 需要开通企业项目功能后才可配置企业项目。
upgradefrom	String	记录集群通过何种升级方式升级到当前版本。

表 4-106 AutopilotPackageConfiguration

参数	参数类型	描述
name	String	组件名称
configurations	Array of **AutopilotConfigurationItem** objects	组件配置项

表 4-107 AutopilotConfigurationItem

参数	参数类型	描述
name	String	组件配置项名称
value	Object	组件配置项值

表 4-108 AutopilotClusterStatus

参数	参数类型	描述
phase	String	集群状态，取值如下 ● Available：可用，表示集群处于正常状态。 ● Unavailable：不可用，表示集群异常，需手动删除。 ● ScalingUp：扩容中，表示集群正处于扩容过程中。 ● ScalingDown：缩容中，表示集群正处于缩容过程中。 ● Creating：创建中，表示集群正处于创建过程中。 ● Deleting：删除中，表示集群正处于删除过程中。 ● Upgrading：升级中，表示集群正处于升级过程中。 ● Resizing：规格变更中，表示集群正处于变更规格中。 ● ResizeFailed：规格变更异常，表示集群变更规格异常。 ● RollingBack：回滚中，表示集群正处于回滚过程中。 ● RollbackFailed：回滚异常，表示集群回滚异常。 ● Hibernating：休眠中，表示集群正处于休眠过程中。 ● Hibernation：已休眠，表示集群正处于休眠状态。 ● Freezing：冻结中，表示集群正处于冻结过程中。 ● Frozen：已冻结，表示集群正处于冻结状态。 ● UnFreezing：解冻中，表示集群正处于解冻过程中。 ● Awaking：唤醒中，表示集群正处于从休眠状态唤醒的过程中。 ● Empty：集群无任何资源（已废弃） ● Error：错误，表示集群资源异常，可尝试手动删除。

参数	参数类型	描述
jobID	String	任务ID,集群当前状态关联的任务ID。当前支持: • 创建集群时返回关联的任务ID，可通过任务ID查询创建集群的附属任务信息； • 删除集群或者删除集群失败时返回关联的任务ID，此字段非空时，可通过任务ID查询删除集群的附属任务信息。 **说明** 任务信息具有一定时效性，仅用于短期跟踪任务进度，请勿用于集群状态判断等额外场景。
reason	String	集群变为当前状态的原因，在集群在非“Available”状态下时，会返回此参数。
message	String	集群变为当前状态的原因的详细信息，在集群在非“Available”状态下时，会返回此参数。
endpoints	Array of **AutopilotClusterEndpoints** objects	集群中 kube-apiserver 的访问地址。
isLocked	Boolean	CBC资源锁定
lockScene	String	CBC资源锁定场景
lockSource	String	锁定资源
lockSourceId	String	锁定的资源ID
deleteOption	Object	删除配置状态（仅删除请求响应包含）
deleteStatus	Object	删除状态信息（仅删除请求响应包含）

表 4-109 AutopilotClusterEndpoints

参数	参数类型	描述
url	String	集群中 kube-apiserver 的访问地址
type	String	集群访问地址的类型 • Internal：用户子网内访问的地址 • External：公网访问的地址

请求示例

无

响应示例

状态码： 200

表示删除指定集群作业下发成功。


```
{
  "kind" : "Cluster",
  "apiVersion" : "v3",
  "metadata" : {
    "name" : "test-cluster",
    "uid" : "a736db34-2270-11ef-a160-0255ac100100",
    "creationTimestamp" : "2024-06-04 12:47:50.886502 +0000 UTC",
    "updateTimestamp" : "2024-06-04 13:24:08.809147153 +0000 UTC",
    "labels" : {
      "FeatureGates" : "arpOptimization,elbv3,xGPU"
    },
    "annotations" : {
      "enableAutopilot" : "true"
    },
    "alias" : "test-cluster"
  },
  "spec" : {
    "category" : "Turbo",
    "type" : "VirtualMachine",
    "flavor" : "cce.autopilot.cluster",
    "version" : "v1.28",
    "platformVersion" : "cce.4.0",
    "hostNetwork" : {
      "vpc" : "c6549063-d459-4ae1-9550-b5fec6741b0f",
      "subnet" : "3b18c2d5-b352-4f59-b421-c2d2d48a1333"
    },
    "containerNetwork" : {
      "mode" : "eni"
    },
    "eniNetwork" : {
      "subnets" : [ {
        "subnetID" : "186f9322-50c5-4e5a-91e3-47da86959afc"
      } ]
    },
    "serviceNetwork" : {
      "IPv4CIDR" : "172.16.0.0/16"
    },
    "authentication" : {
      "mode" : "rbac"
    },
    "billingMode" : 0,
    "kubernetesSvcIpRange" : "172.16.0.0/16",
    "kubeProxyMode" : "iptables",
    "az" : "cn-north-7c",
    "extendParam" : {
      "upgradefrom" : ""
    }
  },
  "status" : {
    "phase" : "Available",
    "jobID" : "ba0c981e-2275-11ef-b73b-0255ac100103",
    "endpoints" : [ {
      "url" : "https://a736db34-2270-11ef-a160-0255ac100100.cluster.cce.cn-
north-7.myhuaweicloud.com:5443",
      "type" : "Internal"
    } ],
    "deleteOption" : {
      "delete_eni" : "delete-block",
      "delete_net" : "delete-block"
    },
    "deleteStatus" : {
      "previous_total" : 47,
      "current_total" : 53,
      "updated" : 47,
      "added" : 6,
      "deleted" : 0
    }
```



```
 }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class DeleteAutopilotClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteAutopilotClusterRequest request = new DeleteAutopilotClusterRequest();
        request.withClusterId("{cluster_id}");
        try {
            DeleteAutopilotClusterResponse response = client.deleteAutopilotCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
```



```
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteAutopilotClusterRequest()
        request.cluster_id = "{cluster_id}"
        response = client.delete_autopilot_cluster(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteAutopilotClusterRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.DeleteAutopilotCluster(request)
```



```
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示删除指定集群作业下发成功。

错误码

请参见**错误码**。

4.1.6 获取集群证书

功能介绍

该API用于获取指定集群的证书信息。

接口约束

该接口适用于1.13及以上集群版本。

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/clustercert

表 4-110 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-111 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-112 请求 Body 参数

参数	是否必选	参数类型	描述
duration	是	Integer	集群证书有效时间，最小值为1天，最大值为5年，因此取值范围为1-1827（以天为单位，实际上限取决于5年内闰年的数量，例如5年内存在一个闰年则上限为1826天）；若填-1则为最大值5年。

响应参数

状态码： 200

表 4-113 响应 Header 参数

参数	参数类型	描述
Port-ID	String	集群控制节点端口ID

表 4-114 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Config”，该值不可修改。
apiVersion	String	API版本，固定值“v1”。
preferences	Object	当前未使用该字段，当前默认为空。
clusters	Array of **Clusters** objects	集群列表。

参数	参数类型	描述
users	Array of **Users** objects	存放了指定用户的一些证书信息和ClientKey信息。
contexts	Array of **Contexts** objects	上下文列表。
current-context	String	当前上下文，若存在publicIp（虚拟机弹性IP）时为 external; 若不存在publicIp为 internal。

表 4-115 Clusters

参数	参数类型	描述
name	String	集群名字。 • 若不存在publicIp（虚拟机弹性IP），则集群列表的集群数量为1，该字段值为“internalCluster”。 • 若存在publicIp，则集群列表的集群数量大于1，所有扩展的cluster的name的值为“externalCluster”。
cluster	**ClusterCert** object	集群信息。

表 4-116 ClusterCert

参数	参数类型	描述
server	String	服务器地址。
certificate-authority-data	String	证书授权数据。
insecure-skip-tls-verify	Boolean	不校验服务端证书，在 cluster 类型为 externalCluster 时，该值为 true。

表 4-117 Users

参数	参数类型	描述
name	String	当前为固定值“user“。
user	**User** object	存放了指定用户的一些证书信息和ClientKey信息。

表 4-118 User

参数	参数类型	描述
client-certificate-data	String	客户端证书。
client-key-data	String	包含来自TLS客户端密钥文件的PEM编码数据。

表 4-119 Contexts

参数	参数类型	描述
name	String	上下文的名称。 • 若不存在publicIp（虚拟机弹性IP），则集群列表的集群数量为1，该字段值为“internal”。 • 若存在publicIp，则集群列表的集群数量大于1，所有扩展的context的name的值为“external”。
context	**Context** object	上下文信息。

表 4-120 Context

参数	参数类型	描述
cluster	String	上下文cluster信息。
user	String	上下文user信息。

请求示例

申请30天有效的集群访问证书

```
{
  "duration" : 30
}
```

响应示例

状态码： 200

表示成功获取指定集群的证书。证书文件格式参见kubernetes v1.Config结构

```
{
  "kind" : "Config",
  "apiVersion" : "v1",
  "preferences" : { },
```

```
 "clusters" : [ {
   "name" : "internalCluster",
   "cluster" : {
     "server" : "https://192.168.1.7:5443",
     "certificate-authority-data" : "Q2VydGlmaWNhdGU6******FTkQgQ0VSVElGSUNBVEUtLS0tLQo="
   }
 } ],
 "users" : [ {
   "name" : "user",
   "user" : {
     "client-certificate-data" : "LS0tLS1CRUdJTiBDR******QVRFLS0tLS0K",
     "client-key-data" : "LS0tLS1CRUdJTi******BLRVktLS0tLQo="
   }
 } ],
 "contexts" : [ {
   "name" : "internal",
   "context" : {
     "cluster" : "internalCluster",
     "user" : "user"
   }
 } ],
 "current-context" : "internal"
}
```

SDK 代码示例

SDK代码示例如下。

Java

申请30天有效的集群访问证书

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class CreateAutopilotKubernetesClusterCertSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAutopilotKubernetesClusterCertRequest request = new CreateAutopilotKubernetesClusterCertRequest();
```



```
        request.withClusterId("{cluster_id}");
        CertDuration body = new CertDuration();
        body.withDuration(30);
        request.withBody(body);
        try {
            CreateAutopilotKubernetesClusterCertResponse response =
client.createAutopilotKubernetesClusterCert(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

申请30天有效的集群访问证书

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAutopilotKubernetesClusterCertRequest()
        request.cluster_id = "{cluster_id}"
        request.body = CertDuration(
            duration=30
        )
        response = client.create_autopilot_kubernetes_cluster_cert(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

申请30天有效的集群访问证书


```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAutopilotKubernetesClusterCertRequest{}
    request.ClusterId = "{cluster_id}"
    request.Body = &model.CertDuration{
        Duration: int32(30),
    }
    response, err := client.CreateAutopilotKubernetesClusterCert(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示成功获取指定集群的证书。证书文件格式参见kubernetes v1.Config结构

错误码

请参见**错误码**。

4.1.7 获取任务信息

功能介绍

该API用于获取任务信息。通过某一任务请求下发后返回的jobID来查询指定任务的进度。

说明

- 集群管理的URL格式为：https://Endpoint/uri。其中uri为资源路径，也即API访问的路径
- 该接口通常使用场景为：
  - 创建、删除集群时，查询相应任务的进度。
  - 创建、删除节点时，查询相应任务的进度。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/jobs/{job_id}

表 4-121 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
job_id	是	String	任务ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-122 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-123 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Job”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**JobMetadata** object	任务元数据。
spec	**JobSpec** object	任务详细参数。
status	**JobStatus** object	任务状态信息。

表 4-124 JobSpec

参数	参数类型	描述
type	String	任务的类型，例：“CreateCluster”- 创建集群。
clusterUID	String	任务所在的集群的ID。
resourceID	String	任务操作的资源ID。
resourceName	String	任务操作的资源名称。
extendParam	Map<String,String>	扩展参数。
subJobs	Array of **Job** objects	子任务的列表。 • 包含了所有子任务的详细信息 • 在创建集群、节点等场景下，通常会由多个子任务共同组成创建任务，在子任务都完成后，任务才会完成

表 4-125 Job

参数	参数类型	描述
kind	String	API类型，固定值“Job”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**JobMetadata** object	任务元数据。
spec	**JobSpec** object	任务详细参数。

参数	参数类型	描述
status	JobStatus object	任务状态信息。

表 4-126 JobMetadata

参数	参数类型	描述
uid	String	任务的ID。
creationTimestamp	String	任务的创建时间。
updateTimestamp	String	任务的更新时间。

表 4-127 JobStatus

参数	参数类型	描述
phase	String	任务的状态，有如下四种状态： • JobPhaseInitializing JobPhase = "Initializing" • JobPhaseRunning JobPhase = "Running" • JobPhaseFailed JobPhase = "Failed" • JobPhaseSuccess JobPhase = "Success"
reason	String	任务变为当前状态的原因

请求示例

无

响应示例

状态码： 200

表示获取任务信息成功。

```
{
  "kind" : "Job",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "354331b2c-229a-11e8-9c75-0255ac100ceb",
    "creationTimestamp" : "2018-08-02 08:12:40.672772389 +0000 UTC",
    "updateTimestamp" : "2018-08-02 08:21:50.478108569 +0000 UTC"
  },
  "spec" : {
    "type" : "CreateCluster",
    "clusterUID" : "4d1ecb2c-229a-11e8-9c75-0255ac100ceb",
    "resourceID" : "6f4dcb2c-229a-11e8-9c75-0255ac100ceb",
```

```
  "resourceName" : "cluster-name",
  "extendParam" : {
    "serverID" : "bc467e3a-2338-11e8-825b-0255ac100c13"
  },
  "subJobs" : [ {
    "kind" : "Job",
    "apiVersion" : "v3",
    "metadata" : {
      "uid" : "fd474fab-9606-11e8-baa9-0255ac10215d",
      "creationTimestamp" : "2018-08-02 03:52:34.615819618 +0000 UTC",
      "updateTimestamp" : "2018-08-02 04:05:29.196243031 +0000 UTC"
    },
    "spec" : {
      "type" : "InstallMaster",
      "clusterUID" : "fcc72de0-9606-11e8-baa8-0255ac10215d",
      "resourceID" : "fd3b4ac0-9606-11e8-baa8-0255ac10215d",
      "extendParam" : {
        "serverID" : "fd3b4ac0-9606-11e8-baa8-0255ac10215d"
      }
    },
    "status" : {
      "phase" : "Success"
    }
  }, {
    "kind" : "Job",
    "apiVersion" : "v3",
    "metadata" : {
      "uid" : "fd474f82-9606-11e8-baa8-0255ac10215d",
      "creationTimestamp" : "2018-08-02 03:52:33.859150791 +0000 UTC",
      "updateTimestamp" : "2018-08-02 03:52:34.615655429 +0000 UTC"
    },
    "spec" : {
      "type" : "CreatePSMCert",
      "clusterUID" : "fcc72de0-9606-11e8-baa8-0255ac10215d"
    },
    "status" : {
      "phase" : "Success"
    }
  } ]
 },
 "status" : {
  "phase" : "Running",
  "reason" : ""
 }
}
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotJobSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
```

```
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotJobRequest request = new ShowAutopilotJobRequest();
        request.withJobId("{job_id}");
        try {
            ShowAutopilotJobResponse response = client.showAutopilotJob(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotJobRequest()
        request.job_id = "{job_id}"
        response = client.show_autopilot_job(request)
        print(response)
    except exceptions.ClientRequestException as e:
```



```
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotJobRequest{}
    request.JobId = "{job_id}"
    response, err := client.ShowAutopilotJob(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取任务信息成功。

错误码

请参见**错误码**。

4.1.8 绑定、解绑集群公网 apiserver 地址

功能介绍

该API用于通过集群ID绑定、解绑集群公网apiserver地址

说明

集群管理的URL格式为：https://Endpoint/uri。其中uri为资源路径，也即API访问的路径。

调用方法

请参见**如何调用API**。

URI

PUT /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/mastereip

表 4-128 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-129 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-130 请求 Body 参数

参数	是否必选	参数类型	描述
spec	是	**MasterEIPRequestSpec** object	绑定、解绑集群公网apiserver地址的请求配置参数

表 4-131 MasterEIPRequestSpec

参数	是否必选	参数类型	描述
action	否	String	绑定或解绑动作，必选参数。 • 绑定：固定值为{"action":"bind"} • 解绑：固定值为{"action":"unbind"}
spec	否	spec object	待绑定的弹性IP配置属性
bandwidth	否	String	带宽(字段已失效，暂不推荐使用)
elasticIp	否	String	弹性网卡IP(字段已失效，暂不推荐使用)

表 4-132 spec

参数	是否必选	参数类型	描述
id	否	String	弹性网卡ID，绑定时必选，解绑时该字段无效

响应参数

状态码： 200

表 4-133 响应 Body 参数

参数	参数类型	描述
metadata	Metadata object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	MasterEIPResponseSpec object	绑定集群公网apiserver地址的配置信息
status	status object	状态信息

表 4-134 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称

参数	参数类型	描述
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-135 MasterEIPResponseSpec

参数	参数类型	描述
action	String	绑定动作
spec	**spec** object	待绑定的弹性IP配置属性
elasticIp	String	弹性公网IP

表 4-136 spec

参数	参数类型	描述
id	String	弹性网卡ID
eip	**EipSpec** object	EIP的详细信息
IsDynamic	Boolean	是否动态创建

表 4-137 EipSpec

参数	参数类型	描述
bandwidth	**bandwidth** object	带宽信息

表 4-138 bandwidth

参数	参数类型	描述
size	Integer	带宽大小

参数	参数类型	描述
sharetype	String	带宽类型

表 4-139 status

参数	参数类型	描述
privateEndpoint	String	集群访问的PrivateIP(HA集群返回VIP)
publicEndpoint	String	集群访问的PublicIP

请求示例

绑定集群公网apiserver地址。

```
{
  "spec" : {
    "action" : "bind",
    "spec" : {
      "id" : "a757a69e-f920-455a-b1ba-d7a22db0fd51"
    }
  }
}
```

响应示例

状态码： 200

表示绑定集群公网apiserver地址成功，解绑成功无响应体。

```
{
  "metadata" : { },
  "spec" : {
    "action" : "bind",
    "spec" : {
      "id" : "a757a69e-f920-455a-b1ba-d7a22db0fd50",
      "eip" : {
        "bandwidth" : {
          "size" : 5,
          "sharetype" : "PER"
        }
      },
      "IsDynamic" : false
    },
    "elasticIp" : "8.8.8.8"
  },
  "status" : {
    "privateEndpoint" : "https://192.168.3.238:5443",
    "publicEndpoint" : "https://8.8.8.8:5443"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

绑定集群公网apiserver地址。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class UpdateAutopilotClusterEipSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateAutopilotClusterEipRequest request = new UpdateAutopilotClusterEipRequest();
        request.withClusterId("{cluster_id}");
        MasterEIPRequest body = new MasterEIPRequest();
        MasterEIPRequestSpecSpec specSpec = new MasterEIPRequestSpecSpec();
        specSpec.withId("a757a69e-f920-455a-b1ba-d7a22db0fd51");
        MasterEIPRequestSpec specbody = new MasterEIPRequestSpec();
        specbody.withAction(MasterEIPRequestSpec.ActionEnum.fromValue("bind"))
            .withSpec(specSpec);
        body.withSpec(specbody);
        request.withBody(body);
        try {
            UpdateAutopilotClusterEipResponse response = client.updateAutopilotClusterEip(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

绑定集群公网apiserver地址。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateAutopilotClusterEipRequest()
        request.cluster_id = "{cluster_id}"
        specSpec = MasterEIPRequestSpecSpec(
            id="a757a69e-f920-455a-b1ba-d7a22db0fd51"
        )
        specbody = MasterEIPRequestSpec(
            action="bind",
            spec=specSpec
        )
        request.body = MasterEIPRequest(
            spec=specbody
        )
        response = client.update_autopilot_cluster_eip(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

绑定集群公网apiserver地址。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"
```

```
    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateAutopilotClusterEipRequest{}
    request.ClusterId = "{cluster_id}"
    idSpec:= "a757a69e-f920-455a-b1ba-d7a22db0fd51"
    specSpec := &model.MasterEipRequestSpecSpec{
        Id: &idSpec,
    }
    actionSpec:= model.GetMasterEipRequestSpecActionEnum().BIND
    specbody := &model.MasterEipRequestSpec{
        Action: &actionSpec,
        Spec: specSpec,
    }
    request.Body = &model.MasterEipRequest{
        Spec: specbody,
    }
    response, err := client.UpdateAutopilotClusterEip(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示绑定集群公网apiserver地址成功，解绑成功无响应体。

错误码

请参见**错误码**。

4.1.9 获取集群访问的地址

功能介绍

该API用于通过集群ID获取集群访问的地址，包括PrivateIP与PublicIP

说明

集群管理的URL格式为：https://Endpoint/uri。其中uri为资源路径，也即API访问的路径。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/openapi

表 4-140 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-141 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-142 响应 Body 参数

参数	参数类型	描述
metadata	**Metadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	**OpenAPISpec** object	集群访问地址的配置参数信息
status	**status** object	状态信息

表 4-143 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-144 OpenAPISpec

参数	参数类型	描述
spec	**spec** object	集群访问的地址

表 4-145 spec

参数	参数类型	描述
eip	**EipSpec** object	EIP的详细信息
IsDynamic	Boolean	是否动态创建

表 4-146 EipSpec

参数	参数类型	描述
bandwidth	**bandwidth** object	带宽信息

表 4-147 bandwidth

参数	参数类型	描述
size	Integer	带宽大小
sharetype	String	带宽类型

表 4-148 status

参数	参数类型	描述
privateEndpoint	String	集群访问的PrivateIP(HA集群返回VIP)
publicEndpoint	String	集群访问的PublicIP

请求示例

无

响应示例

状态码： 200

表示获取集群访问的地址成功。

```
{
  "metadata" : { },
  "spec" : {
    "spec" : {
      "eip" : {
        "bandwidth" : { }
      },
      "IsDynamic" : false
    }
  },
  "status" : {
    "privateEndpoint" : "https://192.168.3.238:5443",
    "publicEndpoint" : ""
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotClusterEndpointsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
```

```
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotClusterEndpointsRequest request = new ShowAutopilotClusterEndpointsRequest();
        request.withClusterId("{cluster_id}");
        try {
            ShowAutopilotClusterEndpointsResponse response = client.showAutopilotClusterEndpoints(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotClusterEndpointsRequest()
        request.cluster_id = "{cluster_id}"
        response = client.show_autopilot_cluster_endpoints(request)
        print(response)
    except exceptions.ClientRequestException as e:
```



```
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotClusterEndpointsRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ShowAutopilotClusterEndpoints(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群访问的地址成功。

错误码

请参见**错误码**。

4.2 插件管理（Autopilot）

4.2.1 创建 AddonInstance

功能介绍

根据提供的插件模板，安装插件实例。

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/addons

请求参数

表 4-149 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-150 请求 Body 参数

参数	是否必选	参数类型	描述
kind	是	String	API类型，固定值“Addon”，该值不可修改，该字段传入无效。
apiVersion	是	String	API版本，固定值“v3”，该值不可修改，该字段传入无效。
metadata	是	**AddonMetadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	是	**InstanceRequestSpec** object	spec是集合类的元素类型，内容为插件实例安装/升级的具体请求信息

表 4-151 AddonMetadata

参数	是否必选	参数类型	描述
uid	否	String	唯一id标识
name	否	String	插件名称
alias	否	String	插件别名
labels	否	Map<String,String>	插件标签，key/value对格式，接口保留字段，填写不会生效
annotations	否	Map<String,String>	插件注解，由key/value组成 • 安装：固定值为{"addon.install/type":"install"} • 升级：固定值为{"addon.upgrade/type":"upgrade"}
updateTimestamp	否	String	更新时间
creationTimestamp	否	String	创建时间

表 4-152 InstanceRequestSpec

参数	是否必选	参数类型	描述
version	否	String	待安装、升级插件的版本号，例如1.0.0 • 安装：该参数非必传，如果不传，匹配集群支持的最新版本 • 升级：该参数必传，需指定版本号
clusterID	是	String	集群id
values	是	Map<String,Object>	插件模板安装参数（各插件不同），升级插件时需要填写全量安装参数，未填写参数将使用插件模板中的默认值，当前插件安装参数可通过查询插件实例接口获取。
addonTemplateName	是	String	待安装插件模板名称，如coredns

响应参数

状态码： 201

表 4-153 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Addon”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**AddonMetadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	**InstanceSpec** object	spec是集合类的元素类型，内容为插件实例具体信息，实例的详细描述主体部分都在spec中给出
status	**AddonInstanceStatus** object	插件实例状态

表 4-154 AddonMetadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	插件名称
alias	String	插件别名
labels	Map<String,String>	插件标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	插件注解，由key/value组成 • 安装：固定值为{"addon.install/type":"install"} • 升级：固定值为{"addon.upgrade/type":"upgrade"}
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-155 InstanceSpec

参数	参数类型	描述
clusterID	String	集群id

参数	参数类型	描述
version	String	插件模板版本号，如1.0.0
addonTemplateName	String	插件模板名称，如coredns
addonTemplateType	String	插件模板类型
addonTemplateLogo	String	插件模板logo图片的地址
addonTemplateLabels	Array of strings	插件模板所属类型
description	String	插件模板描述
values	Map<String,Object>	插件模板安装参数（各插件不同），请根据具体插件模板信息填写安装参数。

表 4-156 AddonInstanceStatus

参数	参数类型	描述
status	String	插件实例状态, 取值如下 ● running：运行中，表示插件全部实例状态都在运行中，插件正常使用。 ● abnormal：不可用，表示插件状态异常，插件不可使用。可单击插件名称查看实例异常事件。 ● installing：安装中，表示插件正在安装中。 ● installFailed：安装失败，表示插件安装失败，需要卸载后重新安装。 ● upgrading：升级中，表示插件正在更新中。 ● upgradeFailed：升级失败，表示插件升级失败，可重试升级或卸载后重新安装。 ● deleting：删除中，表示插件正在删除中。 ● deleteFailed：删除失败，表示插件删除失败，可重试卸载。 ● deleteSuccess：删除成功，表示插件删除成功。 ● available：部分就绪，表示插件下只有部分实例状态为运行中，插件部分功能可用。 ● rollbacking：回滚中，表示插件正在回滚中。 ● rollbackFailed：回滚失败，表示插件回滚失败，可重试回滚或卸载后重新安装。 ● unknown：未知状态，表示插件模板实例不存在。
Reason	String	插件安装失败原因
message	String	安装错误详情
targetVersions	Array of strings	此插件版本，支持升级的集群版本
currentVersion	**Versions** object	当前插件实例使用的具体插件版本信息
isRollbackable	Boolean	是否支持回滚到插件升级前的插件版本
previousVersion	String	插件升级或回滚前的版本

表 4-157 Versions

参数	参数类型	描述
version	String	插件版本号
input	Object	插件安装参数
stable	Boolean	是否为稳定版本
translate	Object	供界面使用的翻译信息
supportVersions	Array of **SupportVersions** objects	支持集群版本号
creationTimestamp	String	创建时间
updateTimestamp	String	更新时间

表 4-158 SupportVersions

参数	参数类型	描述
clusterType	String	支持的集群类型
clusterVersion	Array of strings	支持的集群版本（正则表达式）

请求示例

安装1.28.6版本的coredns插件，插件实例数指定为2。

```
{
  "kind" : "Addon",
  "apiVersion" : "v3",
  "metadata" : {
    "annotations" : {
      "addon.install/type" : "install"
    }
  },
  "spec" : {
    "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
    "version" : "1.28.6",
    "addonTemplateName" : "coredns",
    "values" : {
      "basic" : {
        "cluster_ip" : "10.247.3.10",
        "image_version" : "1.28.6",
        "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
        "swr_user" : "autopilot-official",
        "rbac_enabled" : true,
        "cluster_version" : "v1.28"
      },
      "flavor" : {
        "category" : [ "Autopilot" ],
        "is_default" : true,
```

```
    "name" : "autopilot-flavor1",
    "replicas" : 2,
    "resources" : [ {
      "id" : "coredns",
      "name" : "coredns",
      "limitsCpu" : "1000m",
      "requestsCpu" : "1000m",
      "limitsMem" : "2048Mi",
      "requestsMem" : "2048Mi"
    } ]
  },
  "custom" : {
    "multiAZBalance" : false,
    "multiAZEnabled" : false,
    "node_match_expressions" : [ ],
    "parameterSyncStrategy" : "ensureConsistent",
    "servers" : [ {
      "plugins" : [ {
        "name" : "bind",
        "parameters" : "{$POD_IP}"
      }, {
        "configBlock" : "servfail 5s",
        "name" : "cache",
        "parameters" : 30
      }, {
        "name" : "errors"
      }, {
        "name" : "health",
        "parameters" : "{$POD_IP}:8080"
      }, {
        "name" : "ready",
        "parameters" : "{$POD_IP}:8081"
      }, {
        "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
        "name" : "kubernetes",
        "parameters" : "cluster.local in-addr.arpa ip6.arpa"
      }, {
        "name" : "loadbalance",
        "parameters" : "round_robin"
      }, {
        "name" : "prometheus",
        "parameters" : "{$POD_IP}:9153"
      }, {
        "configBlock" : "policy random",
        "name" : "forward",
        "parameters" : ". /etc/resolv.conf"
      }, {
        "name" : "reload"
      } ],
      "port" : 5353,
      "zones" : [ {
        "zone" : "."
      } ]
    } ],
    "stub_domains" : { },
    "tolerations" : [ {
      "key" : "node.kubernetes.io/not-ready",
      "operator" : "Exists",
      "effect" : "NoExecute",
      "tolerationSeconds" : 60
    }, {
      "key" : "node.kubernetes.io/unreachable",
      "operator" : "Exists",
      "effect" : "NoExecute",
      "tolerationSeconds" : 60
    } ],
    "upstream_nameservers" : [ ]
  }
}
```

```
  }
}
```

响应示例

状态码： 201

OK


```
{
  "kind" : "Addon",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "90b775e0-5774-4e1d-ab3b-516332ba047a",
    "name" : "coredns",
    "alias" : "coredns",
    "creationTimestamp" : "2024-07-18T04:04:21Z",
    "updateTimestamp" : "2024-07-18T04:04:21Z"
  },
  "spec" : {
    "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
    "version" : "1.28.6",
    "addonTemplateName" : "coredns",
    "addonTemplateType" : "helm",
    "addonTemplateLogo" : "",
    "addonTemplateLabels" : [ "ContainerNetwork" ],
    "description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS Services",
    "values" : {
      "basic" : {
        "cluster_ip" : "10.247.3.10",
        "cluster_version" : "v1.28",
        "image_version" : "1.28.6",
        "platform" : "linux-amd64",
        "rbac_enabled" : true,
        "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
        "swr_user" : "autopilot-official"
      },
      "custom" : {
        "multiAZBalance" : false,
        "multiAZEnabled" : false,
        "node_match_expressions" : [ ],
        "parameterSyncStrategy" : "ensureConsistent",
        "servers" : [ {
          "plugins" : [ {
            "name" : "bind",
            "parameters" : "{$POD_IP}"
          }, {
            "configBlock" : "servfail 5s",
            "name" : "cache",
            "parameters" : 30
          }, {
            "name" : "errors"
          }, {
            "name" : "health",
            "parameters" : "{$POD_IP}:8080"
          }, {
            "name" : "ready",
            "parameters" : "{$POD_IP}:8081"
          }, {
            "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
            "name" : "kubernetes",
            "parameters" : "cluster.local in-addr.arpa ip6.arpa"
          }, {
            "name" : "loadbalance",
            "parameters" : "round_robin"
          }, {
            "name" : "prometheus",
            "parameters" : "{$POD_IP}:9153"
          }, {
```

```
          "configBlock" : "policy random",
          "name" : "forward",
          "parameters" : ". /etc/resolv.conf"
        }, {
          "name" : "reload"
        } ],
        "port" : 5353,
        "zones" : [ {
          "zone" : "."
        } ]
      } ],
      "stub_domains" : { },
      "tolerations" : [ {
        "effect" : "NoExecute",
        "key" : "node.kubernetes.io/not-ready",
        "operator" : "Exists",
        "tolerationSeconds" : 60
      }, {
        "effect" : "NoExecute",
        "key" : "node.kubernetes.io/unreachable",
        "operator" : "Exists",
        "tolerationSeconds" : 60
      } ],
      "upstream_nameservers" : [ ]
    },
    "flavor" : {
      "category" : [ "Autopilot" ],
      "is_default" : true,
      "name" : "autopilot-flavor1",
      "replicas" : 2,
      "resources" : [ {
        "id" : "coredns",
        "limitsCpu" : "1000m",
        "limitsMem" : "2048Mi",
        "name" : "coredns",
        "requestsCpu" : "1000m",
        "requestsMem" : "2048Mi"
      } ]
    },
    "systemAutoInject" : {
      "cluster" : {
        "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
        "clusterNetworkMode" : "eni",
        "clusterVersion" : "v1.28.5-r0"
      },
      "user" : {
        "projectID" : "47eb1d64cbeb45cfa01ae20af4f4b563"
      }
    }
  }
},
"status" : {
  "status" : "installing",
  "Reason" : "",
  "message" : "",
  "targetVersions" : null,
  "isRollbackable" : false,
  "currentVersion" : {
    "version" : "1.28.6",
    "input" : {
      "basic" : {
        "cluster_ip" : "10.247.3.10",
        "image_version" : "1.28.6",
        "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
        "swr_user" : "autopilot-official"
      },
      "parameters" : {
        "autopilot-flavor1" : {
          "category" : [ "Autopilot" ],
```

```
  "is_default" : true,
  "name" : "autopilot-flavor1",
  "replicas" : 2,
  "resources" : [ {
    "limitsCpu" : 1,
    "limitsMem" : "2Gi",
    "name" : "coredns",
    "requestsCpu" : 1,
    "requestsMem" : "2Gi"
  } ]
},
"custom" : {
  "multiAZBalance" : false,
  "multiAZEnabled" : false,
  "node_match_expressions" : [ ],
  "parameterSyncStrategy" : "ensureConsistent",
  "servers" : [ {
    "plugins" : [ {
      "name" : "bind",
      "parameters" : "{$POD_IP}"
    }, {
      "configBlock" : "servfail 5s",
      "name" : "cache",
      "parameters" : 30
    }, {
      "name" : "errors"
    }, {
      "name" : "health",
      "parameters" : "{$POD_IP}:8080"
    }, {
      "name" : "ready",
      "parameters" : "{$POD_IP}:8081"
    }, {
      "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
      "name" : "kubernetes",
      "parameters" : "cluster.local in-addr.arpa ip6.arpa"
    }, {
      "name" : "loadbalance",
      "parameters" : "round_robin"
    }, {
      "name" : "prometheus",
      "parameters" : "{$POD_IP}:9153"
    }, {
      "configBlock" : "policy random",
      "name" : "forward",
      "parameters" : ". /etc/resolv.conf"
    }, {
      "name" : "reload"
    } ],
    "port" : 5353,
    "zones" : [ {
      "zone" : "."
    } ]
  } ],
  "stub_domains" : { },
  "tolerations" : [ {
    "effect" : "NoExecute",
    "key" : "node.kubernetes.io/not-ready",
    "operator" : "Exists",
    "tolerationSeconds" : 60
  }, {
    "effect" : "NoExecute",
    "key" : "node.kubernetes.io/unreachable",
    "operator" : "Exists",
    "tolerationSeconds" : 60
  } ],
  "upstream_nameservers" : [ ]
},
"flavor1" : {
```

```
          "is_default" : true,
          "name" : 2500,
          "recommend_cluster_flavor_types" : [ "small" ],
          "replicas" : 2,
          "resources" : [ {
            "limitsCpu" : "500m",
            "limitsMem" : "512Mi",
            "name" : "coredns",
            "requestsCpu" : "500m",
            "requestsMem" : "512Mi"
          } ]
        },
        "flavor2" : {
          "name" : 5000,
          "recommend_cluster_flavor_types" : [ "medium" ],
          "replicas" : 2,
          "resources" : [ {
            "limitsCpu" : "1000m",
            "limitsMem" : "1024Mi",
            "name" : "coredns",
            "requestsCpu" : "1000m",
            "requestsMem" : "1024Mi"
          } ]
        },
        "flavor3" : {
          "name" : 10000,
          "recommend_cluster_flavor_types" : [ "large" ],
          "replicas" : 2,
          "resources" : [ {
            "limitsCpu" : "2000m",
            "limitsMem" : "2048Mi",
            "name" : "coredns",
            "requestsCpu" : "2000m",
            "requestsMem" : "2048Mi"
          } ]
        },
        "flavor4" : {
          "name" : 20000,
          "recommend_cluster_flavor_types" : [ "xlarge" ],
          "replicas" : 4,
          "resources" : [ {
            "limitsCpu" : "2000m",
            "limitsMem" : "2048Mi",
            "name" : "coredns",
            "requestsCpu" : "2000m",
            "requestsMem" : "2048Mi"
          } ]
        }
      }
    },
    "stable" : true,
    "translate" : {
      "en_US" : {
        "addon" : {
          "changeLog" : "Support autopilot cluster",
          "description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS Services"
        },
        "description" : {
          "Parameters.custom.stub_domains" : "The target nameserver may itself be a Kubernetes service. For instance, you can run your own copy of dnsmasq to export custom DNS names into the ClusterDNS namespace, a JSON map using a DNS suffix key (e.g. “acme.local”) and a value consisting of a JSON array of DNS IPs.",
          "Parameters.custom.upstream_nameservers" : "If specified, then the values specified replace the nameservers taken by default from the node’s /etc/resolv.conf. Limits:a maximum of three upstream nameservers can be specified, A JSON array of DNS IPs.",
          "Parameters.flavor1.description" : "Concurrent domain name resolution ability - External domain name：2500 qps, Internal domain name：10000 qps",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "Concurrent domain name resolution ability - External domain
```



```
name：5000 qps, Internal domain name：20000 qps",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "Concurrent domain name resolution ability  -  External domain
name：10000 qps, Internal domain name：40000 qps",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "Concurrent domain name resolution ability  -  External domain
name：20000 qps, Internal domain name：80000 qps",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "stub domain",
          "Parameters.custom.upstream_nameservers" : "upstream nameservers"
        }
      },
      "fr_FR" : {
        "addon" : {
          "changeLog" : "les spécifications du plugin peuvent être associées aux spécifications du cluster. le
fuseau horaire du plug-in est le même que celui du noeud",
          "description" : "Un serveur DNS qui enchaîne les plug-ins et fournit des services DNS Kubernetes."
        },
        "description" : {
          "Parameters.custom.stub_domains" : "Le serveur de noms cible peut lui-même être un service
Kubernetes. Par exemple, vous pouvez exécuter votre propre copie de dnsmasq pour exporter des noms
DNS personnalisés dans l'espace de noms ClusterDNS, une carte JSON à l'aide d'une clé de suffixe DNS (par
exemple, «acme.local») et une valeur constituée d'un tableau JSON d'adresses IP DNS.",
          "Parameters.custom.upstream_nameservers" : "Si spécifié, les valeurs spécifiées remplacent les
serveurs de noms pris par défaut dans le fichier /etc/resolv.conf du nœud. Limites: un maximum de trois
serveurs de noms en amont peuvent être spécifiés, un tableau JSON d'adresses IP DNS.",
          "Parameters.flavor1.description" : "Capacité de résolution de nom de domaine simultanée - Nom de
domaine externe: 2500 qps, Nom de domaine interne: 10000 qp",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "Capacité de résolution de nom de domaine simultanée - Nom de
domaine externe: 5000 qps, Nom de domaine interne: 20000 qp",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "Capacité de résolution de nom de domaine simultanée - Nom de
domaine externe: 10000 qps, Nom de domaine interne: 40000 qp",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "Capacité de résolution de nom de domaine simultanée - Nom de
domaine externe: 20000 qps, Nom de domaine interne: 80000 qp",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "domaine stub",
          "Parameters.custom.upstream_nameservers" : "serveurs de noms en amont"
        }
      },
      "zh_CN" : {
        "addon" : {
          "changeLog" : "支持autopilot集群",
          "description" : "CoreDNS是一款通过链式插件的方式给Kubernetes提供DNS解析服务的DNS服务器"
        },
        "description" : {
          "Parameters.custom.stub_domains" : "用户可对自定义的域名配置域名服务器，格式为一个键值对，键
为DNS后缀域名，值为一个或一组DNS IP地址，如\"acme.local -- 1.2.3.4,6.7.8.9\"。",
          "Parameters.custom.upstream_nameservers" : "解析除集群内服务域名以及自定义域名之外的域名地
址，格式为一个或一组DNS IP地址，如\"8.8.8.8\",\"8.8.4.4\"。",
          "Parameters.flavor1.description" : "并发域名解析能力 - 外部域名：2500 qps，内部域名：10000 qps",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "并发域名解析能力 - 外部域名：5000 qps，内部域名：20000 qps",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "并发域名解析能力 - 外部域名：10000 qps，内部域名：40000
qps",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "并发域名解析能力 - 外部域名：20000 qps，内部域名：80000
qps",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "存根域",
```

```
          "Parameters.custom.upstream_nameservers" : "上游域名服务器"
        }
       }
      },
      "supportVersions" : null,
      "creationTimestamp" : "2024-02-19T11:33:46Z",
      "updateTimestamp" : "2024-02-21T01:24:05Z"
    }
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

安装1.28.6版本的coredns插件，插件实例数指定为2。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;

import java.util.Map;
import java.util.HashMap;

public class CreateAutopilotAddonInstanceSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAutopilotAddonInstanceRequest request = new CreateAutopilotAddonInstanceRequest();
        InstanceRequest body = new InstanceRequest();
        Map<String, Object> listSpecValues = new HashMap<>();
        listSpecValues.put("basic", "{\"cluster_version\":\"v1.28\",\"rbac_enabled\":true,\"swr_user\":\"autopilot-official\",\"image_version\":\"1.28.6\",\"cluster_ip\":\"10.247.3.10\",\"swr_addr\":\"swr.cn-north-7.myhuaweicloud.com\"}");
        listSpecValues.put("flavor", "{\"replicas\":2,\"name\":\"autopilot-flavor1\",\"resources\":[{\"limitsCpu\":\"1000m\",\"name\":\"coredns\",\"id\":\"coredns\",\"limitsMem\":\"2048Mi\",\"requestsMem\":\"2048Mi\",\"requestsCpu\":\"1000m\"}],\"category\":[\"Autopilot\"],\"is_default\":true}");
        listSpecValues.put("custom", "{\"servers\":[{\"port\":5353,\"plugins\":[{\"name\":\"bind\",\"parameters\":\"{$POD_IP}\"},{\"configBlock\":\"servfail 5s\",\"name\":\"cache\",\"parameters\":30},{\"name\":\"errors\"},{\"name\":\"health\",\"parameters\":\"{$POD_IP}:8080\"},{\"name\":\"ready\",\"parameters\":\"{$POD_IP}:8081\"},{\"configBlock\":\"pods insecure\nfallthrough in-addr.arpa ip6.arpa\",\"name\":\"kubernetes\",\"parameters\":\"cluster.local in-addr.arpa ip6.arpa\"},{\"name\":\"loadbalance\",\"parameters\":\"round_robin\"},{\"name\":\"prometheus\",\"parameters\":\"{$POD_IP}:9153\"},
```



```
{\"configBlock\":\"policy random\",\"name\":\"forward\",\"parameters\":\". /etc/resolv.conf\"},{\"name
\":\"reload\"}],\"zones\":[{\"zone\":\".\"}]}],\"tolerations\":[{\"effect\":\"NoExecute\",\"tolerationSeconds
\":60,\"key\":\"node.kubernetes.io/not-ready\",\"operator\":\"Exists\"},{\"effect\":\"NoExecute
\",\"tolerationSeconds\":60,\"key\":\"node.kubernetes.io/unreachable\",\"operator\":\"Exists
\"}],\"multiAZBalance\":false,\"node_match_expressions\":[],\"stub_domains\":{},\"multiAZEnabled
\":false,\"parameterSyncStrategy\":\"ensureConsistent\",\"upstream_nameservers\":[]}");
        InstanceRequestSpec specbody = new InstanceRequestSpec();
        specbody.withVersion("1.28.6")
            .withClusterID("597f2d95-44ab-11ef-9e39-0255ac100115")
            .withValues(listSpecValues)
            .withAddonTemplateName("coredns");
        Map<String, String> listMetadataAnnotations = new HashMap<>();
        listMetadataAnnotations.put("addon.install/type", "install");
        AddonMetadata metadatabody = new AddonMetadata();
        metadatabody.withAnnotations(listMetadataAnnotations);
        body.withSpec(specbody);
        body.withMetadata(metadatabody);
        body.withApiVersion("v3");
        body.withKind("Addon");
        request.withBody(body);
        try {
            CreateAutopilotAddonInstanceResponse response = client.createAutopilotAddonInstance(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

安装1.28.6版本的coredns插件，插件实例数指定为2。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAutopilotAddonInstanceRequest()
        listValuesSpec = {
            "basic": "{\"cluster_version\":\"v1.28\",\"rbac_enabled\":true,\"swr_user\":\"autopilot-official
```

```
\",\"image_version\":\"1.28.6\",\"cluster_ip\":\"10.247.3.10\",\"swr_addr\":\"swr.cn-
north-7.myhuaweicloud.com\"}",
            "flavor": "{\"replicas\":2,\"name\":\"autopilot-flavor1\",\"resources\":[{\"limitsCpu\":\"1000m
\",\"name\":\"coredns\",\"id\":\"coredns\",\"limitsMem\":\"2048Mi\",\"requestsMem\":\"2048Mi
\",\"requestsCpu\":\"1000m\"}],\"category\":[\"Autopilot\"],\"is_default\":true}",
            "custom": "{\"servers\":[{\"port\":5353,\"plugins\":[{\"name\":\"bind\",\"parameters\":\"{$POD_IP}
\"},{\"configBlock\":\"servfail 5s\",\"name\":\"cache\",\"parameters\":30},{\"name\":\"errors\"},{\"name
\":\"health\",\"parameters\":\"{$POD_IP}:8080\"},{\"name\":\"ready\",\"parameters\":\"{$POD_IP}:8081\"},
{\"configBlock\":\"pods insecure\nfallthrough in-addr.arpa ip6.arpa\",\"name\":\"kubernetes\",\"parameters
\":\"cluster.local in-addr.arpa ip6.arpa\"},{\"name\":\"loadbalance\",\"parameters\":\"round_robin\"},{\"name
\":\"prometheus\",\"parameters\":\"{$POD_IP}:9153\"},{\"configBlock\":\"policy random\",\"name\":\"forward
\",\"parameters\":\". /etc/resolv.conf\"},{\"name\":\"reload\"}],\"zones\":[{\"zone\":\".\"}]}],\"tolerations\":
[{\"effect\":\"NoExecute\",\"tolerationSeconds\":60,\"key\":\"node.kubernetes.io/not-ready\",\"operator
\":\"Exists\"},{\"effect\":\"NoExecute\",\"tolerationSeconds\":60,\"key\":\"node.kubernetes.io/unreachable
\",\"operator\":\"Exists\"}],\"multiAZBalance\":false,\"node_match_expressions\":[],\"stub_domains\":
{},\"multiAZEnabled\":false,\"parameterSyncStrategy\":\"ensureConsistent\",\"upstream_nameservers\":[]}"
        }
        specbody = InstanceRequestSpec(
            version="1.28.6",
            cluster_id="597f2d95-44ab-11ef-9e39-0255ac100115",
            values=listValuesSpec,
            addon_template_name="coredns"
        )
        listAnnotationsMetadata = {
            "addon.install/type": "install"
        }
        metadatabody = AddonMetadata(
            annotations=listAnnotationsMetadata
        )
        request.body = InstanceRequest(
            spec=specbody,
            metadata=metadatabody,
            api_version="v3",
            kind="Addon"
        )
        response = client.create_autopilot_addon_instance(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

安装1.28.6版本的coredns插件，插件实例数指定为2。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
```

```
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAutopilotAddonInstanceRequest{}
    var listValuesSpec = map[string]interface{}{
        "basic": "{\"cluster_version\":\"v1.28\",\"rbac_enabled\":true,\"swr_user\":\"autopilot-official
\",\"image_version\":\"1.28.6\",\"cluster_ip\":\"10.247.3.10\",\"swr_addr\":\"swr.cn-
north-7.myhuaweicloud.com\"}",
        "flavor": "{\"replicas\":2,\"name\":\"autopilot-flavor1\",\"resources\":[{\"limitsCpu\":\"1000m\",\"name
\":\"coredns\",\"id\":\"coredns\",\"limitsMem\":\"2048Mi\",\"requestsMem\":\"2048Mi\",\"requestsCpu
\":\"1000m\"}],\"category\":[\"Autopilot\"],\"is_default\":true}",
        "custom": "{\"servers\":[{\"port\":5353,\"plugins\":[{\"name\":\"bind\",\"parameters\":\"{$POD_IP}\"},
{\"configBlock\":\"servfail 5s\",\"name\":\"cache\",\"parameters\":30},{\"name\":\"errors\"},{\"name
\":\"health\",\"parameters\":\"{$POD_IP}:8080\"},{\"name\":\"ready\",\"parameters\":\"{$POD_IP}:8081\"},
{\"configBlock\":\"pods insecure\nfallthrough in-addr.arpa ip6.arpa\",\"name\":\"kubernetes\",\"parameters
\":\"cluster.local in-addr.arpa ip6.arpa\"},{\"name\":\"loadbalance\",\"parameters\":\"round_robin\"},{\"name
\":\"prometheus\",\"parameters\":\"{$POD_IP}:9153\"},{\"configBlock\":\"policy random\",\"name\":\"forward
\",\"parameters\":\". /etc/resolv.conf\"},{\"name\":\"reload\"}],\"zones\":[{\"zone\":\".\"}]}],\"tolerations\":
[{\"effect\":\"NoExecute\",\"tolerationSeconds\":60,\"key\":\"node.kubernetes.io/not-ready\",\"operator
\":\"Exists\"},{\"effect\":\"NoExecute\",\"tolerationSeconds\":60,\"key\":\"node.kubernetes.io/unreachable
\",\"operator\":\"Exists\"}],\"multiAZBalance\":false,\"node_match_expressions\":[],\"stub_domains\":
{},\"multiAZEnabled\":false,\"parameterSyncStrategy\":\"ensureConsistent\",\"upstream_nameservers\":[]}",
    }
    versionSpec:= "1.28.6"
    specbody := &model.InstanceRequestSpec{
        Version: &versionSpec,
        ClusterID: "597f2d95-44ab-11ef-9e39-0255ac100115",
        Values: listValuesSpec,
        AddonTemplateName: "coredns",
    }
    var listAnnotationsMetadata = map[string]string{
        "addon.install/type": "install",
    }
    metadatabody := &model.AddonMetadata{
        Annotations: listAnnotationsMetadata,
    }
    request.Body = &model.InstanceRequest{
        Spec: specbody,
        Metadata: metadatabody,
        ApiVersion: "v3",
        Kind: "Addon",
    }
    response, err := client.CreateAutopilotAddonInstance(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	OK

错误码

请参见**错误码**。

4.2.2 查询 AddonTemplates 列表

功能介绍

插件模板查询接口，查询插件信息。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/addontemplates

表 4-159 Query 参数

参数	是否必选	参数类型	描述
addon_template_name	否	String	指定的插件名称或插件别名，不填写则查询列表。

请求参数

表 4-160 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-161 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Addon”，该值不可修改。

参数	参数类型	描述
apiVersion	String	API版本，固定值“v3”，该值不可修改。
items	Array of **AddonTemplate** objects	插件模板列表

表 4-162 AddonTemplate

参数	参数类型	描述
kind	String	API类型，固定值“Addon”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**AddonMetadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	**Templatespec** object	spec是集合类的元素类型，内容为插件模板具体信息，插件模板的详细描述主体部分都在spec中给出

表 4-163 AddonMetadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	插件名称
alias	String	插件别名
labels	Map<String,String>	插件标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	插件注解，由key/value组成 • 安装：固定值为{"addon.install/type":"install"} • 升级：固定值为{"addon.upgrade/type":"upgrade"}
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-164 Templatespec

参数	参数类型	描述
type	String	模板类型（helm，static）
require	Boolean	是否为必安装插件
labels	Array of strings	模板所属分组
logoURL	String	Logo图片地址
readmeURL	String	插件详情描述及使用说明
description	String	模板描述
versions	Array of **Versions** objects	模板具体版本详情

表 4-165 Versions

参数	参数类型	描述
version	String	插件版本号
input	Object	插件安装参数
stable	Boolean	是否为稳定版本
translate	Object	供界面使用的翻译信息
supportVersions	Array of **SupportVersions** objects	支持集群版本号
creationTimestamp	String	创建时间
updateTimestamp	String	更新时间

表 4-166 SupportVersions

参数	参数类型	描述
clusterType	String	支持的集群类型
clusterVersion	Array of strings	支持的集群版本（正则表达式）

请求示例

无

响应示例

状态码： 200

OK


```
{
  "kind" : "Addon",
  "apiVersion" : "v3",
  "items" : [ {
    "kind" : "Addon",
    "apiVersion" : "v3",
    "metadata" : {
      "uid" : "cie-collector",
      "name" : "cie-collector",
      "alias" : "kube-prometheus-stack",
      "creationTimestamp" : "2024-01-26T09:06:25Z",
      "updateTimestamp" : "2024-01-26T09:06:26Z"
    },
    "spec" : {
      "type" : "helm",
      "labels" : [ "CloudNativeObservability" ],
      "description" : "kube-prometheus-stack collects Kubernetes manifests, Prometheus rules combined with documentation and scripts to provide easy to operate end-to-end Kubernetes cluster monitoring with Prometheus using the Prometheus Operator.",
      "versions" : [ {
        "version" : "3.9.3",
        "input" : {
          "basic" : {
            "aom_url" : "100.79.29.98:8149",
            "region_id" : "cn-north-7",
            "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
            "swr_user" : "autopilot-official"
          },
          "parameters" : {
            "autopilot-flavor1" : {
              "category" : [ "Autopilot" ],
              "deploy_mode" : "server",
              "description" : "Recommanded when the number of containers in the cluster does not exceed 2000.",
              "name" : "Autopilot-Small(<=2000 containers)",
              "resources" : [ {
                "limitsCpu" : "500m",
                "limitsMem" : "1Gi",
                "name" : "prometheusOperator"
              }, {
                "limitsCpu" : "8",
                "limitsMem" : "32Gi",
                "name" : "prometheus"
              }, {
                "limitsCpu" : "1",
                "limitsMem" : "2Gi",
                "name" : "thanosSidecar"
              }, {
                "limitsCpu" : "4",
                "limitsMem" : "16Gi",
                "name" : "thanosQuery"
              }, {
                "limitsCpu" : "4",
                "limitsMem" : "16Gi",
                "name" : "adapter"
              }, {
                "limitsCpu" : "500m",
                "limitsMem" : "1Gi",
```



```
            "name" : "kubeStateMetrics"
          } ]
        },
        "autopilot-flavor2" : {
          "category" : [ "Autopilot" ],
          "deploy_mode" : "server",
          "description" : "Recommanded when the number of containers in the cluster does not exceed
5000.",
          "name" : "Autopilot-Medium(<=5000 containers)",
          "resources" : [ {
            "limitsCpu" : "500m",
            "limitsMem" : "1Gi",
            "name" : "prometheusOperator"
          }, {
            "limitsCpu" : "16",
            "limitsMem" : "64Gi",
            "name" : "prometheus"
          }, {
            "limitsCpu" : "2",
            "limitsMem" : "4Gi",
            "name" : "thanosSidecar"
          }, {
            "limitsCpu" : "8",
            "limitsMem" : "32Gi",
            "name" : "thanosQuery"
          }, {
            "limitsCpu" : "4",
            "limitsMem" : "32Gi",
            "name" : "adapter"
          }, {
            "limitsCpu" : "1",
            "limitsMem" : "2Gi",
            "name" : "kubeStateMetrics"
          } ]
        },
        "autopilot-flavor4" : {
          "category" : [ "Autopilot" ],
          "deploy_mode" : "server",
          "description" : "Custom configuration for this addon.",
          "name" : "custom-resources-autopilot-server",
          "resources" : [ {
            "limitsCpu" : "500m",
            "limitsMem" : "1Gi",
            "name" : "prometheusOperator"
          }, {
            "limitsCpu" : "16",
            "limitsMem" : "64Gi",
            "name" : "prometheus"
          }, {
            "limitsCpu" : "2",
            "limitsMem" : "4Gi",
            "name" : "thanosSidecar"
          }, {
            "limitsCpu" : "8",
            "limitsMem" : "32Gi",
            "name" : "thanosQuery"
          }, {
            "limitsCpu" : "4",
            "limitsMem" : "32Gi",
            "name" : "adapter"
          }, {
            "limitsCpu" : "1",
            "limitsMem" : "2Gi",
            "name" : "kubeStateMetrics"
          } ]
        },
        "autopilot-flavor5" : {
          "category" : [ "Autopilot" ],
          "deploy_mode" : "agent",
```



```
          "description" : "Recommanded flavor for agent mode when the number of containers in the
cluster does not exceed 2000.",
          "is_default" : true,
          "name" : "Autopilot-Agent-Small(<=2000 containers)",
          "resources" : [ {
            "limitsCpu" : "1",
            "limitsMem" : "1Gi",
            "name" : "prometheusOperator",
            "requestsCpu" : "1",
            "requestsMem" : "1Gi"
          }, {
            "limitsCpu" : "1800m",
            "limitsMem" : "2900Mi",
            "name" : "prometheus",
            "requestsCpu" : "1800m",
            "requestsMem" : "2900Mi"
          }, {
            "limitsCpu" : "1",
            "limitsMem" : "1Gi",
            "name" : "kubeStateMetrics",
            "requestsCpu" : "1",
            "requestsMem" : "1Gi"
          }, {
            "limitsMem" : "500Mi",
            "name" : "nodeExporter",
            "requestsMem" : "100m"
          } ]
        },
        "autopilot-flavor6" : {
          "category" : [ "Autopilot" ],
          "deploy_mode" : "agent",
          "description" : "Recommanded flavor for agent mode when the number of containers in the
cluster does not exceed 5000.",
          "name" : "Autopilot-Agent-Medium(<=5000 containers)",
          "resources" : [ {
            "limitsCpu" : "500m",
            "limitsMem" : "1Gi",
            "name" : "prometheusOperator"
          }, {
            "limitsCpu" : "4",
            "limitsMem" : "8Gi",
            "name" : "prometheus"
          }, {
            "limitsCpu" : "500m",
            "limitsMem" : "1Gi",
            "name" : "kubeStateMetrics"
          } ]
        },
        "autopilot-flavor8" : {
          "category" : [ "Autopilot" ],
          "deploy_mode" : "agent",
          "description" : "Custom flavor for agent mode",
          "name" : "custom-resources-autopilot-agent",
          "resources" : [ {
            "limitsCpu" : "500m",
            "limitsMem" : "1Gi",
            "name" : "prometheusOperator"
          }, {
            "limitsCpu" : "4",
            "limitsMem" : "8Gi",
            "name" : "prometheus"
          }, {
            "limitsCpu" : "500m",
            "limitsMem" : "1Gi",
            "name" : "kubeStateMetrics"
          } ]
        },
        "custom" : {
          "aom_app_key" : "",
```

```
        "aom_app_secret" : "",
        "aom_asm_app_key" : "",
        "aom_asm_app_secret" : "",
        "aom_asm_enable" : false,
        "aom_asm_insecure_skip_verify" : true,
        "aom_asm_instance_id" : "",
        "aom_asm_keep_regexp" : "istio.*",
        "aom_asm_project_id" : "",
        "aom_asm_remote_write_url" : "",
        "aom_auth_type" : "Bearer",
        "aom_enable" : false,
        "aom_insecure_skip_verify" : true,
        "aom_instance_id" : "",
        "aom_project_id" : "",
        "aom_region_id" : "",
        "aom_remote_read_url" : "",
        "aom_remote_write_url" : "",
        "appCode" : "",
        "appConfig" : {
          "adapter" : {
            "nodeAffinity_key" : "",
            "nodeAffinity_values" : "",
            "tolerations_effect" : "NoSchedule",
            "tolerations_key" : "",
            "tolerations_operator" : "Exists"
          },
          "alertmanager" : {
            "nodeAffinity_key" : "",
            "nodeAffinity_values" : "",
            "tolerations_effect" : "NoSchedule",
            "tolerations_key" : "",
            "tolerations_operator" : "Exists"
          },
          "kubeEventExporter" : {
            "nodeAffinity_key" : "",
            "nodeAffinity_values" : "",
            "tolerations_effect" : "NoSchedule",
            "tolerations_key" : "",
            "tolerations_operator" : "Exists"
          },
          "kubeStateMetrics" : {
            "nodeAffinity_key" : "",
            "nodeAffinity_values" : "",
            "tolerations_effect" : "NoSchedule",
            "tolerations_key" : "",
            "tolerations_operator" : "Exists"
          },
          "prometheusOperator" : {
            "nodeAffinity_key" : "",
            "nodeAffinity_values" : "",
            "tolerations_effect" : "NoSchedule",
            "tolerations_key" : "",
            "tolerations_operator" : "Exists"
          },
          "prometheusServer" : {
            "nodeAffinity_key" : "",
            "nodeAffinity_values" : "",
            "tolerations_effect" : "NoSchedule",
            "tolerations_key" : "",
            "tolerations_operator" : "Exists"
          },
          "thanosQuery" : {
            "nodeAffinity_key" : "",
            "nodeAffinity_values" : "",
            "tolerations_effect" : "NoSchedule",
            "tolerations_key" : "",
            "tolerations_operator" : "Exists"
          }
        },
```



```
    "basic_auth_password_third" : "",
    "basic_auth_username_third" : "",
    "bearer_token" : "",
    "caCert" : "",
    "certFile" : "",
    "cieInstanceId" : "",
    "cie_url" : "",
    "cluster" : "",
    "clusterId" : "",
    "cluster_category" : "CCE",
    "crd_install" : true,
    "deploy_mode" : "agent",
    "emptydir" : {
      "enabled" : false,
      "sizeLimit" : "10G"
    },
    "enableGcrypto" : true,
    "enableRemote" : false,
    "enableRemoteWrite" : false,
    "enable_autopilot" : true,
    "enable_cpd" : true,
    "enable_custom_metrics" : true,
    "enable_grafana" : true,
    "enable_nodeAffinity" : false,
    "enable_tolerations" : false,
    "enablethird" : false,
    "grafana_pvc_exist" : false,
    "highAvailability" : false,
    "insecureSkipVerify" : false,
    "insecure_skip_verify_third" : false,
    "install_with_cluster" : false,
    "keyFile" : "",
    "ksm_shards" : 1,
    "lightweight" : true,
    "lightweight_sts" : true,
    "lightweight_sts_use_pvc" : false,
    "local_hpa" : false,
    "max_wal_time" : "30m",
    "min_wal_time" : "1m",
    "nodeAffinity_key" : "",
    "nodeAffinity_values" : "",
    "projectId" : "",
    "region" : "cn-north-7",
    "retention" : "1d",
    "scrapeInterval" : "15s",
    "scrape_insecure_skip_verify" : true,
    "shards" : 1,
    "storage_class" : "csi-disk-topology",
    "storage_size" : "10Gi",
    "storage_type" : "SAS",
    "subnetId" : "",
    "supportServerModeSharding" : false,
    "tolerations_effect" : "NoSchedule",
    "tolerations_key" : "",
    "tolerations_operator" : "Exists",
    "truncate_frequency" : "30m",
    "url_third" : ""
  }
},
"scenarios" : [ {
  "category" : [ "Autopilot" ],
  "custom" : {
    "cluster_category" : "Autopilot",
    "enableGcrypto" : false,
    "enable_cpd" : false,
    "storage_class" : "csi-disk"
  },
  "name" : "autopilot-config"
} ]
```

```
      },
      "stable" : true,
      "translate" : {
        "en_US" : {
          "addon" : {
            "changeLog" : "The Autopilot cluster is supported.",
            "description" : "kube-prometheus-stack collects Kubernetes manifests, Grafana dashboards, and
Prometheus rules combined with documentation and scripts to provide easy to operate end-to-end
Kubernetes cluster monitoring with Prometheus using the Prometheus Operator. *Attention:kube-
prometheus-stack is system monitoring component, When resources are insufficient, Kubernetes
preferentially ensures pod scheduling."
          },
          "description" : {
            "Parameters.autopilot-flavor1.description" : "Recommanded when the number of containers in the
cluster does not exceed 2000.",
            "Parameters.autopilot-flavor1.name" : "Small(<=2000 containers)",
            "Parameters.autopilot-flavor2.description" : "Recommanded when the number of containers in the
cluster does not exceed 5000.",
            "Parameters.autopilot-flavor2.name" : "Medium(<=5000 containers)",
            "Parameters.autopilot-flavor3.description" : "Recommanded when the number of containers in the
cluster exceeds 5000.",
            "Parameters.autopilot-flavor3.name" : "Large(>5000 containers)",
            "Parameters.autopilot-flavor4.description" : "Custom configuration for this addon.",
            "Parameters.autopilot-flavor4.name" : "Custom",
            "Parameters.autopilot-flavor5.description" : "Recommanded when the number of containers in the
cluster does not exceed 2000.",
            "Parameters.autopilot-flavor5.name" : "Small(<=2000 containers)",
            "Parameters.autopilot-flavor6.description" : "Recommanded when the number of containers in the
cluster does not exceed 5000.",
            "Parameters.autopilot-flavor6.name" : "Medium(<=5000 containers)",
            "Parameters.autopilot-flavor7.description" : "Recommanded when the number of containers in the
cluster exceeds 5000.",
            "Parameters.autopilot-flavor7.name" : "Large(>5000 containers)",
            "Parameters.autopilot-flavor8.description" : "Custom configuration for this addon.",
            "Parameters.autopilot-flavor8.name" : "Custom",
            "Parameters.custom.deploy_mode" : "prometheus deploy mode",
            "Parameters.custom.highAvailability" : "high availability of prometheus and kube-event-exporter",
            "Parameters.custom.region" : "Availability region",
            "Parameters.custom.retention" : "Prometheus data retention period",
            "Parameters.custom.shards" : "Number of prometheus shards to distribute targets onto",
            "Parameters.custom.storage_size" : "Prometheus server data Persistent Volume size",
            "Parameters.custom.storage_type" : "Prometheus server data Persistent Volume Storage Class",
            "Parameters.custom.zone" : "Availability zone",
            "Parameters.flavor1.description" : "Just a demo for this addon. Recommanded when the number
of containers in the cluster does not exceed 100.",
            "Parameters.flavor1.name" : "Demo(<=100 containers)",
            "Parameters.flavor2.description" : "Recommanded when the number of containers in the cluster
does not exceed 2000.",
            "Parameters.flavor2.name" : "Small(<=2000 containers)",
            "Parameters.flavor3.description" : "Recommanded when the number of containers in the cluster
does not exceed 5000.",
            "Parameters.flavor3.name" : "Medium(<=5000 containers)",
            "Parameters.flavor4.description" : "Recommanded when the number of containers in the cluster
exceeds 5000.",
            "Parameters.flavor4.name" : "Large(>5000 containers)",
            "Parameters.flavor5.description" : "Custom configuration for this addon.",
            "Parameters.flavor5.name" : "custom-resources",
            "Parameters.flavor6.description" : "Default configuration for this addon.",
            "Parameters.flavor6.name" : "Default",
            "Parameters.flavor7.description" : "Custom configuration for this addon.",
            "Parameters.flavor7.name" : "Custom"
          },
          "key" : {
            "Parameters.custom.deploy_mode" : "prometheus deploy mode",
            "Parameters.custom.highAvailability" : "high availability",
            "Parameters.custom.region" : "availability region",
            "Parameters.custom.retention" : "data retention period",
            "Parameters.custom.storage_size" : "data Persistent Volume size",
            "Parameters.custom.storage_type" : "data Persistent Volume Storage Class",
```

```
          "Parameters.custom.zone" : "availability zone"
        }
      },
      "zh_CN" : {
        "addon" : {
          "changeLog" : "新增支持autopilot集群。",
          "description" : "kube-prometheus-stack通过使用Prometheus-operator和Prometheus，提供简单易用的端到端Kubernetes集群监控能力。*注意:kube-prometheus-stack为系统监控插件，当集群资源不足时，Kubernetes会优先保证插件pod的调度。"
        },
        "description" : {
          "Parameters.autopilot-flavor1.description" : "建议在集群中的容器数目不超过2000时使用。",
          "Parameters.autopilot-flavor1.name" : "小规格（2000容器以内）",
          "Parameters.autopilot-flavor2.description" : "建议在集群中的容器数目不超过5000时使用。",
          "Parameters.autopilot-flavor2.name" : "中规格（5000容器以内）",
          "Parameters.autopilot-flavor3.description" : "建议集群中容器数目超过5000时使用此规格。",
          "Parameters.autopilot-flavor3.name" : "大规格（超过5000容器）",
          "Parameters.autopilot-flavor4.description" : "自定义资源配置",
          "Parameters.autopilot-flavor4.name" : "自定义",
          "Parameters.autopilot-flavor5.description" : "建议在集群中的容器数目不超过2000时使用。",
          "Parameters.autopilot-flavor5.name" : "小规格（2000容器以内）",
          "Parameters.autopilot-flavor6.description" : "建议在集群中的容器数目不超过5000时使用。",
          "Parameters.autopilot-flavor6.name" : "中规格（5000容器以内）",
          "Parameters.autopilot-flavor7.description" : "建议集群中容器数目超过5000时使用此规格。",
          "Parameters.autopilot-flavor7.name" : "大规格（超过5000容器）",
          "Parameters.autopilot-flavor8.description" : "自定义资源配置",
          "Parameters.autopilot-flavor8.name" : "自定义",
          "Parameters.custom.deploy_mode" : "Prometheus部署模式",
          "Parameters.custom.highAvailability" : "高可用方式部署普罗米修斯服务与k8s事件采集上报服务，高可用场景下需要两个可用节点来部署双pod实例",
          "Parameters.custom.region" : "可用区域",
          "Parameters.custom.retention" : "普罗米修斯监控数据保留期",
          "Parameters.custom.shards" : "普罗米修斯服务分片数，每个分片被分配不同的采集目标",
          "Parameters.custom.storage_size" : "普罗米修斯服务器数据持久卷大小",
          "Parameters.custom.storage_type" : "普罗米修斯数据存储可用的持久卷类型",
          "Parameters.custom.zone" : "可用区",
          "Parameters.flavor1.description" : "适用于体验和功能演示环境，该规模下prometheus占用资源较少，但处理能力有限。建议在集群内容器数目不超过100时使用。",
          "Parameters.flavor1.name" : "演示规格（100容器以内）",
          "Parameters.flavor2.description" : "建议在集群中的容器数目不超过2000时使用。",
          "Parameters.flavor2.name" : "小规格（2000容器以内）",
          "Parameters.flavor3.description" : "建议在集群中的容器数目不超过5000时使用。",
          "Parameters.flavor3.name" : "中规格（5000容器以内）",
          "Parameters.flavor4.description" : "建议集群中容器数目超过5000时使用此规格。",
          "Parameters.flavor4.name" : "大规格（超过5000容器）",
          "Parameters.flavor5.description" : "自定义资源配置",
          "Parameters.flavor5.name" : "自定义",
          "Parameters.flavor6.description" : "默认资源配置",
          "Parameters.flavor6.name" : "默认",
          "Parameters.flavor7.description" : "自定义资源配置",
          "Parameters.flavor7.name" : "自定义"
        },
        "key" : {
          "Parameters.custom.deploy_mode" : "Prometheus部署模式",
          "Parameters.custom.highAvailability" : "高可用",
          "Parameters.custom.region" : "可用区域",
          "Parameters.custom.retention" : "数据保留期",
          "Parameters.custom.storage_size" : "数据持久卷大小",
          "Parameters.custom.storage_type" : "数据持久卷类型",
          "Parameters.custom.zone" : "可用区"
        }
      }
    },
    "supportVersions" : [ {
      "clusterType" : "VirtualMachine",
      "clusterVersion" : [ "v1.(27|28).*", "v1.(27|28).*" ],
      "category" : [ "Autopilot" ]
    } ],
    "creationTimestamp" : "2024-01-26T09:06:25Z",
```

```
      "updateTimestamp" : "2024-01-26T09:06:25Z"
    } ]
   }
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotAddonTemplatesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotAddonTemplatesRequest request = new ListAutopilotAddonTemplatesRequest();
        try {
            ListAutopilotAddonTemplatesResponse response = client.listAutopilotAddonTemplates(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
```



```
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotAddonTemplatesRequest()
        response = client.list_autopilot_addon_templates(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListAutopilotAddonTemplatesRequest{}
    response, err := client.ListAutopilotAddonTemplates(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
```

```
  }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.2.3 更新 AddonInstance

功能介绍

更新插件实例的功能。

调用方法

请参见**如何调用API**。

URI

PUT /autopilot/v3/addons/{id}

表 4-167 路径参数

参数	是否必选	参数类型	描述
id	是	String	插件实例id

请求参数

表 4-168 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-169 请求 Body 参数

参数	是否必选	参数类型	描述
kind	是	String	API类型，固定值“Addon”，该值不可修改，该字段传入无效。
apiVersion	是	String	API版本，固定值“v3”，该值不可修改，该字段传入无效。
metadata	是	**AddonMetadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	是	**InstanceRequestSpec** object	spec是集合类的元素类型，内容为插件实例安装/升级的具体请求信息

表 4-170 AddonMetadata

参数	是否必选	参数类型	描述
uid	否	String	唯一id标识
name	否	String	插件名称
alias	否	String	插件别名
labels	否	Map<String,String>	插件标签，key/value对格式，接口保留字段，填写不会生效
annotations	否	Map<String,String>	插件注解，由key/value组成 • 安装：固定值为{"addon.install/type":"install"} • 升级：固定值为{"addon.upgrade/type":"upgrade"}
updateTimestamp	否	String	更新时间

参数	是否必选	参数类型	描述
creationTimestamp	否	String	创建时间

表 4-171 InstanceRequestSpec

参数	是否必选	参数类型	描述
version	否	String	待安装、升级插件的版本号，例如1.0.0 • 安装：该参数非必传，如果不传，匹配集群支持的最新版本 • 升级：该参数必传，需指定版本号
clusterID	是	String	集群id
values	是	Map<String,Object>	插件模板安装参数（各插件不同），升级插件时需要填写全量安装参数，未填写参数将使用插件模板中的默认值，当前插件安装参数可通过查询插件实例接口获取。
addonTemplateName	是	String	待安装插件模板名称，如coredns

响应参数

状态码： 200

表 4-172 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Addon”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**AddonMetadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	**InstanceSpec** object	spec是集合类的元素类型，内容为插件实例具体信息，实例的详细描述主体部分都在spec中给出
status	**AddonInstanceStatus** object	插件实例状态

表 4-173 AddonMetadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	插件名称
alias	String	插件别名
labels	Map<String,String>	插件标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	插件注解，由key/value组成 ● 安装：固定值为{"addon.install/type":"install"} ● 升级：固定值为{"addon.upgrade/type":"upgrade"}
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-174 InstanceSpec

参数	参数类型	描述
clusterID	String	集群id
version	String	插件模板版本号，如1.0.0
addonTemplateName	String	插件模板名称，如coredns
addonTemplateType	String	插件模板类型
addonTemplateLogo	String	插件模板logo图片的地址
addonTemplateLabels	Array of strings	插件模板所属类型
description	String	插件模板描述
values	Map<String,Object>	插件模板安装参数（各插件不同），请根据具体插件模板信息填写安装参数。

表 4-175 AddonInstanceStatus

参数	参数类型	描述
status	String	插件实例状态, 取值如下 • running：运行中，表示插件全部实例状态都在运行中，插件正常使用。 • abnormal：不可用，表示插件状态异常，插件不可使用。可单击插件名称查看实例异常事件。 • installing：安装中，表示插件正在安装中。 • installFailed：安装失败，表示插件安装失败，需要卸载后重新安装。 • upgrading：升级中，表示插件正在更新中。 • upgradeFailed：升级失败，表示插件升级失败，可重试升级或卸载后重新安装。 • deleting：删除中，表示插件正在删除中。 • deleteFailed：删除失败，表示插件删除失败，可重试卸载。 • deleteSuccess：删除成功，表示插件删除成功。 • available：部分就绪，表示插件下只有部分实例状态为运行中，插件部分功能可用。 • rollbacking：回滚中，表示插件正在回滚中。 • rollbackFailed：回滚失败，表示插件回滚失败，可重试回滚或卸载后重新安装。 • unknown：未知状态，表示插件模板实例不存在。
Reason	String	插件安装失败原因
message	String	安装错误详情
targetVersions	Array of strings	此插件版本，支持升级的集群版本
currentVersion	**Versions** object	当前插件实例使用的具体插件版本信息
isRollbackable	Boolean	是否支持回滚到插件升级前的插件版本
previousVersion	String	插件升级或回滚前的版本

表 4-176 Versions

参数	参数类型	描述
version	String	插件版本号
input	Object	插件安装参数
stable	Boolean	是否为稳定版本
translate	Object	供界面使用的翻译信息
supportVersions	Array of **SupportVersions** objects	支持集群版本号
creationTimestamp	String	创建时间
updateTimestamp	String	更新时间

表 4-177 SupportVersions

参数	参数类型	描述
clusterType	String	支持的集群类型
clusterVersion	Array of strings	支持的集群版本（正则表达式）

请求示例

更新coredns插件，更新后的插件版本为1.28.6。

```
{
  "kind" : "Addon",
  "apiVersion" : "v3",
  "metadata" : {
    "annotations" : {
      "addon.upgrade/type" : "upgrade"
    }
  },
  "spec" : {
    "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
    "version" : "1.28.6",
    "addonTemplateName" : "coredns",
    "values" : {
      "basic" : {
        "cluster_ip" : "10.247.3.10",
        "image_version" : "1.28.6",
        "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
        "swr_user" : "autopilot-official",
        "rbac_enabled" : true,
        "cluster_version" : "v1.28"
      },
      "flavor" : {
        "category" : [ "Autopilot" ],
        "is_default" : true,
```

```
    "name" : "autopilot-flavor1",
    "replicas" : 2,
    "resources" : [ {
      "id" : "coredns",
      "name" : "coredns",
      "limitsCpu" : "2000m",
      "requestsCpu" : "2000m",
      "limitsMem" : "2048Mi",
      "requestsMem" : "2048Mi"
    } ]
  },
  "custom" : {
    "multiAZBalance" : false,
    "multiAZEnabled" : false,
    "node_match_expressions" : [ ],
    "parameterSyncStrategy" : "ensureConsistent",
    "servers" : [ {
      "plugins" : [ {
        "name" : "bind",
        "parameters" : "{$POD_IP}"
      }, {
        "configBlock" : "servfail 5s",
        "name" : "cache",
        "parameters" : 30
      }, {
        "name" : "errors"
      }, {
        "name" : "health",
        "parameters" : "{$POD_IP}:8080"
      }, {
        "name" : "ready",
        "parameters" : "{$POD_IP}:8081"
      }, {
        "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
        "name" : "kubernetes",
        "parameters" : "cluster.local in-addr.arpa ip6.arpa"
      }, {
        "name" : "loadbalance",
        "parameters" : "round_robin"
      }, {
        "name" : "prometheus",
        "parameters" : "{$POD_IP}:9153"
      }, {
        "configBlock" : "policy random",
        "name" : "forward",
        "parameters" : ". /etc/resolv.conf"
      }, {
        "name" : "reload"
      } ],
      "port" : 5353,
      "zones" : [ {
        "zone" : "."
      } ]
    } ],
    "stub_domains" : { },
    "tolerations" : [ {
      "key" : "node.kubernetes.io/not-ready",
      "operator" : "Exists",
      "effect" : "NoExecute",
      "tolerationSeconds" : 60
    }, {
      "key" : "node.kubernetes.io/unreachable",
      "operator" : "Exists",
      "effect" : "NoExecute",
      "tolerationSeconds" : 60
    } ],
    "upstream_nameservers" : [ ],
    "extraConfig" : { },
    "nodeSelector" : { }
```



```
    }
   }
  }
}
```

响应示例

状态码： 200

OK

```
{
 "kind" : "Addon",
 "apiVersion" : "v3",
 "metadata" : {
  "uid" : "90b775e0-5774-4e1d-ab3b-516332ba047a",
  "name" : "coredns",
  "alias" : "coredns",
  "creationTimestamp" : "2024-07-18T04:04:21Z",
  "updateTimestamp" : "2024-07-18T04:04:21Z"
 },
 "spec" : {
  "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
  "version" : "1.28.6",
  "addonTemplateName" : "coredns",
  "addonTemplateType" : "helm",
  "addonTemplateLogo" : "",
  "addonTemplateLabels" : [ "ContainerNetwork" ],
  "description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS Services",
  "values" : {
   "basic" : {
    "cluster_ip" : "10.247.3.10",
    "cluster_version" : "v1.28",
    "image_version" : "1.28.6",
    "rbac_enabled" : true,
    "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
    "swr_user" : "autopilot-official"
   },
   "custom" : {
    "extraConfig" : { },
    "multiAZBalance" : false,
    "multiAZEnabled" : false,
    "nodeSelector" : { },
    "node_match_expressions" : [ ],
    "parameterSyncStrategy" : "ensureConsistent",
    "servers" : [ {
     "plugins" : [ {
      "name" : "bind",
      "parameters" : "{$POD_IP}"
     }, {
      "configBlock" : "servfail 5s",
      "name" : "cache",
      "parameters" : 30
     }, {
      "name" : "errors"
     }, {
      "name" : "health",
      "parameters" : "{$POD_IP}:8080"
     }, {
      "name" : "ready",
      "parameters" : "{$POD_IP}:8081"
     }, {
      "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
      "name" : "kubernetes",
      "parameters" : "cluster.local in-addr.arpa ip6.arpa"
     }, {
      "name" : "loadbalance",
      "parameters" : "round_robin"
     }, {
```

```
          "name" : "prometheus",
          "parameters" : "{$POD_IP}:9153"
        }, {
          "configBlock" : "policy random",
          "name" : "forward",
          "parameters" : ". /etc/resolv.conf"
        }, {
          "name" : "reload"
        } ],
        "port" : 5353,
        "zones" : [ {
          "zone" : "."
        } ]
      } ],
      "stub_domains" : { },
      "tolerations" : [ {
        "effect" : "NoExecute",
        "key" : "node.kubernetes.io/not-ready",
        "operator" : "Exists",
        "tolerationSeconds" : 60
      }, {
        "effect" : "NoExecute",
        "key" : "node.kubernetes.io/unreachable",
        "operator" : "Exists",
        "tolerationSeconds" : 60
      } ],
      "upstream_nameservers" : [ ]
    },
    "flavor" : {
      "category" : [ "Autopilot" ],
      "is_default" : true,
      "name" : "autopilot-flavor1",
      "replicas" : 2,
      "resources" : [ {
        "id" : "coredns",
        "limitsCpu" : "2000m",
        "limitsMem" : "2048Mi",
        "name" : "coredns",
        "requestsCpu" : "2000m",
        "requestsMem" : "2048Mi"
      } ]
    },
    "systemAutoInject" : {
      "cluster" : {
        "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
        "clusterNetworkMode" : "eni",
        "clusterVersion" : "v1.28.5-r0"
      },
      "user" : {
        "projectID" : "47eb1d64cbeb45cfa01ae20af4f4b563"
      }
    }
  }
},
"status" : {
  "status" : "upgrading",
  "Reason" : "addon upgrading",
  "message" : "",
  "targetVersions" : null,
  "isRollbackable" : false,
  "currentVersion" : {
    "version" : "1.28.6",
    "input" : {
      "basic" : {
        "cluster_ip" : "10.247.3.10",
        "image_version" : "1.28.6",
        "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
        "swr_user" : "autopilot-official"
      },
```



```
"parameters" : {
  "autopilot-flavor1" : {
    "category" : [ "Autopilot" ],
    "is_default" : true,
    "name" : "autopilot-flavor1",
    "replicas" : 2,
    "resources" : [ {
      "limitsCpu" : 1,
      "limitsMem" : "2Gi",
      "name" : "coredns",
      "requestsCpu" : 1,
      "requestsMem" : "2Gi"
    } ]
  },
  "custom" : {
    "multiAZBalance" : false,
    "multiAZEnabled" : false,
    "node_match_expressions" : [ ],
    "parameterSyncStrategy" : "ensureConsistent",
    "servers" : [ {
      "plugins" : [ {
        "name" : "bind",
        "parameters" : "{$POD_IP}"
      }, {
        "configBlock" : "servfail 5s",
        "name" : "cache",
        "parameters" : 30
      }, {
        "name" : "errors"
      }, {
        "name" : "health",
        "parameters" : "{$POD_IP}:8080"
      }, {
        "name" : "ready",
        "parameters" : "{$POD_IP}:8081"
      }, {
        "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
        "name" : "kubernetes",
        "parameters" : "cluster.local in-addr.arpa ip6.arpa"
      }, {
        "name" : "loadbalance",
        "parameters" : "round_robin"
      }, {
        "name" : "prometheus",
        "parameters" : "{$POD_IP}:9153"
      }, {
        "configBlock" : "policy random",
        "name" : "forward",
        "parameters" : ". /etc/resolv.conf"
      }, {
        "name" : "reload"
      } ],
      "port" : 5353,
      "zones" : [ {
        "zone" : "."
      } ]
    } ],
    "stub_domains" : { },
    "tolerations" : [ {
      "effect" : "NoExecute",
      "key" : "node.kubernetes.io/not-ready",
      "operator" : "Exists",
      "tolerationSeconds" : 60
    }, {
      "effect" : "NoExecute",
      "key" : "node.kubernetes.io/unreachable",
      "operator" : "Exists",
      "tolerationSeconds" : 60
    } ],
```

```
          "upstream_nameservers" : [ ]
        },
        "flavor1" : {
          "is_default" : true,
          "name" : 2500,
          "recommend_cluster_flavor_types" : [ "small" ],
          "replicas" : 2,
          "resources" : [ {
            "limitsCpu" : "500m",
            "limitsMem" : "512Mi",
            "name" : "coredns",
            "requestsCpu" : "500m",
            "requestsMem" : "512Mi"
          } ]
        },
        "flavor2" : {
          "name" : 5000,
          "recommend_cluster_flavor_types" : [ "medium" ],
          "replicas" : 2,
          "resources" : [ {
            "limitsCpu" : "1000m",
            "limitsMem" : "1024Mi",
            "name" : "coredns",
            "requestsCpu" : "1000m",
            "requestsMem" : "1024Mi"
          } ]
        },
        "flavor3" : {
          "name" : 10000,
          "recommend_cluster_flavor_types" : [ "large" ],
          "replicas" : 2,
          "resources" : [ {
            "limitsCpu" : "2000m",
            "limitsMem" : "2048Mi",
            "name" : "coredns",
            "requestsCpu" : "2000m",
            "requestsMem" : "2048Mi"
          } ]
        },
        "flavor4" : {
          "name" : 20000,
          "recommend_cluster_flavor_types" : [ "xlarge" ],
          "replicas" : 4,
          "resources" : [ {
            "limitsCpu" : "2000m",
            "limitsMem" : "2048Mi",
            "name" : "coredns",
            "requestsCpu" : "2000m",
            "requestsMem" : "2048Mi"
          } ]
        }
      }
    },
    "stable" : true,
    "translate" : {
      "en_US" : {
        "addon" : {
          "changeLog" : "Support autopilot cluster",
          "description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS Services"
        },
        "description" : {
          "Parameters.custom.stub_domains" : "The target nameserver may itself be a Kubernetes service. For instance, you can run your own copy of dnsmasq to export custom DNS names into the ClusterDNS namespace, a JSON map using a DNS suffix key (e.g. “acme.local”) and a value consisting of a JSON array of DNS IPs.",
          "Parameters.custom.upstream_nameservers" : "If specified, then the values specified replace the nameservers taken by default from the node’s /etc/resolv.conf. Limits:a maximum of three upstream nameservers can be specified, A JSON array of DNS IPs.",
          "Parameters.flavor1.description" : "Concurrent domain name resolution ability  -  External domain
```



```
name：2500 qps，Internal domain name：10000 qps",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "Concurrent domain name resolution ability - External domain
name：5000 qps, Internal domain name：20000 qps",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "Concurrent domain name resolution ability - External domain
name：10000 qps, Internal domain name：40000 qps",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "Concurrent domain name resolution ability - External domain
name：20000 qps, Internal domain name：80000 qps",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "stub domain",
          "Parameters.custom.upstream_nameservers" : "upstream nameservers"
        }
      },
      "fr_FR" : {
        "addon" : {
          "changeLog" : "les spécifications du plugin peuvent être associées aux spécifications du cluster. le
fuseau horaire du plug-in est le même que celui du noeud",
          "description" : "Un serveur DNS qui enchaîne les plug-ins et fournit des services DNS Kubernetes."
        },
        "description" : {
          "Parameters.custom.stub_domains" : "Le serveur de noms cible peut lui-même être un service
Kubernetes. Par exemple, vous pouvez exécuter votre propre copie de dnsmasq pour exporter des noms
DNS personnalisés dans l'espace de noms ClusterDNS, une carte JSON à l'aide d'une clé de suffixe DNS (par
exemple, «acme.local») et une valeur constituée d'un tableau JSON d'adresses IP DNS.",
          "Parameters.custom.upstream_nameservers" : "Si spécifié, les valeurs spécifiées remplacent les
serveurs de noms pris par défaut dans le fichier /etc/resolv.conf du nœud. Limites: un maximum de trois
serveurs de noms en amont peuvent être spécifiés, un tableau JSON d'adresses IP DNS.",
          "Parameters.flavor1.description" : "Capacité de résolution de nom de domaine simultanée - Nom de
domaine externe: 2500 qps, Nom de domaine interne: 10000 qp",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "Capacité de résolution de nom de domaine simultanée - Nom de
domaine externe: 5000 qps, Nom de domaine interne: 20000 qp",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "Capacité de résolution de nom de domaine simultanée - Nom de
domaine externe: 10000 qps, Nom de domaine interne: 40000 qp",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "Capacité de résolution de nom de domaine simultanée - Nom de
domaine externe: 20000 qps, Nom de domaine interne: 80000 qp",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "domaine stub",
          "Parameters.custom.upstream_nameservers" : "serveurs de noms en amont"
        }
      },
      "zh_CN" : {
        "addon" : {
          "changeLog" : "支持autopilot集群",
          "description" : "CoreDNS是一款通过链式插件的方式给Kubernetes提供DNS解析服务的DNS服务器"
        },
        "description" : {
          "Parameters.custom.stub_domains" : "用户可对自定义的域名配置域名服务器，格式为一个键值对，键
为DNS后缀域名，值为一个或一组DNS IP地址，如\"acme.local -- 1.2.3.4,6.7.8.9\"。",
          "Parameters.custom.upstream_nameservers" : "解析除集群内服务域名以及自定义域名之外的域名地
址，格式为一个或一组DNS IP地址，如\"8.8.8.8\",\"8.8.4.4\"。",
          "Parameters.flavor1.description" : "并发域名解析能力 - 外部域名：2500 qps，内部域名：10000 qps",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "并发域名解析能力 - 外部域名：5000 qps，内部域名：20000 qps",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "并发域名解析能力 - 外部域名：10000 qps，内部域名：40000
qps",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "并发域名解析能力 - 外部域名：20000 qps，内部域名：80000
qps",
          "Parameters.flavor4.name" : 20000
```



```
        },
        "key" : {
          "Parameters.custom.stub_domains" : "存根域",
          "Parameters.custom.upstream_nameservers" : "上游域名服务器"
        }
      }
    },
    "supportVersions" : null,
    "creationTimestamp" : "2024-02-19T11:33:46Z",
    "updateTimestamp" : "2024-02-21T01:24:05Z"
  }
 }
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新coredns插件，更新后的插件版本为1.28.6。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;

import java.util.Map;
import java.util.HashMap;

public class UpdateAutopilotAddonInstanceSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateAutopilotAddonInstanceRequest request = new UpdateAutopilotAddonInstanceRequest();
        request.withId("{id}");
        InstanceRequest body = new InstanceRequest();
        Map<String, Object> listSpecValues = new HashMap<>();
        listSpecValues.put("basic", "{\"cluster_version\":\"v1.28\",\"rbac_enabled\":true,\"swr_user\":\"autopilot-
official\",\"image_version\":\"1.28.6\",\"cluster_ip\":\"10.247.3.10\",\"swr_addr\":\"swr.cn-
north-7.myhuaweicloud.com\"}");
        listSpecValues.put("flavor", "{\"replicas\":2,\"name\":\"autopilot-flavor1\",\"resources\":[{\"limitsCpu
\":\"2000m\",\"name\":\"coredns\",\"id\":\"coredns\",\"limitsMem\":\"2048Mi\",\"requestsMem\":\"2048Mi
\",\"requestsCpu\":\"2000m\"}],\"category\":[\"Autopilot\"],\"is_default\":true}");
        listSpecValues.put("custom", "{\"extraConfig\":{},\"servers\":[{\"port\":5353,\"plugins\":[{\"name
\":\"bind\",\"parameters\":\"{$POD_IP}\"},{\"configBlock\":\"servfail 5s\",\"name\":\"cache\",\"parameters
```



```
\":30},{\"name\":\"errors\"},{\"name\":\"health\",\"parameters\":\"{$POD_IP}:8080\"},{\"name\":\"ready
\",\"parameters\":\"{$POD_IP}:8081\"},{\"configBlock\":\"pods insecure\nfallthrough in-addr.arpa ip6.arpa
\",\"name\":\"kubernetes\",\"parameters\":\"cluster.local in-addr.arpa ip6.arpa\"},{\"name\":\"loadbalance
\",\"parameters\":\"round_robin\"},{\"name\":\"prometheus\",\"parameters\":\"{$POD_IP}:9153\"},
{\"configBlock\":\"policy random\",\"name\":\"forward\",\"parameters\":\". /etc/resolv.conf\"},{\"name
\":\"reload\"}],\"zones\":[{\"zone\":\".\"}]}],\"tolerations\":[{\"effect\":\"NoExecute\",\"tolerationSeconds
\":60,\"key\":\"node.kubernetes.io/not-ready\",\"operator\":\"Exists\"},{\"effect\":\"NoExecute
\",\"tolerationSeconds\":60,\"key\":\"node.kubernetes.io/unreachable\",\"operator\":\"Exists
\"}],\"multiAZBalance\":false,\"node_match_expressions\":[],\"stub_domains\":{},\"multiAZEnabled
\":false,\"parameterSyncStrategy\":\"ensureConsistent\",\"upstream_nameservers\":[],\"nodeSelector\":{}}");
        InstanceRequestSpec specbody = new InstanceRequestSpec();
        specbody.withVersion("1.28.6")
            .withClusterID("597f2d95-44ab-11ef-9e39-0255ac100115")
            .withValues(listSpecValues)
            .withAddonTemplateName("coredns");
        Map<String, String> listMetadataAnnotations = new HashMap<>();
        listMetadataAnnotations.put("addon.upgrade/type", "upgrade");
        AddonMetadata metadatabody = new AddonMetadata();
        metadatabody.withAnnotations(listMetadataAnnotations);
        body.withSpec(specbody);
        body.withMetadata(metadatabody);
        body.withApiVersion("v3");
        body.withKind("Addon");
        request.withBody(body);
        try {
            UpdateAutopilotAddonInstanceResponse response = client.updateAutopilotAddonInstance(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

更新coredns插件，更新后的插件版本为1.28.6。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()
```

```
    try:
        request = UpdateAutopilotAddonInstanceRequest()
        request.id = "{id}"
        listValuesSpec = {
            "basic": "{\"cluster_version\":\"v1.28\",\"rbac_enabled\":true,\"swr_user\":\"autopilot-official
\",\"image_version\":\"1.28.6\",\"cluster_ip\":\"10.247.3.10\",\"swr_addr\":\"swr.cn-
north-7.myhuaweicloud.com\"}",
            "flavor": "{\"replicas\":2,\"name\":\"autopilot-flavor1\",\"resources\":[{\"limitsCpu\":\"2000m
\",\"name\":\"coredns\",\"id\":\"coredns\",\"limitsMem\":\"2048Mi\",\"requestsMem\":\"2048Mi
\",\"requestsCpu\":\"2000m\"}],\"category\":[\"Autopilot\"],\"is_default\":true}",
            "custom": "{\"extraConfig\":{},\"servers\":[{\"port\":5353,\"plugins\":[{\"name\":\"bind
\",\"parameters\":\"{$POD_IP}\"},{\"configBlock\":\"servfail 5s\",\"name\":\"cache\",\"parameters\":30},
{\"name\":\"errors\"},{\"name\":\"health\",\"parameters\":\"{$POD_IP}:8080\"},{\"name\":\"ready
\",\"parameters\":\"{$POD_IP}:8081\"},{\"configBlock\":\"pods insecure\nfallthrough in-addr.arpa ip6.arpa
\",\"name\":\"kubernetes\",\"parameters\":\"cluster.local in-addr.arpa ip6.arpa\"},{\"name\":\"loadbalance
\",\"parameters\":\"round_robin\"},{\"name\":\"prometheus\",\"parameters\":\"{$POD_IP}:9153\"},
{\"configBlock\":\"policy random\",\"name\":\"forward\",\"parameters\":\". /etc/resolv.conf\"},{\"name
\":\"reload\"}],\"zones\":[{\"zone\":\".\"}]}],\"tolerations\":[{\"effect\":\"NoExecute\",\"tolerationSeconds
\":60,\"key\":\"node.kubernetes.io/not-ready\",\"operator\":\"Exists\"},{\"effect\":\"NoExecute
\",\"tolerationSeconds\":60,\"key\":\"node.kubernetes.io/unreachable\",\"operator\":\"Exists
\"}],\"multiAZBalance\":false,\"node_match_expressions\":[],\"stub_domains\":{},\"multiAZEnabled
\":false,\"parameterSyncStrategy\":\"ensureConsistent\",\"upstream_nameservers\":[],\"nodeSelector\":{}}"
        }
        specbody = InstanceRequestSpec(
            version="1.28.6",
            cluster_id="597f2d95-44ab-11ef-9e39-0255ac100115",
            values=listValuesSpec,
            addon_template_name="coredns"
        )
        listAnnotationsMetadata = {
            "addon.upgrade/type": "upgrade"
        }
        metadatabody = AddonMetadata(
            annotations=listAnnotationsMetadata
        )
        request.body = InstanceRequest(
            spec=specbody,
            metadata=metadatabody,
            api_version="v3",
            kind="Addon"
        )
        response = client.update_autopilot_addon_instance(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新coredns插件，更新后的插件版本为1.28.6。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
```

```
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateAutopilotAddonInstanceRequest{}
    request.Id = "{id}"
    var listValuesSpec = map[string]interface{}{
        "basic": "{\"cluster_version\":\"v1.28\",\"rbac_enabled\":true,\"swr_user\":\"autopilot-official
\",\"image_version\":\"1.28.6\",\"cluster_ip\":\"10.247.3.10\",\"swr_addr\":\"swr.cn-
north-7.myhuaweicloud.com\"}",
        "flavor": "{\"replicas\":2,\"name\":\"autopilot-flavor1\",\"resources\":[{\"limitsCpu\":\"2000m\",\"name
\":\"coredns\",\"id\":\"coredns\",\"limitsMem\":\"2048Mi\",\"requestsMem\":\"2048Mi\",\"requestsCpu
\":\"2000m\"}],\"category\":[\"Autopilot\"],\"is_default\":true}",
        "custom": "{\"extraConfig\":{},\"servers\":[{\"port\":5353,\"plugins\":[{\"name\":\"bind\",\"parameters
\":\"{$POD_IP}\"},{\"configBlock\":\"servfail 5s\",\"name\":\"cache\",\"parameters\":30},{\"name\":\"errors\"},
{\"name\":\"health\",\"parameters\":\"{$POD_IP}:8080\"},{\"name\":\"ready\",\"parameters
\":\"{$POD_IP}:8081\"},{\"configBlock\":\"pods insecure\nfallthrough in-addr.arpa ip6.arpa\",\"name
\":\"kubernetes\",\"parameters\":\"cluster.local in-addr.arpa ip6.arpa\"},{\"name\":\"loadbalance
\",\"parameters\":\"round_robin\"},{\"name\":\"prometheus\",\"parameters\":\"{$POD_IP}:9153\"},
{\"configBlock\":\"policy random\",\"name\":\"forward\",\"parameters\":\". /etc/resolv.conf\"},{\"name
\":\"reload\"}],\"zones\":[{\"zone\":\".\"}]}],\"tolerations\":[{\"effect\":\"NoExecute\",\"tolerationSeconds
\":60,\"key\":\"node.kubernetes.io/not-ready\",\"operator\":\"Exists\"},{\"effect\":\"NoExecute
\",\"tolerationSeconds\":60,\"key\":\"node.kubernetes.io/unreachable\",\"operator\":\"Exists
\"}],\"multiAZBalance\":false,\"node_match_expressions\":[],\"stub_domains\":{},\"multiAZEnabled
\":false,\"parameterSyncStrategy\":\"ensureConsistent\",\"upstream_nameservers\":[],\"nodeSelector\":{}}",
    }
    versionSpec:= "1.28.6"
    specbody := &model.InstanceRequestSpec{
        Version: &versionSpec,
        ClusterID: "597f2d95-44ab-11ef-9e39-0255ac100115",
        Values: listValuesSpec,
        AddonTemplateName: "coredns",
    }
    var listAnnotationsMetadata = map[string]string{
        "addon.upgrade/type": "upgrade",
    }
    metadatabody := &model.AddonMetadata{
        Annotations: listAnnotationsMetadata,
    }
    request.Body = &model.InstanceRequest{
        Spec: specbody,
        Metadata: metadatabody,
        ApiVersion: "v3",
        Kind: "Addon",
    }
    response, err := client.UpdateAutopilotAddonInstance(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.2.4 回滚 AddonInstance

功能介绍

将插件实例回滚到升级前的版本。只有在当前插件实例版本支持回滚到升级前的版本（status.isRollbackable为true），且插件实例状态为running（运行中）、available（可用）、abnormal（不可用）、upgradeFailed（升级失败）、rollbackFailed（回滚失败）时支持回滚。

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/addons/{id}/operation/rollback

表 4-178 路径参数

参数	是否必选	参数类型	描述
id	是	String	插件实例ID

请求参数

表 4-179 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-180 请求 Body 参数

参数	是否必选	参数类型	描述
clusterID	是	String	集群ID

响应参数

状态码： 200

表 4-181 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Addon”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**AddonMetadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	**InstanceSpec** object	spec是集合类的元素类型，内容为插件实例具体信息，实例的详细描述主体部分都在spec中给出
status	**AddonInstanceStatus** object	插件实例状态

表 4-182 AddonMetadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	插件名称
alias	String	插件别名
labels	Map<String,String>	插件标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	插件注解，由key/value组成 • 安装：固定值为{"addon.install/type":"install"} • 升级：固定值为{"addon.upgrade/type":"upgrade"}
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-183 InstanceSpec

参数	参数类型	描述
clusterID	String	集群id
version	String	插件模板版本号，如1.0.0
addonTemplateName	String	插件模板名称，如coredns
addonTemplateType	String	插件模板类型
addonTemplateLogo	String	插件模板logo图片的地址
addonTemplateLabels	Array of strings	插件模板所属类型
description	String	插件模板描述
values	Map<String,Object>	插件模板安装参数（各插件不同），请根据具体插件模板信息填写安装参数。

表 4-184 AddonInstanceStatus

参数	参数类型	描述
status	String	插件实例状态, 取值如下 • running：运行中，表示插件全部实例状态都在运行中，插件正常使用。 • abnormal：不可用，表示插件状态异常，插件不可使用。可单击插件名称查看实例异常事件。 • installing：安装中，表示插件正在安装中。 • installFailed：安装失败，表示插件安装失败，需要卸载后重新安装。 • upgrading：升级中，表示插件正在更新中。 • upgradeFailed：升级失败，表示插件升级失败，可重试升级或卸载后重新安装。 • deleting：删除中，表示插件正在删除中。 • deleteFailed：删除失败，表示插件删除失败，可重试卸载。 • deleteSuccess：删除成功，表示插件删除成功。 • available：部分就绪，表示插件下只有部分实例状态为运行中，插件部分功能可用。 • rollbacking：回滚中，表示插件正在回滚中。 • rollbackFailed：回滚失败，表示插件回滚失败，可重试回滚或卸载后重新安装。 • unknown：未知状态，表示插件模板实例不存在。
Reason	String	插件安装失败原因
message	String	安装错误详情
targetVersions	Array of strings	此插件版本，支持升级的集群版本
currentVersion	**Versions** object	当前插件实例使用的具体插件版本信息
isRollbackable	Boolean	是否支持回滚到插件升级前的插件版本
previousVersion	String	插件升级或回滚前的版本

表 4-185 Versions

参数	参数类型	描述
version	String	插件版本号
input	Object	插件安装参数
stable	Boolean	是否为稳定版本
translate	Object	供界面使用的翻译信息
supportVersions	Array of **SupportVersions** objects	支持集群版本号
creationTimestamp	String	创建时间
updateTimestamp	String	更新时间

表 4-186 SupportVersions

参数	参数类型	描述
clusterType	String	支持的集群类型
clusterVersion	Array of strings	支持的集群版本（正则表达式）

请求示例

```
{
  "clusterID" : "******"
}
```

响应示例

状态码： 200

插件实例回滚成功

```
{
  "kind" : "Addon",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "4eba2678-330f-430b-aefc-821e1bbeff34",
    "name" : "coredns",
    "alias" : "coredns",
    "creationTimestamp" : "2024-07-18T03:03:17Z",
    "updateTimestamp" : "2024-07-18T03:04:44Z"
  },
  "spec" : {
    "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
    "version" : "1.28.4",
    "addonTemplateName" : "coredns",
    "addonTemplateType" : "helm",
```

```
"addonTemplateLogo" : "",
"addonTemplateLabels" : [ "ContainerNetwork" ],
"description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS Services",
"values" : {
  "basic" : {
    "cluster_ip" : "10.247.3.10",
    "cluster_version" : "v1.28",
    "image_version" : "1.28.4",
    "platform" : "linux-amd64",
    "rbac_enabled" : true,
    "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
    "swr_user" : "autopilot-official"
  },
  "custom" : {
    "extraConfig" : { },
    "multiAZBalance" : false,
    "multiAZEnabled" : false,
    "nodeSelector" : { },
    "node_match_expressions" : [ ],
    "parameterSyncStrategy" : "ensureConsistent",
    "servers" : [ {
      "plugins" : [ {
        "name" : "bind",
        "parameters" : "{$POD_IP}"
      }, {
        "configBlock" : "servfail 5s",
        "name" : "cache",
        "parameters" : 30
      }, {
        "name" : "errors"
      }, {
        "name" : "health",
        "parameters" : "{$POD_IP}:8080"
      }, {
        "name" : "ready",
        "parameters" : "{$POD_IP}:8081"
      }, {
        "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
        "name" : "kubernetes",
        "parameters" : "cluster.local in-addr.arpa ip6.arpa"
      }, {
        "name" : "loadbalance",
        "parameters" : "round_robin"
      }, {
        "name" : "prometheus",
        "parameters" : "{$POD_IP}:9153"
      }, {
        "configBlock" : "policy random",
        "name" : "forward",
        "parameters" : ". /etc/resolv.conf"
      }, {
        "name" : "reload"
      } ],
      "port" : 5353,
      "zones" : [ {
        "zone" : "."
      } ]
    } ],
    "stub_domains" : { },
    "tolerations" : [ {
      "effect" : "NoExecute",
      "key" : "node.kubernetes.io/not-ready",
      "operator" : "Exists",
      "tolerationSeconds" : 60
    }, {
      "effect" : "NoExecute",
      "key" : "node.kubernetes.io/unreachable",
      "operator" : "Exists",
      "tolerationSeconds" : 60
```

```
    } ],
    "upstream_nameservers" : [ ]
  },
  "flavor" : {
    "category" : [ "Autopilot" ],
    "is_default" : true,
    "name" : "autopilot-flavor1",
    "replicas" : 2,
    "resources" : [ {
      "id" : "coredns",
      "limitsCpu" : "1000m",
      "limitsMem" : "1024Mi",
      "name" : "coredns",
      "requestsCpu" : "1000m",
      "requestsMem" : "1024Mi"
    } ]
  },
  "image" : {
    "pullPolicy" : "Always"
  },
  "isClusterService" : true,
  "multiAZPreferred" : {
    "podAntiAffinity" : {
      "preferredDuringSchedulingIgnoredDuringExecution" : [ {
        "podAffinityTerm" : {
          "labelSelector" : {
            "matchExpressions" : [ {
              "key" : "app",
              "operator" : "In",
              "values" : [ "coredns" ]
            } ]
          },
          "topologyKey" : "topology.kubernetes.io/zone"
        },
        "weight" : 100
      } ]
    }
  },
  "multiAZRequired" : {
    "podAntiAffinity" : {
      "requiredDuringSchedulingIgnoredDuringExecution" : [ {
        "labelSelector" : {
          "matchExpressions" : [ {
            "key" : "az-antiaffinity-app",
            "operator" : "In",
            "values" : [ "coredns" ]
          } ]
        },
        "topologyKey" : "topology.kubernetes.io/zone"
      } ]
    }
  },
  "nodeSelector" : { },
  "rbac" : {
    "create" : true,
    "serviceAccountName" : "default"
  },
  "service" : {
    "annotations" : {
      "prometheus.io/port" : "9153",
      "prometheus.io/scrape" : "true"
    },
    "clusterIP" : "10.247.3.10",
    "type" : "ClusterIP"
  },
  "systemAutoInject" : {
    "cluster" : {
      "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
      "clusterNetworkMode" : "eni",
```



```
        "clusterVersion" : "v1.28.5-r0"
      },
      "user" : {
        "projectID" : "47eb1d64cbeb45cfa01ae20af4f4b563"
      }
    },
    "topologySpreadConstraints" : [ {
      "labelSelector" : {
        "matchLabels" : {
          "app" : "coredns"
        }
      },
      "maxSkew" : 1,
      "topologyKey" : "topology.kubernetes.io/zone",
      "whenUnsatisfiable" : "DoNotSchedule"
    } ],
    "zoneFiles" : [ ]
  }
},
"status" : {
  "status" : "rollbacking",
  "Reason" : "Rollback to 1",
  "message" : "",
  "targetVersions" : [ "1.28.6" ],
  "isRollbackable" : false,
  "previousVersion" : "1.28.6",
  "currentVersion" : {
    "version" : "1.28.4",
    "input" : {
      "basic" : {
        "cluster_ip" : "10.247.3.10",
        "image_version" : "1.28.4",
        "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
        "swr_user" : "autopilot-official"
      },
      "parameters" : {
        "autopilot-flavor1" : {
          "category" : [ "Autopilot" ],
          "is_default" : true,
          "name" : "autopilot-flavor1",
          "replicas" : 2,
          "resources" : [ {
            "limitsCpu" : 1,
            "limitsMem" : "1Gi",
            "name" : "coredns",
            "requestsCpu" : 1,
            "requestsMem" : "1Gi"
          } ]
        },
        "custom" : {
          "multiAZBalance" : false,
          "multiAZEnabled" : false,
          "node_match_expressions" : [ ],
          "parameterSyncStrategy" : "ensureConsistent",
          "servers" : [ {
            "plugins" : [ {
              "name" : "bind",
              "parameters" : "{$POD_IP}"
            }, {
              "configBlock" : "servfail 5s",
              "name" : "cache",
              "parameters" : 30
            }, {
              "name" : "errors"
            }, {
              "name" : "health",
              "parameters" : "{$POD_IP}:8080"
            }, {
              "name" : "ready",
```



```
          "parameters" : "{$POD_IP}:8081"
        }, {
          "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
          "name" : "kubernetes",
          "parameters" : "cluster.local in-addr.arpa ip6.arpa"
        }, {
          "name" : "loadbalance",
          "parameters" : "round_robin"
        }, {
          "name" : "prometheus",
          "parameters" : "{$POD_IP}:9153"
        }, {
          "configBlock" : "policy random",
          "name" : "forward",
          "parameters" : ". /etc/resolv.conf"
        }, {
          "name" : "reload"
        } ],
        "port" : 5353,
        "zones" : [ {
          "zone" : "."
        } ]
      } ],
      "stub_domains" : { },
      "tolerations" : [ {
        "effect" : "NoExecute",
        "key" : "node.kubernetes.io/not-ready",
        "operator" : "Exists",
        "tolerationSeconds" : 60
      }, {
        "effect" : "NoExecute",
        "key" : "node.kubernetes.io/unreachable",
        "operator" : "Exists",
        "tolerationSeconds" : 60
      } ],
      "upstream_nameservers" : [ ]
    },
    "flavor1" : {
      "is_default" : true,
      "name" : 2500,
      "recommend_cluster_flavor_types" : [ "small" ],
      "replicas" : 2,
      "resources" : [ {
        "limitsCpu" : "500m",
        "limitsMem" : "512Mi",
        "name" : "coredns",
        "requestsCpu" : "500m",
        "requestsMem" : "512Mi"
      } ]
    },
    "flavor2" : {
      "name" : 5000,
      "recommend_cluster_flavor_types" : [ "medium" ],
      "replicas" : 2,
      "resources" : [ {
        "limitsCpu" : "1000m",
        "limitsMem" : "1024Mi",
        "name" : "coredns",
        "requestsCpu" : "1000m",
        "requestsMem" : "1024Mi"
      } ]
    },
    "flavor3" : {
      "name" : 10000,
      "recommend_cluster_flavor_types" : [ "large" ],
      "replicas" : 2,
      "resources" : [ {
        "limitsCpu" : "2000m",
        "limitsMem" : "2048Mi",
```

```
            "name" : "coredns",
            "requestsCpu" : "2000m",
            "requestsMem" : "2048Mi"
          } ]
        },
        "flavor4" : {
          "name" : 20000,
          "recommend_cluster_flavor_types" : [ "xlarge" ],
          "replicas" : 4,
          "resources" : [ {
            "limitsCpu" : "2000m",
            "limitsMem" : "2048Mi",
            "name" : "coredns",
            "requestsCpu" : "2000m",
            "requestsMem" : "2048Mi"
          } ]
        }
      }
    },
    "stable" : true,
    "translate" : {
      "en_US" : {
        "addon" : {
          "changeLog" : "plugin specifications can be associated with cluster specifications. The time zone of the plug-in is the same as that of the node",
          "description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS Services"
        },
        "description" : {
          "Parameters.custom.stub_domains" : "The target nameserver may itself be a Kubernetes service. For instance, you can run your own copy of dnsmasq to export custom DNS names into the ClusterDNS namespace, a JSON map using a DNS suffix key (e.g. “acme.local”) and a value consisting of a JSON array of DNS IPs.",
          "Parameters.custom.upstream_nameservers" : "If specified, then the values specified replace the nameservers taken by default from the node’s /etc/resolv.conf. Limits:a maximum of three upstream nameservers can be specified, A JSON array of DNS IPs.",
          "Parameters.flavor1.description" : "Concurrent domain name resolution ability - External domain name：2500 qps, Internal domain name：10000 qps",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "Concurrent domain name resolution ability - External domain name：5000 qps, Internal domain name：20000 qps",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "Concurrent domain name resolution ability - External domain name：10000 qps, Internal domain name：40000 qps",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "Concurrent domain name resolution ability - External domain name：20000 qps, Internal domain name：80000 qps",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "stub domain",
          "Parameters.custom.upstream_nameservers" : "upstream nameservers"
        }
      },
      "fr_FR" : {
        "addon" : {
          "changeLog" : "les spécifications du plugin peuvent être associées aux spécifications du cluster. le fuseau horaire du plug-in est le même que celui du noeud",
          "description" : "Un serveur DNS qui enchaîne les plug-ins et fournit des services DNS Kubernetes."
        },
        "description" : {
          "Parameters.custom.stub_domains" : "Le serveur de noms cible peut lui-même être un service Kubernetes. Par exemple, vous pouvez exécuter votre propre copie de dnsmasq pour exporter des noms DNS personnalisés dans l'espace de noms ClusterDNS, une carte JSON à l'aide d'une clé de suffixe DNS (par exemple, «acme.local») et une valeur constituée d'un tableau JSON d'adresses IP DNS.",
          "Parameters.custom.upstream_nameservers" : "Si spécifié, les valeurs spécifiées remplacent les serveurs de noms pris par défaut dans le fichier /etc/resolv.conf du nœud. Limites: un maximum de trois serveurs de noms en amont peuvent être spécifiés, un tableau JSON d'adresses IP DNS.",
          "Parameters.flavor1.description" : "Capacité de résolution de nom de domaine simultanée - Nom de domaine externe: 2500 qps, Nom de domaine interne: 10000 qp",
```



```
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "Capacité de résolution de nom de domaine simultanée - Nom de domaine externe: 5000 qps, Nom de domaine interne: 20000 qp",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "Capacité de résolution de nom de domaine simultanée - Nom de domaine externe: 10000 qps, Nom de domaine interne: 40000 qp",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "Capacité de résolution de nom de domaine simultanée - Nom de domaine externe: 20000 qps, Nom de domaine interne: 80000 qp",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "domaine stub",
          "Parameters.custom.upstream_nameservers" : "serveurs de noms en amont"
        }
      },
      "zh_CN" : {
        "addon" : {
          "changeLog" : "支持插件规格与集群规格联动; 插件与节点时区一致",
          "description" : "CoreDNS是一款通过链式插件的方式给Kubernetes提供DNS解析服务的DNS服务器"
        },
        "description" : {
          "Parameters.custom.stub_domains" : "用户可对自定义的域名配置域名服务器，格式为一个键值对，键为DNS后缀域名，值为一个或一组DNS IP地址，如\"acme.local -- 1.2.3.4,6.7.8.9\"。",
          "Parameters.custom.upstream_nameservers" : "解析除集群内服务域名以及自定义域名之外的域名地址，格式为一个或一组DNS IP地址，如\"8.8.8.8\",\"8.8.4.4\"。",
          "Parameters.flavor1.description" : "并发域名解析能力 - 外部域名：2500 qps，内部域名：10000 qps",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "并发域名解析能力 - 外部域名：5000 qps，内部域名：20000 qps",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "并发域名解析能力 - 外部域名：10000 qps，内部域名：40000 qps",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "并发域名解析能力 - 外部域名：20000 qps，内部域名：80000 qps",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "存根域",
          "Parameters.custom.upstream_nameservers" : "上游域名服务器"
        }
      }
    },
    "supportVersions" : null,
    "creationTimestamp" : "2024-01-22T11:05:45Z",
    "updateTimestamp" : "2024-01-22T11:05:45Z"
  }
 }
}
```


状态码

状态码	描述
200	插件实例回滚成功

错误码

请参见**错误码**。

4.2.5 删除 AddonInstance

功能介绍

删除插件实例的功能。

调用方法

请参见**如何调用API**。

URI

DELETE /autopilot/v3/addons/{id}

表 4-187 路径参数

参数	是否必选	参数类型	描述
id	是	String	插件实例id

表 4-188 Query 参数

参数	是否必选	参数类型	描述
cluster_id	否	String	集群 ID，获取方式请参见**如何获取接口URI中参数**

请求参数

表 4-189 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-190 响应 Body 参数

参数	参数类型	描述
-	String	

请求示例

无

响应示例

状态码： 200

OK

```
success
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class DeleteAutopilotAddonInstanceSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteAutopilotAddonInstanceRequest request = new DeleteAutopilotAddonInstanceRequest();
        request.withId("{id}");
        try {
            DeleteAutopilotAddonInstanceResponse response = client.deleteAutopilotAddonInstance(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
```

```
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteAutopilotAddonInstanceRequest()
        request.id = "{id}"
        response = client.delete_autopilot_addon_instance(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
```

```
    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteAutopilotAddonInstanceRequest{}
    request.Id = "{id}"
    response, err := client.DeleteAutopilotAddonInstance(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.2.6 获取 AddonInstance 详情

功能介绍

获取插件实例详情。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/addons/{id}

表 4-191 路径参数

参数	是否必选	参数类型	描述
id	是	String	插件实例id

表 4-192 Query 参数

参数	是否必选	参数类型	描述
cluster_id	否	String	集群 ID，获取方式请参见**如何获取接口URI中参数**

请求参数

表 4-193 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-194 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Addon”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**AddonMetadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	**InstanceSpec** object	spec是集合类的元素类型，内容为插件实例具体信息，实例的详细描述主体部分都在spec中给出
status	**AddonInstanceStatus** object	插件实例状态

表 4-195 AddonMetadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	插件名称
alias	String	插件别名
labels	Map<String,String>	插件标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	插件注解，由key/value组成 ● 安装：固定值为{"addon.install/type":"install"} ● 升级：固定值为{"addon.upgrade/type":"upgrade"}
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-196 InstanceSpec

参数	参数类型	描述
clusterID	String	集群id
version	String	插件模板版本号，如1.0.0
addonTemplateName	String	插件模板名称，如coredns
addonTemplateType	String	插件模板类型
addonTemplateLogo	String	插件模板logo图片的地址
addonTemplateLabels	Array of strings	插件模板所属类型
description	String	插件模板描述
values	Map<String,Object>	插件模板安装参数（各插件不同），请根据具体插件模板信息填写安装参数。

表 4-197 AddonInstanceStatus

参数	参数类型	描述
status	String	插件实例状态, 取值如下 • running：运行中，表示插件全部实例状态都在运行中，插件正常使用。 • abnormal：不可用，表示插件状态异常，插件不可使用。可单击插件名称查看实例异常事件。 • installing：安装中，表示插件正在安装中。 • installFailed：安装失败，表示插件安装失败，需要卸载后重新安装。 • upgrading：升级中，表示插件正在更新中。 • upgradeFailed：升级失败，表示插件升级失败，可重试升级或卸载后重新安装。 • deleting：删除中，表示插件正在删除中。 • deleteFailed：删除失败，表示插件删除失败，可重试卸载。 • deleteSuccess：删除成功，表示插件删除成功。 • available：部分就绪，表示插件下只有部分实例状态为运行中，插件部分功能可用。 • rollbacking：回滚中，表示插件正在回滚中。 • rollbackFailed：回滚失败，表示插件回滚失败，可重试回滚或卸载后重新安装。 • unknown：未知状态，表示插件模板实例不存在。
Reason	String	插件安装失败原因
message	String	安装错误详情
targetVersions	Array of strings	此插件版本，支持升级的集群版本
currentVersion	**Versions** object	当前插件实例使用的具体插件版本信息
isRollbackable	Boolean	是否支持回滚到插件升级前的插件版本
previousVersion	String	插件升级或回滚前的版本

表 4-198 Versions

参数	参数类型	描述
version	String	插件版本号
input	Object	插件安装参数
stable	Boolean	是否为稳定版本
translate	Object	供界面使用的翻译信息
supportVersions	Array of **SupportVersions** objects	支持集群版本号
creationTimestamp	String	创建时间
updateTimestamp	String	更新时间

表 4-199 SupportVersions

参数	参数类型	描述
clusterType	String	支持的集群类型
clusterVersion	Array of strings	支持的集群版本（正则表达式）

请求示例

无

响应示例

状态码： 200

OK


```
{
  "kind" : "Addon",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "90b775e0-5774-4e1d-ab3b-516332ba047a",
    "name" : "coredns",
    "alias" : "coredns",
    "creationTimestamp" : "2024-07-18T04:04:21Z",
    "updateTimestamp" : "2024-07-18T04:04:21Z"
  },
  "spec" : {
    "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
    "version" : "1.28.6",
    "addonTemplateName" : "coredns",
    "addonTemplateType" : "helm",
    "addonTemplateLogo" : "",
```

```
  "addonTemplateLabels" : [ "ContainerNetwork" ],
  "description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS Services",
  "values" : null
},
"status" : {
  "status" : "abnormal",
  "Reason" : "",
  "message" : "",
  "targetVersions" : null,
  "isRollbackable" : false,
  "currentVersion" : {
    "version" : "1.28.6",
    "input" : {
      "basic" : {
        "cluster_ip" : "10.247.3.10",
        "image_version" : "1.28.6",
        "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
        "swr_user" : "autopilot-official"
      },
      "parameters" : {
        "autopilot-flavor1" : {
          "category" : [ "Autopilot" ],
          "is_default" : true,
          "name" : "autopilot-flavor1",
          "replicas" : 2,
          "resources" : [ {
            "limitsCpu" : 1,
            "limitsMem" : "2Gi",
            "name" : "coredns",
            "requestsCpu" : 1,
            "requestsMem" : "2Gi"
          } ]
        },
        "custom" : {
          "multiAZBalance" : false,
          "multiAZEnabled" : false,
          "node_match_expressions" : [ ],
          "parameterSyncStrategy" : "ensureConsistent",
          "servers" : [ {
            "plugins" : [ {
              "name" : "bind",
              "parameters" : "{$POD_IP}"
            }, {
              "configBlock" : "servfail 5s",
              "name" : "cache",
              "parameters" : 30
            }, {
              "name" : "errors"
            }, {
              "name" : "health",
              "parameters" : "{$POD_IP}:8080"
            }, {
              "name" : "ready",
              "parameters" : "{$POD_IP}:8081"
            }, {
              "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
              "name" : "kubernetes",
              "parameters" : "cluster.local in-addr.arpa ip6.arpa"
            }, {
              "name" : "loadbalance",
              "parameters" : "round_robin"
            }, {
              "name" : "prometheus",
              "parameters" : "{$POD_IP}:9153"
            }, {
              "configBlock" : "policy random",
              "name" : "forward",
              "parameters" : ". /etc/resolv.conf"
            }, {
```



```
        "name" : "reload"
      } ],
      "port" : 5353,
      "zones" : [ {
        "zone" : "."
      } ]
    } ],
    "stub_domains" : { },
    "tolerations" : [ {
      "effect" : "NoExecute",
      "key" : "node.kubernetes.io/not-ready",
      "operator" : "Exists",
      "tolerationSeconds" : 60
    }, {
      "effect" : "NoExecute",
      "key" : "node.kubernetes.io/unreachable",
      "operator" : "Exists",
      "tolerationSeconds" : 60
    } ],
    "upstream_nameservers" : [ ]
  },
  "flavor1" : {
    "is_default" : true,
    "name" : 2500,
    "recommend_cluster_flavor_types" : [ "small" ],
    "replicas" : 2,
    "resources" : [ {
      "limitsCpu" : "500m",
      "limitsMem" : "512Mi",
      "name" : "coredns",
      "requestsCpu" : "500m",
      "requestsMem" : "512Mi"
    } ]
  },
  "flavor2" : {
    "name" : 5000,
    "recommend_cluster_flavor_types" : [ "medium" ],
    "replicas" : 2,
    "resources" : [ {
      "limitsCpu" : "1000m",
      "limitsMem" : "1024Mi",
      "name" : "coredns",
      "requestsCpu" : "1000m",
      "requestsMem" : "1024Mi"
    } ]
  },
  "flavor3" : {
    "name" : 10000,
    "recommend_cluster_flavor_types" : [ "large" ],
    "replicas" : 2,
    "resources" : [ {
      "limitsCpu" : "2000m",
      "limitsMem" : "2048Mi",
      "name" : "coredns",
      "requestsCpu" : "2000m",
      "requestsMem" : "2048Mi"
    } ]
  },
  "flavor4" : {
    "name" : 20000,
    "recommend_cluster_flavor_types" : [ "xlarge" ],
    "replicas" : 4,
    "resources" : [ {
      "limitsCpu" : "2000m",
      "limitsMem" : "2048Mi",
      "name" : "coredns",
      "requestsCpu" : "2000m",
      "requestsMem" : "2048Mi"
    } ]
```



```
        }
       }
      },
      "stable" : true,
      "translate" : {
       "en_US" : {
        "addon" : {
         "changeLog" : "Support autopilot cluster",
         "description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS Services"
        },
        "description" : {
         "Parameters.custom.stub_domains" : "The target nameserver may itself be a Kubernetes service. For instance, you can run your own copy of dnsmasq to export custom DNS names into the ClusterDNS namespace, a JSON map using a DNS suffix key (e.g. “acme.local”) and a value consisting of a JSON array of DNS IPs.",
         "Parameters.custom.upstream_nameservers" : "If specified, then the values specified replace the nameservers taken by default from the node’s /etc/resolv.conf. Limits:a maximum of three upstream nameservers can be specified, A JSON array of DNS IPs.",
         "Parameters.flavor1.description" : "Concurrent domain name resolution ability - External domain name：2500 qps, Internal domain name：10000 qps",
         "Parameters.flavor1.name" : 2500,
         "Parameters.flavor2.description" : "Concurrent domain name resolution ability - External domain name：5000 qps, Internal domain name：20000 qps",
         "Parameters.flavor2.name" : 5000,
         "Parameters.flavor3.description" : "Concurrent domain name resolution ability - External domain name：10000 qps, Internal domain name：40000 qps",
         "Parameters.flavor3.name" : 10000,
         "Parameters.flavor4.description" : "Concurrent domain name resolution ability - External domain name：20000 qps, Internal domain name：80000 qps",
         "Parameters.flavor4.name" : 20000
        },
        "key" : {
         "Parameters.custom.stub_domains" : "stub domain",
         "Parameters.custom.upstream_nameservers" : "upstream nameservers"
        }
       },
       "fr_FR" : {
        "addon" : {
         "changeLog" : "les spécifications du plugin peuvent être associées aux spécifications du cluster. le fuseau horaire du plug-in est le même que celui du noeud",
         "description" : "Un serveur DNS qui enchaîne les plug-ins et fournit des services DNS Kubernetes."
        },
        "description" : {
         "Parameters.custom.stub_domains" : "Le serveur de noms cible peut lui-même être un service Kubernetes. Par exemple, vous pouvez exécuter votre propre copie de dnsmasq pour exporter des noms DNS personnalisés dans l'espace de noms ClusterDNS, une carte JSON à l'aide d'une clé de suffixe DNS (par exemple, «acme.local») et une valeur constituée d'un tableau JSON d'adresses IP DNS.",
         "Parameters.custom.upstream_nameservers" : "Si spécifié, les valeurs spécifiées remplacent les serveurs de noms pris par défaut dans le fichier /etc/resolv.conf du nœud. Limites: un maximum de trois serveurs de noms en amont peuvent être spécifiés, un tableau JSON d'adresses IP DNS.",
         "Parameters.flavor1.description" : "Capacité de résolution de nom de domaine simultanée - Nom de domaine externe: 2500 qps, Nom de domaine interne: 10000 qp",
         "Parameters.flavor1.name" : 2500,
         "Parameters.flavor2.description" : "Capacité de résolution de nom de domaine simultanée - Nom de domaine externe: 5000 qps, Nom de domaine interne: 20000 qp",
         "Parameters.flavor2.name" : 5000,
         "Parameters.flavor3.description" : "Capacité de résolution de nom de domaine simultanée - Nom de domaine externe: 10000 qps, Nom de domaine interne: 40000 qp",
         "Parameters.flavor3.name" : 10000,
         "Parameters.flavor4.description" : "Capacité de résolution de nom de domaine simultanée - Nom de domaine externe: 20000 qps, Nom de domaine interne: 80000 qp",
         "Parameters.flavor4.name" : 20000
        },
        "key" : {
         "Parameters.custom.stub_domains" : "domaine stub",
         "Parameters.custom.upstream_nameservers" : "serveurs de noms en amont"
        }
       },
       "zh_CN" : {
```

```
        "addon" : {
          "changeLog" : "支持autopilot集群",
          "description" : "CoreDNS是一款通过链式插件的方式给Kubernetes提供DNS解析服务的DNS服务器"
        },
        "description" : {
          "Parameters.custom.stub_domains" : "用户可对自定义的域名配置域名服务器，格式为一个键值对，键为DNS后缀域名，值为一个或一组DNS IP地址，如\"acme.local -- 1.2.3.4,6.7.8.9\"。",
          "Parameters.custom.upstream_nameservers" : "解析除集群内服务域名以及自定义域名之外的域名地址，格式为一个或一组DNS IP地址，如\"8.8.8.8\",\"8.8.4.4\"。",
          "Parameters.flavor1.description" : "并发域名解析能力 - 外部域名：2500 qps，内部域名：10000 qps",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "并发域名解析能力 - 外部域名：5000 qps，内部域名：20000 qps",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "并发域名解析能力 - 外部域名：10000 qps，内部域名：40000 qps",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "并发域名解析能力 - 外部域名：20000 qps，内部域名：80000 qps",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "存根域",
          "Parameters.custom.upstream_nameservers" : "上游域名服务器"
        }
      }
    },
    "supportVersions" : null,
    "creationTimestamp" : "2024-02-19T11:33:46Z",
    "updateTimestamp" : "2024-02-21T01:24:05Z"
  }
 }
}
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotAddonInstanceSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
```

```
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotAddonInstanceRequest request = new ShowAutopilotAddonInstanceRequest();
        request.withId("{id}");
        try {
            ShowAutopilotAddonInstanceResponse response = client.showAutopilotAddonInstance(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotAddonInstanceRequest()
        request.id = "{id}"
        response = client.show_autopilot_addon_instance(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)
```



```
func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotAddonInstanceRequest{}
    request.Id = "{id}"
    response, err := client.ShowAutopilotAddonInstance(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.2.7 获取 AddonInstance 列表

功能介绍

获取集群所有已安装插件实例

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/addons

表 4-200 Query 参数

参数	是否必选	参数类型	描述
cluster_id	是	String	集群 ID，获取方式请参见**如何获取接口URI中参数**

请求参数

表 4-201 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-202 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“Addon”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
items	Array of **AddonInstance** objects	插件实例列表

表 4-203 AddonInstance

参数	参数类型	描述
kind	String	API类型，固定值“Addon”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。

参数	参数类型	描述
metadata	**AddonMetadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	**InstanceSpec** object	spec是集合类的元素类型，内容为插件实例具体信息，实例的详细描述主体部分都在spec中给出
status	**AddonInstanceStatus** object	插件实例状态

表 4-204 AddonMetadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	插件名称
alias	String	插件别名
labels	Map<String,String>	插件标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	插件注解，由key/value组成 • 安装：固定值为{"addon.install/type":"install"} • 升级：固定值为{"addon.upgrade/type":"upgrade"}
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-205 InstanceSpec

参数	参数类型	描述
clusterID	String	集群id
version	String	插件模板版本号，如1.0.0
addonTemplateName	String	插件模板名称，如coredns
addonTemplateType	String	插件模板类型

参数	参数类型	描述
addonTemplateLogo	String	插件模板logo图片的地址
addonTemplateLabels	Array of strings	插件模板所属类型
description	String	插件模板描述
values	Map<String,Object>	插件模板安装参数（各插件不同），请根据具体插件模板信息填写安装参数。

表 4-206 AddonInstanceStatus

参数	参数类型	描述
status	String	插件实例状态, 取值如下 ● running：运行中，表示插件全部实例状态都在运行中，插件正常使用。 ● abnormal：不可用，表示插件状态异常，插件不可使用。可单击插件名称查看实例异常事件。 ● installing：安装中，表示插件正在安装中。 ● installFailed：安装失败，表示插件安装失败，需要卸载后重新安装。 ● upgrading：升级中，表示插件正在更新中。 ● upgradeFailed：升级失败，表示插件升级失败，可重试升级或卸载后重新安装。 ● deleting：删除中，表示插件正在删除中。 ● deleteFailed：删除失败，表示插件删除失败，可重试卸载。 ● deleteSuccess：删除成功，表示插件删除成功。 ● available：部分就绪，表示插件下只有部分实例状态为运行中，插件部分功能可用。 ● rollbacking：回滚中，表示插件正在回滚中。 ● rollbackFailed：回滚失败，表示插件回滚失败，可重试回滚或卸载后重新安装。 ● unknown：未知状态，表示插件模板实例不存在。
Reason	String	插件安装失败原因
message	String	安装错误详情
targetVersions	Array of strings	此插件版本，支持升级的集群版本

参数	参数类型	描述
currentVersion	**Versions** object	当前插件实例使用的具体插件版本信息
isRollbackable	Boolean	是否支持回滚到插件升级前的插件版本
previousVersion	String	插件升级或回滚前的版本

表 4-207 Versions

参数	参数类型	描述
version	String	插件版本号
input	Object	插件安装参数
stable	Boolean	是否为稳定版本
translate	Object	供界面使用的翻译信息
supportVersions	Array of **SupportVersions** objects	支持集群版本号
creationTimestamp	String	创建时间
updateTimestamp	String	更新时间

表 4-208 SupportVersions

参数	参数类型	描述
clusterType	String	支持的集群类型
clusterVersion	Array of strings	支持的集群版本（正则表达式）

请求示例

无

响应示例

状态码： 200

ok

```
{
  "kind" : "Addon",
```

```
"apiVersion" : "v3",
"items" : [ {
  "kind" : "Addon",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "90b775e0-5774-4e1d-ab3b-516332ba047a",
    "name" : "coredns",
    "alias" : "coredns",
    "creationTimestamp" : "2024-07-18T04:04:21Z",
    "updateTimestamp" : "2024-07-18T04:04:21Z"
  },
  "spec" : {
    "clusterID" : "597f2d95-44ab-11ef-9e39-0255ac100115",
    "version" : "1.28.6",
    "addonTemplateName" : "coredns",
    "addonTemplateType" : "helm",
    "addonTemplateLogo" : "",
    "addonTemplateLabels" : [ "ContainerNetwork" ],
    "description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS Services",
    "values" : null
  },
  "status" : {
    "status" : "installing",
    "Reason" : "",
    "message" : "",
    "targetVersions" : null,
    "isRollbackable" : false,
    "currentVersion" : {
      "version" : "1.28.6",
      "input" : {
        "basic" : {
          "cluster_ip" : "10.247.3.10",
          "image_version" : "1.28.6",
          "swr_addr" : "swr.cn-north-7.myhuaweicloud.com",
          "swr_user" : "autopilot-official"
        },
        "parameters" : {
          "autopilot-flavor1" : {
            "category" : [ "Autopilot" ],
            "is_default" : true,
            "name" : "autopilot-flavor1",
            "replicas" : 2,
            "resources" : [ {
              "limitsCpu" : 1,
              "limitsMem" : "2Gi",
              "name" : "coredns",
              "requestsCpu" : 1,
              "requestsMem" : "2Gi"
            } ]
          },
          "custom" : {
            "multiAZBalance" : false,
            "multiAZEnabled" : false,
            "node_match_expressions" : [ ],
            "parameterSyncStrategy" : "ensureConsistent",
            "servers" : [ {
              "plugins" : [ {
                "name" : "bind",
                "parameters" : "{$POD_IP}"
              }, {
                "configBlock" : "servfail 5s",
                "name" : "cache",
                "parameters" : 30
              }, {
                "name" : "errors"
              }, {
                "name" : "health",
                "parameters" : "{$POD_IP}:8080"
              }, {
```



```
        "name" : "ready",
        "parameters" : "{$POD_IP}:8081"
      }, {
        "configBlock" : "pods insecure\nfallthrough in-addr.arpa ip6.arpa",
        "name" : "kubernetes",
        "parameters" : "cluster.local in-addr.arpa ip6.arpa"
      }, {
        "name" : "loadbalance",
        "parameters" : "round_robin"
      }, {
        "name" : "prometheus",
        "parameters" : "{$POD_IP}:9153"
      }, {
        "configBlock" : "policy random",
        "name" : "forward",
        "parameters" : ". /etc/resolv.conf"
      }, {
        "name" : "reload"
      } ],
      "port" : 5353,
      "zones" : [ {
        "zone" : "."
      } ]
    } ],
    "stub_domains" : { },
    "tolerations" : [ {
      "effect" : "NoExecute",
      "key" : "node.kubernetes.io/not-ready",
      "operator" : "Exists",
      "tolerationSeconds" : 60
    }, {
      "effect" : "NoExecute",
      "key" : "node.kubernetes.io/unreachable",
      "operator" : "Exists",
      "tolerationSeconds" : 60
    } ],
    "upstream_nameservers" : [ ]
  },
  "flavor1" : {
    "is_default" : true,
    "name" : 2500,
    "recommend_cluster_flavor_types" : [ "small" ],
    "replicas" : 2,
    "resources" : [ {
      "limitsCpu" : "500m",
      "limitsMem" : "512Mi",
      "name" : "coredns",
      "requestsCpu" : "500m",
      "requestsMem" : "512Mi"
    } ]
  },
  "flavor2" : {
    "name" : 5000,
    "recommend_cluster_flavor_types" : [ "medium" ],
    "replicas" : 2,
    "resources" : [ {
      "limitsCpu" : "1000m",
      "limitsMem" : "1024Mi",
      "name" : "coredns",
      "requestsCpu" : "1000m",
      "requestsMem" : "1024Mi"
    } ]
  },
  "flavor3" : {
    "name" : 10000,
    "recommend_cluster_flavor_types" : [ "large" ],
    "replicas" : 2,
    "resources" : [ {
      "limitsCpu" : "2000m",
```

```
            "limitsMem" : "2048Mi",
            "name" : "coredns",
            "requestsCpu" : "2000m",
            "requestsMem" : "2048Mi"
          } ]
        },
        "flavor4" : {
          "name" : 20000,
          "recommend_cluster_flavor_types" : [ "xlarge" ],
          "replicas" : 4,
          "resources" : [ {
            "limitsCpu" : "2000m",
            "limitsMem" : "2048Mi",
            "name" : "coredns",
            "requestsCpu" : "2000m",
            "requestsMem" : "2048Mi"
          } ]
        }
      }
    },
    "stable" : true,
    "translate" : {
      "en_US" : {
        "addon" : {
          "changeLog" : "Support autopilot cluster",
          "description" : "CoreDNS is a DNS server that chains plugins and provides Kubernetes DNS
Services"
        },
        "description" : {
          "Parameters.custom.stub_domains" : "The target nameserver may itself be a Kubernetes service.
For instance, you can run your own copy of dnsmasq to export custom DNS names into the ClusterDNS
namespace, a JSON map using a DNS suffix key (e.g. “acme.local”) and a value consisting of a JSON
array of DNS IPs.",
          "Parameters.custom.upstream_nameservers" : "If specified, then the values specified replace the
nameservers taken by default from the node’s /etc/resolv.conf. Limits:a maximum of three upstream
nameservers can be specified, A JSON array of DNS IPs.",
          "Parameters.flavor1.description" : "Concurrent domain name resolution ability - External domain
name：2500 qps, Internal domain name：10000 qps",
          "Parameters.flavor1.name" : 2500,
          "Parameters.flavor2.description" : "Concurrent domain name resolution ability - External domain
name：5000 qps, Internal domain name：20000 qps",
          "Parameters.flavor2.name" : 5000,
          "Parameters.flavor3.description" : "Concurrent domain name resolution ability - External domain
name：10000 qps, Internal domain name：40000 qps",
          "Parameters.flavor3.name" : 10000,
          "Parameters.flavor4.description" : "Concurrent domain name resolution ability - External domain
name：20000 qps, Internal domain name：80000 qps",
          "Parameters.flavor4.name" : 20000
        },
        "key" : {
          "Parameters.custom.stub_domains" : "stub domain",
          "Parameters.custom.upstream_nameservers" : "upstream nameservers"
        }
      },
      "fr_FR" : {
        "addon" : {
          "changeLog" : "les spécifications du plugin peuvent être associées aux spécifications du cluster. le
fuseau horaire du plug-in est le même que celui du noeud",
          "description" : "Un serveur DNS qui enchaîne les plug-ins et fournit des services DNS Kubernetes."
        },
        "description" : {
          "Parameters.custom.stub_domains" : "Le serveur de noms cible peut lui-même être un service
Kubernetes. Par exemple, vous pouvez exécuter votre propre copie de dnsmasq pour exporter des noms
DNS personnalisés dans l'espace de noms ClusterDNS, une carte JSON à l'aide d'une clé de suffixe DNS (par
exemple, «acme.local») et une valeur constituée d'un tableau JSON d'adresses IP DNS.",
          "Parameters.custom.upstream_nameservers" : "Si spécifié, les valeurs spécifiées remplacent les
serveurs de noms pris par défaut dans le fichier /etc/resolv.conf du nœud. Limites: un maximum de trois
serveurs de noms en amont peuvent être spécifiés, un tableau JSON d'adresses IP DNS.",
          "Parameters.flavor1.description" : "Capacité de résolution de nom de domaine simultanée - Nom
```



```
de domaine externe: 2500 qps, Nom de domaine interne: 10000 qp",
            "Parameters.flavor1.name" : 2500,
            "Parameters.flavor2.description" : "Capacité de résolution de nom de domaine simultanée - Nom
de domaine externe: 5000 qps, Nom de domaine interne: 20000 qp",
            "Parameters.flavor2.name" : 5000,
            "Parameters.flavor3.description" : "Capacité de résolution de nom de domaine simultanée - Nom
de domaine externe: 10000 qps, Nom de domaine interne: 40000 qp",
            "Parameters.flavor3.name" : 10000,
            "Parameters.flavor4.description" : "Capacité de résolution de nom de domaine simultanée - Nom
de domaine externe: 20000 qps, Nom de domaine interne: 80000 qp",
            "Parameters.flavor4.name" : 20000
          },
          "key" : {
            "Parameters.custom.stub_domains" : "domaine stub",
            "Parameters.custom.upstream_nameservers" : "serveurs de noms en amont"
          }
        },
        "zh_CN" : {
          "addon" : {
            "changeLog" : "支持autopilot集群",
            "description" : "CoreDNS是一款通过链式插件的方式给Kubernetes提供DNS解析服务的DNS服务器"
          },
          "description" : {
            "Parameters.custom.stub_domains" : "用户可对自定义的域名配置域名服务器，格式为一个键值对，
键为DNS后缀域名，值为一个或一组DNS IP地址，如\"acme.local -- 1.2.3.4,6.7.8.9\"。",
            "Parameters.custom.upstream_nameservers" : "解析除集群内服务域名以及自定义域名之外的域名地
址，格式为一个或一组DNS IP地址，如\"8.8.8.8\",\"8.8.4.4\"。",
            "Parameters.flavor1.description" : "并发域名解析能力 - 外部域名：2500 qps，内部域名：10000
qps",
            "Parameters.flavor1.name" : 2500,
            "Parameters.flavor2.description" : "并发域名解析能力 - 外部域名：5000 qps，内部域名：20000
qps",
            "Parameters.flavor2.name" : 5000,
            "Parameters.flavor3.description" : "并发域名解析能力 - 外部域名：10000 qps，内部域名：40000
qps",
            "Parameters.flavor3.name" : 10000,
            "Parameters.flavor4.description" : "并发域名解析能力 - 外部域名：20000 qps，内部域名：80000
qps",
            "Parameters.flavor4.name" : 20000
          },
          "key" : {
            "Parameters.custom.stub_domains" : "存根域",
            "Parameters.custom.upstream_nameservers" : "上游域名服务器"
          }
        }
      },
      "supportVersions" : null,
      "creationTimestamp" : "2024-02-19T11:33:46Z",
      "updateTimestamp" : "2024-02-21T01:24:05Z"
    }
  }
 } ]
}
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
```

```
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotAddonInstancesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotAddonInstancesRequest request = new ListAutopilotAddonInstancesRequest();
        try {
            ListAutopilotAddonInstancesResponse response = client.listAutopilotAddonInstances(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotAddonInstancesRequest()
```

```
        response = client.list_autopilot_addon_instances(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListAutopilotAddonInstancesRequest{}
    response, err := client.ListAutopilotAddonInstances(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	ok

错误码

请参见**错误码**。

4.3 集群升级（Autopilot）

4.3.1 集群升级

功能介绍

集群升级。

说明

- 集群升级涉及多维度的组件升级操作，强烈建议统一通过CCE控制台执行交互式升级，降低集群升级过程的业务意外受损风险；
- 当前集群升级相关接口受限开放。

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgrade

表 4-209 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-210 请求 Body 参数

参数	是否必选	参数类型	描述
metadata	是	**UpgradeClusterRequestMetadata** object	升级元数据
spec	是	**UpgradeSpec** object	升级配置信息

表 4-211 UpgradeClusterRequestMetadata

参数	是否必选	参数类型	描述
apiVersion	是	String	api版本，默认为v3
kind	是	String	资源类型，默认为UpgradeTask

表 4-212 UpgradeSpec

参数	是否必选	参数类型	描述
clusterUpgradeAction	否	**ClusterUpgradeAction** object	集群升级配置

表 4-213 ClusterUpgradeAction

参数	是否必选	参数类型	描述
addons	否	Array of **UpgradeAddonConfig** objects	插件配置列表
nodeOrder	否	Map<String,Array<**NodePriority**>>	节点池内节点升级顺序配置。 **说明** key表示节点池ID，默认节点池取值为"DefaultPool"
nodePoolOrder	否	Map<String,Integer>	节点池升级顺序配置，key/value对格式。 **说明** key表示节点池ID，默认节点池取值为"DefaultPool" value表示对应节点池的优先级，默认值为0，优先级最低，数值越大优先级越高
strategy	是	**UpgradeStrategy** object	升级策略
targetVersion	是	String	目标集群版本，例如"v1.23"

表 4-214 UpgradeAddonConfig

参数	是否必选	参数类型	描述
addonTemplateName	是	String	插件名称

参数	是否必选	参数类型	描述
operation	是	String	执行动作，当前升级场景支持操作为"patch"
version	是	String	目标插件版本号 **说明** 目标插件版本必须与目标集群版本配套 Standard和Turbo集群的插件版本与集群版本配套关系见**查询AddonTemplates列表** Autopilot集群的插件版本与集群版本配套关系见**查询AddonTemplates列表**
values	否	Object	插件参数列表，Key:Value格式

表 4-215 NodePriority

参数	是否必选	参数类型	描述
nodeSelector	是	**NodeSelector** object	节点标签选择器，选择一批节点
priority	是	Integer	该批次节点的优先级，默认值为0，优先级最低，数值越大优先级越高

表 4-216 NodeSelector

参数	是否必选	参数类型	描述
key	是	String	标签键
value	否	Array of strings	标签值列表
operator	是	String	标签逻辑运算符

表 4-217 UpgradeStrategy

参数	是否必选	参数类型	描述
type	是	String	升级策略类型，当前仅支持原地升级类型"inPlaceRollingUpdate"

参数	是否必选	参数类型	描述
inPlaceRollingUpdate	否	**InPlaceRollingUpdate** object	原地升级配置，指定原地升级策略类型时必选。

表 4-218 InPlaceRollingUpdate

参数	是否必选	参数类型	描述
userDefinedStep	否	Integer	节点升级步长，取值范围为[1, 40]，建议取值20

响应参数

状态码： 200

表 4-219 响应 Body 参数

参数	参数类型	描述
metadata	**UpgradeCluserResponseMetadata** object	升级任务元数据信息
spec	**UpgradeResponseSpec** object	升级配置信息

表 4-220 UpgradeCluserResponseMetadata

参数	参数类型	描述
uid	String	升级任务ID，可通过调用获取集群升级任务详情API查询进展

表 4-221 UpgradeResponseSpec

参数	参数类型	描述
clusterUpgradeAction	**ClusterUpgradeResponseAction** object	集群升级配置

表 4-222 ClusterUpgradeResponseAction

参数	参数类型	描述
version	String	当前集群版本
targetVersion	String	目标集群版本，例如"v1.23"
targetPlatformVersion	String	目标集群的平台版本号，表示集群版本(version)下的内部版本，不支持用户指定。
strategy	**UpgradeStrategy** object	升级策略
config	Object	升级过程中指定的集群配置

表 4-223 UpgradeStrategy

参数	参数类型	描述
type	String	升级策略类型，当前仅支持原地升级类型"inPlaceRollingUpdate"
inPlaceRollingUpdate	**InPlaceRollingUpdate** object	原地升级配置，指定原地升级策略类型时必选。

表 4-224 InPlaceRollingUpdate

参数	参数类型	描述
userDefinedStep	Integer	节点升级步长，取值范围为[1, 40]，建议取值20

请求示例

升级集群至v1.28版本，并设置节点升级步长为20。

```
POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgrade

{
  "metadata" : {
    "apiVersion" : "v3",
    "kind" : "UpgradeTask"
  },
  "spec" : {
    "clusterUpgradeAction" : {
      "strategy" : {
        "type" : "inPlaceRollingUpdate",
        "inPlaceRollingUpdate" : {
          "userDefinedStep" : 20
        }
      },
      "targetVersion" : "v1.23"
    }
```

```
  }
}
```

响应示例

状态码： 200

表示集群升级任务下发成功。

```
{
  "metadata" : {
    "uid" : "976a33e2-f545-11ed-87af-0255ac1002c2"
  },
  "spec" : {
    "clusterUpgradeAction" : {
      "version" : "v1.19.16-r20",
      "targetVersion" : "v1.23.8-r0",
      "targetPlatformVersion" : "cce.10",
      "strategy" : {
        "type" : "inPlaceRollingUpdate",
        "inPlaceRollingUpdate" : {
          "userDefinedStep" : 20
        }
      },
      "config" : { }
    }
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

升级集群至v1.28版本，并设置节点升级步长为20。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class UpgradeAutopilotClusterSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);
```

```
        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        UpgradeAutopilotClusterRequest request = new UpgradeAutopilotClusterRequest();
        request.withClusterId("{cluster_id}");
        UpgradeClusterRequestBody body = new UpgradeClusterRequestBody();
        InPlaceRollingUpdate inPlaceRollingUpdateStrategy = new InPlaceRollingUpdate();
        inPlaceRollingUpdateStrategy.withUserDefinedStep(20);
        UpgradeStrategy strategyClusterUpgradeAction = new UpgradeStrategy();
        strategyClusterUpgradeAction.withType("inPlaceRollingUpdate")
            .withInPlaceRollingUpdate(inPlaceRollingUpdateStrategy);
        ClusterUpgradeAction clusterUpgradeActionSpec = new ClusterUpgradeAction();
        clusterUpgradeActionSpec.withStrategy(strategyClusterUpgradeAction)
            .withTargetVersion("v1.23");
        UpgradeSpec specbody = new UpgradeSpec();
        specbody.withClusterUpgradeAction(clusterUpgradeActionSpec);
        UpgradeClusterRequestMetadata metadatabody = new UpgradeClusterRequestMetadata();
        metadatabody.withApiVersion("v3")
            .withKind("UpgradeTask");
        body.withSpec(specbody);
        body.withMetadata(metadatabody);
        request.withBody(body);
        try {
            UpgradeAutopilotClusterResponse response = client.upgradeAutopilotCluster(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

升级集群至v1.28版本，并设置节点升级步长为20。

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()
```

```
    try:
        request = UpgradeAutopilotClusterRequest()
        request.cluster_id = "{cluster_id}"
        inPlaceRollingUpdateStrategy = InPlaceRollingUpdate(
            user_defined_step=20
        )
        strategyClusterUpgradeAction = UpgradeStrategy(
            type="inPlaceRollingUpdate",
            in_place_rolling_update=inPlaceRollingUpdateStrategy
        )
        clusterUpgradeActionSpec = ClusterUpgradeAction(
            strategy=strategyClusterUpgradeAction,
            target_version="v1.23"
        )
        specbody = UpgradeSpec(
            cluster_upgrade_action=clusterUpgradeActionSpec
        )
        metadatabody = UpgradeClusterRequestMetadata(
            api_version="v3",
            kind="UpgradeTask"
        )
        request.body = UpgradeClusterRequestBody(
            spec=specbody,
            metadata=metadatabody
        )
        response = client.upgrade_autopilot_cluster(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

升级集群至v1.28版本，并设置节点升级步长为20。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())
```

```
    request := &model.UpgradeAutopilotClusterRequest{}
    request.ClusterId = "{cluster_id}"
    userDefinedStepInPlaceRollingUpdate:= int32(20)
    inPlaceRollingUpdateStrategy := &model.InPlaceRollingUpdate{
        UserDefinedStep: &userDefinedStepInPlaceRollingUpdate,
    }
    strategyClusterUpgradeAction := &model.UpgradeStrategy{
        Type: "inPlaceRollingUpdate",
        InPlaceRollingUpdate: inPlaceRollingUpdateStrategy,
    }
    clusterUpgradeActionSpec := &model.ClusterUpgradeAction{
        Strategy: strategyClusterUpgradeAction,
        TargetVersion: "v1.23",
    }
    specbody := &model.UpgradeSpec{
        ClusterUpgradeAction: clusterUpgradeActionSpec,
    }
    metadatabody := &model.UpgradeClusterRequestMetadata{
        ApiVersion: "v3",
        Kind: "UpgradeTask",
    }
    request.Body = &model.UpgradeClusterRequestBody{
        Spec: specbody,
        Metadata: metadatabody,
    }
    response, err := client.UpgradeAutopilotCluster(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示集群升级任务下发成功。

错误码

请参见**错误码**。

4.3.2 获取集群升级任务详情

功能介绍

获取集群升级任务详情，任务ID由调用集群升级API后从响应体中uid字段获取。

说明

- 集群升级涉及多维度的组件升级操作，强烈建议统一通过CCE控制台执行交互式升级，降低集群升级过程的业务意外受损风险；
- 当前集群升级相关接口受限开放。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgrade/tasks/{task_id}

表 4-225 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。
task_id	是	String	升级任务ID，调用集群升级API后从响应体中uid字段获取。

请求参数

无

响应参数

状态码： 200

表 4-226 响应 Body 参数

参数	参数类型	描述
apiVersion	String	api版本，默认为v3
kind	String	资源类型，默认为UpgradeTask
metadata	**UpgradeTask Metadata** object	升级任务元数据信息
spec	**UpgradeTask Spec** object	升级任务信息
status	**UpgradeTask Status** object	升级任务状态

表 4-227 UpgradeTaskMetadata

参数	参数类型	描述
uid	String	升级任务ID
creationTimestamp	String	任务创建时间
updateTimestamp	String	任务更新时间

表 4-228 UpgradeTaskSpec

参数	参数类型	描述
version	String	升级前集群版本
targetVersion	String	升级的目标集群版本
items	Object	升级任务附属信息

表 4-229 UpgradeTaskStatus

参数	参数类型	描述
phase	String	升级任务状态. **说明** Init：初始化 Queuing：等待 Running：运行中 Pause：暂停 Success：成功 Failed：失败
progress	String	升级任务进度
completionTime	String	升级任务结束时间

请求示例

无

响应示例

状态码： 200

表示获取集群升级任务详情成功。

```
{
  "kind" : "UpgradeTask",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "creationTimestamp" : "2022-12-16 13:40:20.75671 +0800 CST",
    "updateTimestamp" : "2022-12-16 13:40:20.756712 +0800 CST"
  },
  "spec" : {
    "version" : "v1.19.16-r4",
    "targetVersion" : "v1.23.5-r0"
  },
  "status" : {
    "phase" : "Init",
    "progress" : "0.00",
    "completionTime" : "2022-12-16 13:40:20.756712 +0800 CST"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotUpgradeClusterTaskSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotUpgradeClusterTaskRequest request = new ShowAutopilotUpgradeClusterTaskRequest();
        request.withClusterId("{cluster_id}");
        request.withTaskId("{task_id}");
        try {
            ShowAutopilotUpgradeClusterTaskResponse response = client.showAutopilotUpgradeClusterTask(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
```



```
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotUpgradeClusterTaskRequest()
        request.cluster_id = "{cluster_id}"
        request.task_id = "{task_id}"
        response = client.show_autopilot_upgrade_cluster_task(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
```



```
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotUpgradeClusterTaskRequest{}
    request.ClusterId = "{cluster_id}"
    request.TaskId = "{task_id}"
    response, err := client.ShowAutopilotUpgradeClusterTask(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群升级任务详情成功。

错误码

请参见**错误码**。

4.3.3 重试集群升级任务

功能介绍

重新执行失败的集群升级任务。

说明

- 集群升级涉及多维度的组件升级操作，强烈建议统一通过CCE控制台执行交互式升级，降低集群升级过程的业务意外受损风险；
- 当前集群升级相关接口受限开放。

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgrade/retry

表 4-230 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

无

响应参数

无

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class RetryAutopilotUpgradeClusterTaskSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
```

```
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        RetryAutopilotUpgradeClusterTaskRequest request = new RetryAutopilotUpgradeClusterTaskRequest();
        request.withClusterId("{cluster_id}");
        try {
            RetryAutopilotUpgradeClusterTaskResponse response =
client.retryAutopilotUpgradeClusterTask(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = RetryAutopilotUpgradeClusterTaskRequest()
        request.cluster_id = "{cluster_id}"
        response = client.retry_autopilot_upgrade_cluster_task(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
```



```
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.RetryAutopilotUpgradeClusterTaskRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.RetryAutopilotUpgradeClusterTask(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示重试集群升级任务下发成功。

错误码

请参见**错误码**。

4.3.4 获取集群升级任务详情列表

功能介绍

获取集群升级任务详情列表

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgrade/tasks

表 4-231 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

无

响应参数

状态码： 200

表 4-232 响应 Body 参数

参数	参数类型	描述
apiVersion	String	api版本，默认为v3
kind	String	资源类型
metadata	**UpgradeTaskMetadata** object	元数据信息
items	Array of **UpgradeTaskResponseBody** objects	集群升级任务列表

表 4-233 UpgradeTaskResponseBody

参数	参数类型	描述
apiVersion	String	api版本，默认为v3
kind	String	资源类型，默认为UpgradeTask
metadata	**UpgradeTaskMetadata** object	升级任务元数据信息
spec	**UpgradeTaskSpec** object	升级任务信息
status	**UpgradeTaskStatus** object	升级任务状态

表 4-234 UpgradeTaskMetadata

参数	参数类型	描述
uid	String	升级任务ID
creationTimestamp	String	任务创建时间
updateTimestamp	String	任务更新时间

表 4-235 UpgradeTaskSpec

参数	参数类型	描述
version	String	升级前集群版本
targetVersion	String	升级的目标集群版本
items	Object	升级任务附属信息

表 4-236 UpgradeTaskStatus

参数	参数类型	描述
phase	String	升级任务状态. **说明** Init：初始化 Queuing：等待 Running：运行中 Pause：暂停 Success：成功 Failed：失败
progress	String	升级任务进度
completionTime	String	升级任务结束时间

请求示例

无

响应示例

状态码： 200

表示获取集群升级任务详情列表成功。

```
{
  "kind" : "List",
  "apiVersion" : "v3",
  "metadata" : { },
  "items" : [ {
    "kind" : "UpgradeTask",
    "apiVersion" : "v3",
    "metadata" : {
      "uid" : "f40cafed-7bf1-4c3b-b619-80113b4bbb18",
      "creationTimestamp" : "2023-11-24 16:41:12.09236 +0800 CST",
      "updateTimestamp" : "2023-11-24 16:44:05.634206 +0800 CST"
    },
    "spec" : {
      "version" : "v1.17.17-r0",
      "targetVersion" : "v1.19.16-r80"
    },
    "status" : {
      "phase" : "Success",
      "completionTime" : "2023-11-24 16:44:05.634206 +0800 CST"
    }
  }, {
    "kind" : "UpgradeTask",
    "apiVersion" : "v3",
    "metadata" : {
      "uid" : "91755b96-5fd8-4a6a-bda1-983de9055996",
      "creationTimestamp" : "2023-11-24 19:54:35.194306 +0800 CST",
      "updateTimestamp" : "2023-11-24 20:14:35.194306 +0800 CST"
    },
    "spec" : {
      "version" : "v1.19.16-r80",
      "targetVersion" : "v1.23.8-r10"
    },
```

```
   "status" : {
     "phase" : "Success",
     "completionTime" : "2023-11-24 20:14:35.194306 +0800 CST"
   }
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotUpgradeClusterTasksSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotUpgradeClusterTasksRequest request = new ListAutopilotUpgradeClusterTasksRequest();
        request.withClusterId("{cluster_id}");
        try {
            ListAutopilotUpgradeClusterTasksResponse response = client.listAutopilotUpgradeClusterTasks(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```


Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotUpgradeClusterTasksRequest()
        request.cluster_id = "{cluster_id}"
        response = client.list_autopilot_upgrade_cluster_tasks(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
        WithCredential(auth).
        Build())

    request := &model.ListAutopilotUpgradeClusterTasksRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ListAutopilotUpgradeClusterTasks(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群升级任务详情列表成功。

错误码

请参见**错误码**。

4.3.5 集群升级前检查

功能介绍

集群升级前检查

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/precheck

表 4-237 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-238 请求 Body 参数

参数	是否必选	参数类型	描述
apiVersion	是	String	API版本，默认为v3
kind	是	String	资源类型，默认为PreCheckTask
spec	是	**PrecheckSpec** object	spec是集合类的元素类型，您对需要升级前检查的配置信息的主体部分都在spec中给出。CCE通过spec的描述来执行检查。

表 4-239 PrecheckSpec

参数	是否必选	参数类型	描述
clusterID	否	String	集群ID
clusterVersion	否	String	集群版本
targetVersion	否	String	升级目标版本
skippedCheckItemList	否	Array of **skippedCheckItemList** objects	跳过检查的项目列表

表 4-240 skippedCheckItemList

参数	是否必选	参数类型	描述
name	否	String	跳过的检查项名称
resourceSelector	否	**resourceSelector** object	资源标签选择器，仅节点检查涉及该参数，集群检查和插件检查不涉及。

表 4-241 resourceSelector

参数	是否必选	参数类型	描述
key	是	String	标签键值，取值如下 • node.uid：节点UID。
values	否	Array of strings	标签值列表

参数	是否必选	参数类型	描述
operator	是	String	标签逻辑运算符，当前支持如下取值 • In

响应参数

状态码： 200

表 4-242 响应 Body 参数

参数	参数类型	描述
apiVersion	String	API版本
kind	String	资源类型
metadata	**PrecheckCluserResponseMetadata** object	升级前检查元数据
spec	**PrecheckSpec** object	spec是集合类的元素类型，您对需要升级前检查的配置信息的主体部分都在spec中给出。CCE通过spec的描述来执行检查。
status	**PrecheckStatus** object	集群升级前检查状态

表 4-243 PrecheckCluserResponseMetadata

参数	参数类型	描述
uid	String	检查任务ID

表 4-244 PrecheckSpec

参数	参数类型	描述
clusterID	String	集群ID
clusterVersion	String	集群版本
targetVersion	String	升级目标版本

参数	参数类型	描述
skippedCheckItemList	Array of **skippedCheckItemList** objects	跳过检查的项目列表

表 4-245 skippedCheckItemList

参数	参数类型	描述
name	String	跳过的检查项名称
resourceSelector	**resourceSelector** object	资源标签选择器，仅节点检查涉及该参数，集群检查和插件检查不涉及。

表 4-246 resourceSelector

参数	参数类型	描述
key	String	标签键值，取值如下 • node.uid：节点UID。
values	Array of strings	标签值列表
operator	String	标签逻辑运算符，当前支持如下取值 • In

表 4-247 PrecheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败 • Error 错误
expireTimeStamp	String	检查结果过期时间
message	String	信息，一般是执行错误的日志信息
clusterCheckStatus	**clusterCheckStatus** object	集群限制检查状态

参数	参数类型	描述
addonCheckStatus	**addonCheckStatus** object	插件检查状态
nodeCheckStatus	**nodeCheckStatus** object	节点检查状态

表 4-248 clusterCheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
itemsStatus	Array of **PreCheckItemStatus** objects	检查项状态集合

表 4-249 addonCheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
itemsStatus	Array of **PreCheckItemStatus** objects	检查项状态集合

表 4-250 nodeCheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
nodeStageStatus	Array of **NodeStageStatus** objects	节点检查状态

表 4-251 NodeStageStatus

参数	参数类型	描述
nodeInfo	**NodeInfo** object	节点信息
itemsStatus	Array of **PreCheckItemStatus** objects	检查项状态集合

表 4-252 NodeInfo

参数	参数类型	描述
uid	String	节点UID
name	String	节点名称
status	String	状态
nodeType	String	节点类型

表 4-253 PreCheckItemStatus

参数	参数类型	描述
name	String	检查项名称
kind	String	检查项类型，取值如下 • Exception: 异常类，需要用户解决 • Risk：风险类，用户确认后可选择跳过

参数	参数类型	描述
group	String	检查项分组，取值如下 • LimitCheck: 集群限制检查 • MasterCheck：控制节点检查 • NodeCheck：用户节点检查 • AddonCheck：插件检查 • ExecuteException：检查流程错误
level	String	检查项风险级别，取值如下 • Info: 提示级别 • Warning：风险级别 • Fatal：严重级别
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
message	String	提示信息
riskSource	**riskSource** object	风险项
errorCodes	Array of strings	错误码集合

表 4-254 riskSource

参数	参数类型	描述
configurationRisks	Array of **configurationRisks** objects	配置风险项
deprecatedAPIRisks	Array of **deprecatedAPIRisks** objects	废弃API风险
nodeRisks	Array of **nodeRisks** objects	节点风险
addonRisks	Array of **addonRisks** objects	插件风险

表 4-255 configurationRisks

参数	参数类型	描述
package	String	组件名称
sourceFile	String	涉及文件路径
nodeMsg	String	节点信息
field	String	参数值
operation	String	修改操作类型
originalValue	String	原始值
value	String	当前值

表 4-256 deprecatedAPIRisks

参数	参数类型	描述
url	String	请求路径，如/apis/policy/v1beta1/podsecuritypolicies
userAgent	String	客户端信息

表 4-257 nodeRisks

参数	参数类型	描述
NodeID	String	用户节点ID

表 4-258 addonRisks

参数	参数类型	描述
addonTemplateName	String	插件模板名称
alias	String	插件别名

请求示例

集群升级前检查请求体

```
POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/precheck

{
  "kind" : "PreCheckTask",
  "apiVersion" : "v3",
```

```
  "spec" : {
    "clusterID" : "8978deaa-1743-11ee-8e46-0255ac10004c",
    "clusterVersion" : "v1.15.11-r1",
    "targetVersion" : "v1.19.16-r80",
    "skippedCheckItemList" : [ ]
  }
}
```

响应示例

状态码： 200

执行集群升级前检查成功。

```
{
  "kind" : "PreCheckTask",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "9991b45e-a2be-4b49-aca4-50a25fa6f81e"
  },
  "spec" : {
    "clusterID" : "8978deaa-1743-11ee-8e46-0255ac10004c",
    "clusterVersion" : "v1.15.11-r1",
    "targetVersion" : "v1.19.16-r80"
  },
  "status" : {
    "phase" : "Init",
    "clusterCheckStatus" : {
      "phase" : "Init"
    },
    "addonCheckStatus" : {
      "phase" : "Init"
    },
    "nodeCheckStatus" : {
      "phase" : "Init"
    }
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

集群升级前检查请求体

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;

import java.util.List;
import java.util.ArrayList;

public class CreateAutopilotPreCheckSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
```

```
environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAutopilotPreCheckRequest request = new CreateAutopilotPreCheckRequest();
        request.withClusterId("{cluster_id}");
        PrecheckClusterRequestBody body = new PrecheckClusterRequestBody();
        PrecheckSpec specbody = new PrecheckSpec();
        specbody.withClusterID("8978deaa-1743-11ee-8e46-0255ac10004c")
            .withClusterVersion("v1.15.11-r1")
            .withTargetVersion("v1.19.16-r80");
        body.withSpec(specbody);
        body.withKind("PreCheckTask");
        body.withApiVersion("v3");
        request.withBody(body);
        try {
            CreateAutopilotPreCheckResponse response = client.createAutopilotPreCheck(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

集群升级前检查请求体

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
```



```
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAutopilotPreCheckRequest()
        request.cluster_id = "{cluster_id}"
        specbody = PrecheckSpec(
            cluster_id="8978deaa-1743-11ee-8e46-0255ac10004c",
            cluster_version="v1.15.11-r1",
            target_version="v1.19.16-r80"
        )
        request.body = PrecheckClusterRequestBody(
            spec=specbody,
            kind="PreCheckTask",
            api_version="v3"
        )
        response = client.create_autopilot_pre_check(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

集群升级前检查请求体

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAutopilotPreCheckRequest{}
    request.ClusterId = "{cluster_id}"
    clusterIDSpec:= "8978deaa-1743-11ee-8e46-0255ac10004c"
    clusterVersionSpec:= "v1.15.11-r1"
    targetVersionSpec:= "v1.19.16-r80"
    specbody := &model.PrecheckSpec{
        ClusterID: &clusterIDSpec,
        ClusterVersion: &clusterVersionSpec,
```

```
        TargetVersion: &targetVersionSpec,
    }
    request.Body = &model.PrecheckClusterRequestBody{
        Spec: specbody,
        Kind: "PreCheckTask",
        ApiVersion: "v3",
    }
    response, err := client.CreateAutopilotPreCheck(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	执行集群升级前检查成功。

错误码

请参见**错误码**。

4.3.6 获取集群升级前检查任务详情

功能介绍

获取集群升级前检查任务详情，任务ID由调用集群检查API后从响应体中uid字段获取。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/precheck/tasks/{task_id}

表 4-259 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。

参数	是否必选	参数类型	描述
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。
task_id	是	String	升级任务ID，调用集群升级API后从响应体中uid字段获取。

请求参数

无

响应参数

状态码： 200

表 4-260 响应 Body 参数

参数	参数类型	描述
apiVersion	String	api版本，默认为v3
kind	String	资源类型，默认为PreCheckTask
metadata	**PrecheckTaskMetadata** object	升级前检查任务元数据信息
spec	**PrecheckSpec** object	升级前检查任务信息
status	**PrecheckStatus** object	升级前检查任务状态

表 4-261 PrecheckTaskMetadata

参数	参数类型	描述
uid	String	任务ID
creationTimestamp	String	任务创建时间
updateTimestamp	String	任务更新时间

表 4-262 PrecheckSpec

参数	参数类型	描述
clusterID	String	集群ID
clusterVersion	String	集群版本
targetVersion	String	升级目标版本
skippedCheckItemList	Array of **skippedCheckItemList** objects	跳过检查的项目列表

表 4-263 skippedCheckItemList

参数	参数类型	描述
name	String	跳过的检查项名称
resourceSelector	**resourceSelector** object	资源标签选择器，仅节点检查涉及该参数，集群检查和插件检查不涉及。

表 4-264 resourceSelector

参数	参数类型	描述
key	String	标签键值，取值如下 • node.uid：节点UID。
values	Array of strings	标签值列表
operator	String	标签逻辑运算符，当前支持如下取值 • In

表 4-265 PrecheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败 • Error 错误

参数	参数类型	描述
expireTimeStamp	String	检查结果过期时间
message	String	信息，一般是执行错误的日志信息
clusterCheckStatus	**clusterCheckStatus** object	集群限制检查状态
addonCheckStatus	**addonCheckStatus** object	插件检查状态
nodeCheckStatus	**nodeCheckStatus** object	节点检查状态

表 4-266 clusterCheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
itemsStatus	Array of **PreCheckItemStatus** objects	检查项状态集合

表 4-267 addonCheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
itemsStatus	Array of **PreCheckItemStatus** objects	检查项状态集合

表 4-268 nodeCheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
nodeStageStatus	Array of **NodeStageStatus** objects	节点检查状态

表 4-269 NodeStageStatus

参数	参数类型	描述
nodeInfo	**NodeInfo** object	节点信息
itemsStatus	Array of **PreCheckItemStatus** objects	检查项状态集合

表 4-270 NodeInfo

参数	参数类型	描述
uid	String	节点UID
name	String	节点名称
status	String	状态
nodeType	String	节点类型

表 4-271 PreCheckItemStatus

参数	参数类型	描述
name	String	检查项名称
kind	String	检查项类型，取值如下 • Exception: 异常类，需要用户解决 • Risk：风险类，用户确认后可选择跳过

参数	参数类型	描述
group	String	检查项分组，取值如下 • LimitCheck: 集群限制检查 • MasterCheck：控制节点检查 • NodeCheck：用户节点检查 • AddonCheck：插件检查 • ExecuteException：检查流程错误
level	String	检查项风险级别，取值如下 • Info: 提示级别 • Warning：风险级别 • Fatal：严重级别
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
message	String	提示信息
riskSource	**riskSource** object	风险项
errorCodes	Array of strings	错误码集合

表 4-272 riskSource

参数	参数类型	描述
configurationRisks	Array of **configurationRisks** objects	配置风险项
deprecatedAPIRisks	Array of **deprecatedAPIRisks** objects	废弃API风险
nodeRisks	Array of **nodeRisks** objects	节点风险
addonRisks	Array of **addonRisks** objects	插件风险

表 4-273 configurationRisks

参数	参数类型	描述
package	String	组件名称
sourceFile	String	涉及文件路径
nodeMsg	String	节点信息
field	String	参数值
operation	String	修改操作类型
originalValue	String	原始值
value	String	当前值

表 4-274 deprecatedAPIRisks

参数	参数类型	描述
url	String	请求路径，如/apis/policy/v1beta1/podsecuritypolicies
userAgent	String	客户端信息

表 4-275 nodeRisks

参数	参数类型	描述
NodeID	String	用户节点ID

表 4-276 addonRisks

参数	参数类型	描述
addonTemplateName	String	插件模板名称
alias	String	插件别名

请求示例

无

响应示例

状态码： 200

表示获取集群升级前检查任务详情成功。


```
{
  "kind" : "PreCheckTask",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "f61e008c-1600-41c0-9bde-121de5a30660",
    "creationTimestamp" : "2023-11-25 07:20:04.592972 +0000 UTC",
    "updateTimestamp" : "2023-11-25 07:21:05.518966 +0000 UTC"
  },
  "spec" : {
    "clusterVersion" : "v1.19.16-r4",
    "targetVersion" : "v1.23.5-r0"
  },
  "status" : {
    "phase" : "Success",
    "expireTimeStamp" : "2023-11-25 08:21:05.518966 +0000 UTC",
    "clusterCheckStatus" : {
      "phase" : "Success",
      "itemsStatus" : [ {
        "name" : "DeprecatedApiCheck",
        "kind" : "Risk",
        "group" : "LimitCheck",
        "level" : "Info",
        "phase" : "Success",
        "message" : "check item succeed",
        "riskSource" : { }
      }, {
        "name" : "NodeContainerdPodRestartRisk",
        "kind" : "Risk",
        "group" : "LimitCheck",
        "level" : "Warning",
        "phase" : "Success",
        "message" : "check item succeed",
        "riskSource" : { }
      }, {
        "name" : "ResiduePackageVersion",
        "kind" : "Exception",
        "group" : "LimitCheck",
        "level" : "Fatal",
        "phase" : "Success",
        "message" : "check item succeed",
        "riskSource" : { }
      } ]
    },
    "addonCheckStatus" : {
      "phase" : "Success",
      "itemsStatus" : [ {
        "name" : "AddonLimit",
        "kind" : "Exception",
        "group" : "AddonCheck",
        "level" : "Warning",
        "phase" : "Success",
        "message" : "check item succeed",
        "riskSource" : { }
      }, {
        "name" : "CoreDNSConfLimit",
        "kind" : "Exception",
        "group" : "AddonCheck",
        "level" : "Fatal",
        "phase" : "Success",
        "message" : "check item succeed",
        "riskSource" : { }
      } ]
    },
    "nodeCheckStatus" : {
      "phase" : "Success"
    }
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotPreCheckSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotPreCheckRequest request = new ShowAutopilotPreCheckRequest();
        request.withClusterId("{cluster_id}");
        request.withTaskId("{task_id}");
        try {
            ShowAutopilotPreCheckResponse response = client.showAutopilotPreCheck(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *
```

```
if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotPreCheckRequest()
        request.cluster_id = "{cluster_id}"
        request.task_id = "{task_id}"
        response = client.show_autopilot_pre_check(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotPreCheckRequest{}
    request.ClusterId = "{cluster_id}"
    request.TaskId = "{task_id}"
    response, err := client.ShowAutopilotPreCheck(request)
    if err == nil {
```



```
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群升级前检查任务详情成功。

错误码

请参见**错误码**。

4.3.7 获取集群升级前检查任务详情列表

功能介绍

获取集群升级前检查任务详情列表

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/precheck/tasks

表 4-277 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

无

响应参数

状态码： 200

表 4-278 响应 Body 参数

参数	参数类型	描述
apiVersion	String	api版本，默认为v3
kind	String	类型
metadata	**Metadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
items	Array of **PrecheckClusterTask** objects	集群检查任务列表

表 4-279 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-280 PrecheckClusterTask

参数	参数类型	描述
apiVersion	String	api版本，默认为v3
kind	String	资源类型，默认为PreCheckTask
metadata	**PrecheckTaskMetadata** object	升级前检查任务元数据信息

参数	参数类型	描述
spec	**PrecheckSpec** object	升级前检查任务信息
status	**PrecheckStatus** object	升级前检查任务状态

表 4-281 PrecheckTaskMetadata

参数	参数类型	描述
uid	String	任务ID
creationTimestamp	String	任务创建时间
updateTimestamp	String	任务更新时间

表 4-282 PrecheckSpec

参数	参数类型	描述
clusterID	String	集群ID
clusterVersion	String	集群版本
targetVersion	String	升级目标版本
skippedCheckItemList	Array of **skippedCheckItemList** objects	跳过检查的项目列表

表 4-283 skippedCheckItemList

参数	参数类型	描述
name	String	跳过的检查项名称
resourceSelector	**resourceSelector** object	资源标签选择器，仅节点检查涉及该参数，集群检查和插件检查不涉及。

表 4-284 resourceSelector

参数	参数类型	描述
key	String	标签键值，取值如下 • node.uid：节点UID。
values	Array of strings	标签值列表
operator	String	标签逻辑运算符，当前支持如下取值 • In

表 4-285 PrecheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败 • Error 错误
expireTimeStamp	String	检查结果过期时间
message	String	信息，一般是执行错误的日志信息
clusterCheckStatus	**clusterCheckStatus** object	集群限制检查状态
addonCheckStatus	**addonCheckStatus** object	插件检查状态
nodeCheckStatus	**nodeCheckStatus** object	节点检查状态

表 4-286 clusterCheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败

参数	参数类型	描述
itemsStatus	Array of **PreCheckItemStatus** objects	检查项状态集合

表 4-287 addonCheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
itemsStatus	Array of **PreCheckItemStatus** objects	检查项状态集合

表 4-288 nodeCheckStatus

参数	参数类型	描述
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败
nodeStageStatus	Array of **NodeStageStatus** objects	节点检查状态

表 4-289 NodeStageStatus

参数	参数类型	描述
nodeInfo	**NodeInfo** object	节点信息

参数	参数类型	描述
itemsStatus	Array of **PreCheckItemStatus** objects	检查项状态集合

表 4-290 NodeInfo

参数	参数类型	描述
uid	String	节点UID
name	String	节点名称
status	String	状态
nodeType	String	节点类型

表 4-291 PreCheckItemStatus

参数	参数类型	描述
name	String	检查项名称
kind	String	检查项类型，取值如下 • Exception: 异常类，需要用户解决 • Risk：风险类，用户确认后可选择跳过
group	String	检查项分组，取值如下 • LimitCheck: 集群限制检查 • MasterCheck：控制节点检查 • NodeCheck：用户节点检查 • AddonCheck：插件检查 • ExecuteException：检查流程错误
level	String	检查项风险级别，取值如下 • Info: 提示级别 • Warning：风险级别 • Fatal：严重级别
phase	String	状态，取值如下 • Init: 初始化 • Running 运行中 • Success 成功 • Failed 失败

参数	参数类型	描述
message	String	提示信息
riskSource	**riskSource** object	风险项
errorCodes	Array of strings	错误码集合

表 4-292 riskSource

参数	参数类型	描述
configurationRisks	Array of **configurationRisks** objects	配置风险项
deprecatedAPIRisks	Array of **deprecatedAPIRisks** objects	废弃API风险
nodeRisks	Array of **nodeRisks** objects	节点风险
addonRisks	Array of **addonRisks** objects	插件风险

表 4-293 configurationRisks

参数	参数类型	描述
package	String	组件名称
sourceFile	String	涉及文件路径
nodeMsg	String	节点信息
field	String	参数值
operation	String	修改操作类型
originalValue	String	原始值
value	String	当前值

表 4-294 deprecatedAPIRisks

参数	参数类型	描述
url	String	请求路径，如/apis/policy/v1beta1/podsecuritypolicies
userAgent	String	客户端信息

表 4-295 nodeRisks

参数	参数类型	描述
NodeID	String	用户节点ID

表 4-296 addonRisks

参数	参数类型	描述
addonTemplateName	String	插件模板名称
alias	String	插件别名

请求示例

无

响应示例

状态码： 200

表示获取集群升级前检查任务详情列表成功。


```
{
  "kind" : "List",
  "apiVersion" : "v3",
  "metadata" : { },
  "items" : [ {
    "kind" : "PreCheckTask",
    "apiVersion" : "v3",
    "metadata" : {
      "uid" : "10b52d23-080a-4b7d-bf83-64b4687ca786",
      "creationTimestamp" : "2023-12-16 07:07:11.099111 +0000 UTC",
      "updateTimestamp" : "2023-12-16 07:09:10.425622 +0000 UTC"
    },
    "spec" : {
      "clusterVersion" : "v1.23.5-r0",
      "targetVersion" : "v1.23.11-r0"
    },
    "status" : {
      "phase" : "Failed",
      "clusterCheckStatus" : {
        "phase" : "Success",
        "itemsStatus" : [ {
```

```
      "name" : "DeprecatedApiCheck",
      "kind" : "Risk",
      "group" : "LimitCheck",
      "level" : "Info",
      "phase" : "Success",
      "message" : "check item succeed",
      "riskSource" : { }
    }, {
      "name" : "BlackLimit",
      "kind" : "Exception",
      "group" : "LimitCheck",
      "level" : "Fatal",
      "phase" : "Success",
      "message" : "check item succeed",
      "riskSource" : { }
    }, {
      "name" : "MasterSSH",
      "kind" : "Exception",
      "group" : "LimitCheck",
      "level" : "Fatal",
      "phase" : "Success",
      "message" : "check item succeed",
      "riskSource" : { }
    }, {
      "name" : "ReleaseLimit",
      "kind" : "Exception",
      "group" : "LimitCheck",
      "level" : "Warning",
      "phase" : "Success",
      "message" : "check item succeed",
      "riskSource" : { }
    }, {
      "name" : "ClusterNoArm",
      "kind" : "Exception",
      "group" : "LimitCheck",
      "level" : "Warning",
      "phase" : "Success",
      "message" : "check item succeed",
      "riskSource" : { }
    } ]
  },
  "addonCheckStatus" : {
    "phase" : "Failed",
    "itemsStatus" : [ {
      "name" : "AddonLimit",
      "kind" : "Exception",
      "group" : "AddonCheck",
      "level" : "Warning",
      "phase" : "Failed",
      "message" : "addon [ CoreDNS ] status is abnormal, check and try again",
      "riskSource" : {
        "addonRisks" : [ {
          "addonTemplateName" : "coredns",
          "alias" : "CoreDNS"
        } ]
      }
    }, {
      "name" : "CoreDNSConfLimit",
      "kind" : "Exception",
      "group" : "AddonCheck",
      "level" : "Fatal",
      "phase" : "Success",
      "message" : "check item succeed",
      "riskSource" : { }
    }, {
      "name" : "EverestLimitHungVersion",
      "kind" : "Risk",
      "group" : "AddonCheck",
      "level" : "Fatal",
```

```
        "phase" : "Success",
        "message" : "check item succeed",
        "riskSource" : { }
      } ]
    },
    "nodeCheckStatus" : {
      "phase" : "Success"
    }
  }
 } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotPreCheckTasksSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotPreCheckTasksRequest request = new ListAutopilotPreCheckTasksRequest();
        request.withClusterId("{cluster_id}");
        try {
            ListAutopilotPreCheckTasksResponse response = client.listAutopilotPreCheckTasks(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
```



```
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotPreCheckTasksRequest()
        request.cluster_id = "{cluster_id}"
        response = client.list_autopilot_pre_check_tasks(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()
```

```
    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListAutopilotPreCheckTasksRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ListAutopilotPreCheckTasks(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群升级前检查任务详情列表成功。

错误码

请参见**错误码**。

4.3.8 集群升级后确认

功能介绍

集群升级后确认，该接口建议配合Console使用，主要用于升级步骤完成后，客户确认集群状态和业务正常后做反馈。

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/postcheck

表 4-297 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。

参数	是否必选	参数类型	描述
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-298 请求 Body 参数

参数	是否必选	参数类型	描述
apiVersion	是	String	API版本，默认为v3
kind	是	String	资源类型
spec	是	**PostcheckSpec** object	spec是升级后确认的配置信息。

表 4-299 PostcheckSpec

参数	是否必选	参数类型	描述
clusterID	否	String	集群ID
clusterVersion	否	String	集群升级源版本
targetVersion	否	String	集群升级目标版本

响应参数

状态码： 200

表 4-300 响应 Body 参数

参数	参数类型	描述
apiVersion	String	API版本
kind	String	资源类型
metadata	**PostcheckClusterResponseMetadata** object	升级后确认元数据
spec	**PostcheckSpec** object	集群升级后确认的配置信息
status	**status** object	集群升级后确认的状态信息

表 4-301 PostcheckCluserResponseMetadata

参数	参数类型	描述
uid	String	任务ID

表 4-302 PostcheckSpec

参数	参数类型	描述
clusterID	String	集群ID
clusterVersion	String	集群升级源版本
targetVersion	String	集群升级目标版本

表 4-303 status

参数	参数类型	描述
phase	String	状态，取值如下 ● Success 成功 ● Failed 失败 ● Error 错误

请求示例

集群升级后确认

```
POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/postcheck

{
  "kind" : "PostCheckTask",
  "apiVersion" : "v3",
  "spec" : {
    "clusterID" : "8978deaa-1743-11ee-8e46-0255ac10004c",
    "clusterVersion" : "v1.15.11-r1",
    "targetVersion" : "v1.19.16-r80"
  }
}
```

响应示例

状态码： 200

集群升级后确认成功。

```
{
  "kind" : "PostCheckTask",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "e99fedf8-348c-4084-b0fd-81bf187df4e0"
  },
  "spec" : {
```

```
    "clusterID" : "8978deaa-1743-11ee-8e46-0255ac10004c",
    "clusterVersion" : "v1.15.11-r1",
    "targetVersion" : "v1.19.16-r80"
  },
  "status" : {
    "phase" : "Success"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

集群升级后确认

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class CreateAutopilotPostCheckSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAutopilotPostCheckRequest request = new CreateAutopilotPostCheckRequest();
        request.withClusterId("{cluster_id}");
        PostcheckClusterRequestBody body = new PostcheckClusterRequestBody();
        PostcheckSpec specbody = new PostcheckSpec();
        specbody.withClusterID("8978deaa-1743-11ee-8e46-0255ac10004c")
            .withClusterVersion("v1.15.11-r1")
            .withTargetVersion("v1.19.16-r80");
        body.withSpec(specbody);
        body.withKind("PostCheckTask");
        body.withApiVersion("v3");
        request.withBody(body);
        try {
            CreateAutopilotPostCheckResponse response = client.createAutopilotPostCheck(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
```



```
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

集群升级后确认

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAutopilotPostCheckRequest()
        request.cluster_id = "{cluster_id}"
        specbody = PostcheckSpec(
            cluster_id="8978deaa-1743-11ee-8e46-0255ac10004c",
            cluster_version="v1.15.11-r1",
            target_version="v1.19.16-r80"
        )
        request.body = PostcheckClusterRequestBody(
            spec=specbody,
            kind="PostCheckTask",
            api_version="v3"
        )
        response = client.create_autopilot_post_check(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

集群升级后确认

```
package main

import (
```



```
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAutopilotPostCheckRequest{}
    request.ClusterId = "{cluster_id}"
    clusterIDSpec:= "8978deaa-1743-11ee-8e46-0255ac10004c"
    clusterVersionSpec:= "v1.15.11-r1"
    targetVersionSpec:= "v1.19.16-r80"
    specbody := &model.PostcheckSpec{
        ClusterID: &clusterIDSpec,
        ClusterVersion: &clusterVersionSpec,
        TargetVersion: &targetVersionSpec,
    }
    request.Body = &model.PostcheckClusterRequestBody{
        Spec: specbody,
        Kind: "PostCheckTask",
        ApiVersion: "v3",
    }
    response, err := client.CreateAutopilotPostCheck(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	集群升级后确认成功。

错误码

请参见**错误码**。

4.3.9 集群备份

功能介绍

集群备份

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3.1/projects/{project_id}/clusters/{cluster_id}/operation/snapshot

表 4-304 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

无

响应参数

状态码： 200

表 4-305 响应 Body 参数

参数	参数类型	描述
uid	String	任务ID
metadata	**SnapshotClusterResponseMetadata** object	备份任务元数据

表 4-306 SnapshotCluserResponseMetadata

参数	参数类型	描述
apiVersion	String	API版本，默认为v3.1
kind	String	任务类型

请求示例

集群升级备份请求示例

```
POST /autopilot/v3.1/projects/{project_id}/clusters/{cluster_id}/operation/snapshot
```

响应示例

状态码： 200

表示创建集群备份任务成功。

```
{
  "uid" : "15376f1b-daa6-4e2d-96a6-e9d5d7caeaa2",
  "metadata" : {
    "kind" : "Snapshot",
    "apiVersion" : "v3.1"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class CreateAutopilotClusterMasterSnapshotSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);
```

```
        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAutopilotClusterMasterSnapshotRequest request = new
CreateAutopilotClusterMasterSnapshotRequest();
        request.withClusterId("{cluster_id}");
        try {
            CreateAutopilotClusterMasterSnapshotResponse response =
client.createAutopilotClusterMasterSnapshot(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAutopilotClusterMasterSnapshotRequest()
        request.cluster_id = "{cluster_id}"
        response = client.create_autopilot_cluster_master_snapshot(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
```



```
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAutopilotClusterMasterSnapshotRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.CreateAutopilotClusterMasterSnapshot(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示创建集群备份任务成功。

错误码

请参见**错误码**。

4.3.10 获取集群备份任务详情列表

功能介绍

获取集群备份任务详情列表

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3.1/projects/{project_id}/clusters/{cluster_id}/operation/snapshot/tasks

表 4-307 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

无

响应参数

状态码： 200

表 4-308 响应 Body 参数

参数	参数类型	描述
apiVersion	String	api版本，默认为v3.1
kind	String	任务类型
metadata	**SnapshotTaskMetadata** object	备份任务元数据信息
items	Array of **SnapshotTask** objects	备份任务列表
status	**SnapshotTaskStatus** object	备份任务状态

表 4-309 SnapshotTask

参数	参数类型	描述
kind	String	任务类型
apiVersion	String	API版本
metadata	**SnapshotTaskMetadata** object	备份任务元数据信息
spec	**SnapshotSpec** object	备份任务配置信息（待废弃）
status	**SnapshotStatus** object	备份任务状态

表 4-310 SnapshotTaskMetadata

参数	参数类型	描述
uid	String	任务的ID。
creationTimestamp	String	任务的创建时间。
updateTimestamp	String	任务的更新时间。

表 4-311 SnapshotSpec

参数	参数类型	描述
items	Array of **SnapshotSpecItems** objects	备份任务详情

表 4-312 SnapshotSpecItems

参数	参数类型	描述
id	String	子任务ID
type	String	子任务类型
status	String	状态
creationTimestamp	String	任务创建时间

参数	参数类型	描述
updateTimestamp	String	任务更新时间
message	String	信息

表 4-313 SnapshotStatus

参数	参数类型	描述
phase	String	任务状态
progress	String	任务进度
completionTime	String	完成时间

表 4-314 SnapshotTaskStatus

参数	参数类型	描述
latestBackupTime	String	最近一次备份的时间

请求示例

无

响应示例

状态码： 200

表示获取集群备份任务详情列表成功。

```
{
  "kind" : "List",
  "apiVersion" : "v3.1",
  "metadata" : { },
  "items" : [ {
    "kind" : "SnapshotTask",
    "apiVersion" : "v3.1",
    "metadata" : {
      "uid" : "87d326f9-46b0-486e-a4ba-1f82ec9315ed",
      "creationTimestamp" : "2023-11-25 17:03:46.739012 +0800 CST",
      "updateTimestamp" : "2023-11-25 17:03:46.739027 +0800 CST"
    },
    "spec" : { },
    "status" : {
      "phase" : "Running",
      "progress" : "67",
      "completionTime" : "2023-11-25 17:03:46.739027 +0800 CST"
    }
  } ],
```

```
  "status" : {
    "latestBackupTime" : "2023-11-25 17:03:47.980844 +0800 CST"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotClusterMasterSnapshotTasksSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotClusterMasterSnapshotTasksRequest request = new ListAutopilotClusterMasterSnapshotTasksRequest();
        request.withClusterId("{cluster_id}");
        try {
            ListAutopilotClusterMasterSnapshotTasksResponse response = client.listAutopilotClusterMasterSnapshotTasks(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```


Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotClusterMasterSnapshotTasksRequest()
        request.cluster_id = "{cluster_id}"
        response = client.list_autopilot_cluster_master_snapshot_tasks(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
        WithCredential(auth).
        Build())

    request := &model.ListAutopilotClusterMasterSnapshotTasksRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ListAutopilotClusterMasterSnapshotTasks(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群备份任务详情列表成功。

错误码

请参见**错误码**。

4.3.11 获取集群升级相关信息

功能介绍

获取集群升级相关信息

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/upgradeinfo

表 4-315 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

无

响应参数

状态码： 200

表 4-316 响应 Body 参数

参数	参数类型	描述
kind	String	类型
apiVersion	String	API版本
metadata	**Metadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
spec	**UpgradeInfoSpec** object	升级配置相关信息
status	**UpgradeInfoStatus** object	升级状态信息

表 4-317 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-318 UpgradeInfoSpec

参数	参数类型	描述
lastUpgradeInfo	**UpgradeInfoStatus** object	上次集群升级信息

参数	参数类型	描述
versionInfo	**UpgradeVersionInfo** object	版本信息
upgradeFeatureGates	**UpgradeFeatureGates** object	集群升级特性开关

表 4-319 UpgradeVersionInfo

参数	参数类型	描述
release	String	正式版本号，如：v1.19.10
patch	String	补丁版本号，如r0
suggestPatch	String	推荐升级的目标补丁版本号，如r0
targetVersions	Array of strings	升级目标版本集合

表 4-320 UpgradeFeatureGates

参数	参数类型	描述
supportUpgradePageV4	Boolean	集群升级Console界面是否支持V4版本，该字段一般由CCE Console使用。

表 4-321 UpgradeInfoStatus

参数	参数类型	描述
phase	String	升级任务状态. **说明** Init：初始化 Running：运行中 Pause：暂停 Success：成功 Failed：失败
progress	String	升级任务进度
completionTime	String	升级任务结束时间

请求示例

无

响应示例

状态码： 200

表示获取集群升级相关信息成功。

```
{
  "kind" : "UpgradeInfo",
  "apiVersion" : "v3",
  "metadata" : { },
  "spec" : {
    "lastUpgradeInfo" : {
      "phase" : "Success",
      "completionTime" : "2023-11-25 11:18:54.478926 +0800 CST"
    },
    "versionInfo" : {
      "release" : "v1.27.2",
      "patch" : "r0",
      "suggestPatch" : "r0",
      "targetVersions" : [ "v1.27.3-r0" ]
    },
    "upgradeFeatureGates" : {
      "supportUpgradePageV4" : true
    }
  },
  "status" : {
    "phase" : "Success",
    "completionTime" : "2023-11-25 11:18:54.478926 +0800 CST"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotClusterUpgradeInfoSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
```

```
            .withProjectId(projectId)
            .withAk(ak)
            .withSk(sk);

        CceClient client = CceClient.newBuilder()
            .withCredential(auth)
            .withRegion(CceRegion.valueOf("<YOUR REGION>"))
            .build();
        ShowAutopilotClusterUpgradeInfoRequest request = new ShowAutopilotClusterUpgradeInfoRequest();
        request.withClusterId("{cluster_id}");
        try {
            ShowAutopilotClusterUpgradeInfoResponse response =
client.showAutopilotClusterUpgradeInfo(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotClusterUpgradeInfoRequest()
        request.cluster_id = "{cluster_id}"
        response = client.show_autopilot_cluster_upgrade_info(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```


Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotClusterUpgradeInfoRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ShowAutopilotClusterUpgradeInfo(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群升级相关信息成功。

错误码

请参见**错误码**。

4.3.12 获取集群升级路径

功能介绍

获取集群升级路径

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/clusterupgradepaths

请求参数

表 4-322 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-323 响应 Body 参数

参数	参数类型	描述
apiVersion	String	API版本
kind	String	资源类型
metadata	**Metadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
upgradePaths	Array of **UpgradePath** objects	升级路径集合

表 4-324 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-325 UpgradePath

参数	参数类型	描述
version	String	集群版本，v1.19及以下集群形如v1.19.16-r20，v1.21及以上形如v1.21,v1.23，详细请参考CCE集群版本号说明。
platformVersion	String	CCE集群平台版本号，表示集群版本(version)下的内部版本。用于跟踪某一集群版本内的迭代，集群版本内唯一，跨集群版本重新计数。platformVersion格式为：cce.X.Y- X: 表示内部特性版本。集群版本中特性或者补丁修复，或者OS支持等变更场景。其值从1开始单调递增。- Y: 表示内部特性版本的补丁版本。仅用于特性版本上线后的软件包更新，不涉及其他修改。其值从0开始单调递增。
targetVersions	Array of strings	可升级的目标版本集合

请求示例

无

响应示例

状态码： 200

表示获取集群升级路径信息成功。

```
{
  "kind" : "ClusterUpgradePaths",
  "apiVersion" : "v3",
```

```
 "metadata" : { },
 "upgradePaths" : [ {
   "version" : "v1.25",
   "platformVersion" : "cce.5.0",
   "targetVersions" : [ "v1.25.6-r0", "v1.27.3-r0" ]
 }, {
   "version" : "v1.25",
   "platformVersion" : "cce.4.0",
   "targetVersions" : [ "v1.25.6-r0", "v1.27.3-r0" ]
 }, {
   "version" : "v1.23",
   "platformVersion" : "cce.10.0",
   "targetVersions" : [ "v1.23.11-r0", "v1.25.6-r0", "v1.27.3-r0" ]
 }, {
   "version" : "v1.23",
   "platformVersion" : "cce.9.0",
   "targetVersions" : [ "v1.23.11-r0", "v1.25.6-r0", "v1.27.3-r0" ]
 } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotClusterUpgradePathsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotClusterUpgradePathsRequest request = new ListAutopilotClusterUpgradePathsRequest();
        try {
            ListAutopilotClusterUpgradePathsResponse response = client.listAutopilotClusterUpgradePaths(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
```



```
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotClusterUpgradePathsRequest()
        response = client.list_autopilot_cluster_upgrade_paths(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()
```

```
    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListAutopilotClusterUpgradePathsRequest{}
    response, err := client.ListAutopilotClusterUpgradePaths(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群升级路径信息成功。

错误码

请参见**错误码**。

4.3.13 获取集群升级特性开关配置

功能介绍

获取集群升级特性开关配置

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/clusterupgradefeaturegates

请求参数

表 4-326 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-327 响应 Body 参数

参数	参数类型	描述
apiVersion	String	API版本
kind	String	资源类型
metadata	**Metadata** object	基本信息，为集合类的元素类型，包含一组由不同名称定义的属性
upgradeFeatureGates	Map<String,String>	特性开关信息,格式为key/value键值对。 • Key: 目前有下列值：DisplayPreCheckDetail(展示所有集群升级前检查项详情),EvsSnapshot(使用EVS快照备份集群), LabelForSkippedNode(支持为集群升级过程中跳过的节点打标签), UpgradeStrategy(集群升级策略) • Value: Support 支持,Disable 关闭,Default 使用CCE服务默认规则判断

表 4-328 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间

参数	参数类型	描述
creationTimestamp	String	创建时间

请求示例

无

响应示例

状态码： 200

表示获取集群升级路径信息成功。

```
{
  "kind" : "ClusterUpgradeFeatureGates",
  "apiVersion" : "v3",
  "metadata" : { },
  "upgradeFeatureGates" : {
    "DisplayPreCheckDetail" : "Support",
    "EvsSnapshot" : "Support",
    "LabelForSkippedNode" : "Support",
    "UpgradeStrategy" : "Support"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotClusterUpgradeFeatureGatesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
```

```
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotClusterUpgradeFeatureGatesRequest request = new
ListAutopilotClusterUpgradeFeatureGatesRequest();
        try {
            ListAutopilotClusterUpgradeFeatureGatesResponse response =
client.listAutopilotClusterUpgradeFeatureGates(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotClusterUpgradeFeatureGatesRequest()
        response = client.list_autopilot_cluster_upgrade_feature_gates(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)
```



```
func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListAutopilotClusterUpgradeFeatureGatesRequest{}
    response, err := client.ListAutopilotClusterUpgradeFeatureGates(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取集群升级路径信息成功。

错误码

请参见**错误码**。

4.3.14 开启集群升级流程引导任务

功能介绍

该API用于创建一个集群升级流程引导任务。请在调用本接口完成引导任务创建之后，通过集群升级前检查开始检查任务。

升级流程任务用于控制集群升级任务的执行流程，执行流程为 升级前检查 => 集群升级 => 升级后检查。

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgradeworkflows

表 4-329 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-330 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-331 请求 Body 参数

参数	是否必选	参数类型	描述
kind	是	String	API类型，固定值“WorkFlowTask”，该值不可修改。
apiVersion	是	String	API版本，固定值“v3”，该值不可修改。
spec	是	**WorkFlowSpec** object	集合类的元素类型，您对集群升级流程主体都在spec中给出。CCE通过spec的描述来创建或更新对象。

表 4-332 WorkFlowSpec

参数	是否必选	参数类型	描述
clusterID	否	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效
clusterVersion	否	String	本次集群升级的当前版本
targetVersion	是	String	本次集群升级的目标版本

响应参数

状态码： 201

表 4-333 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“WorkFlowTask”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**Metadata** object	升级流程的元数据信息
spec	**WorkFlowSpec** object	集合类的元素类型，您对集群升级流程主体都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	**WorkFlowStatus** object	集合类的元素类型，用于记录本次集群升级流程的当前状态信息，包含了集群升级流程的各个流程的执行状态

表 4-334 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间

参数	参数类型	描述
creationTimestamp	String	创建时间

表 4-335 WorkFlowSpec

参数	参数类型	描述
clusterID	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效
clusterVersion	String	本次集群升级的当前版本
targetVersion	String	本次集群升级的目标版本

表 4-336 WorkFlowStatus

参数	参数类型	描述
phase	String	集群升级流程的执行状态： Init: 表示该升级流程中还未有任何任务开始运行 Running: 表示该升级流程中已有任务开始执行 Pending: 表示该升级流程中有任务执行失败 Success: 表示该升级流程中所有任务都已执行成功 Cancel: 表示该升级流程已被取消
pointStatuses	Array of **PointStatus** objects	升级流程中的各个任务项的执行状态
lineStatuses	Array of **LineStatus** objects	表示该升级流程的任务执行线路

表 4-337 PointStatus

参数	参数类型	描述
taskType	String	集群升级任务类型： Cluster: 集群升级任务 PreCheck: 集群升级预检查任务 Rollback: 集群升级回归任务 Snapshot: 集群升级快照任务 PostCheck: 集群升级后检查任务

参数	参数类型	描述
taskID	String	升级任务项ID
status	String	集群升级状态： Init: 任务初始状态 Queuing: 任务已进入执行队列 Running: 任务开始执行 Success: 任务执行成功 Failed: 任务执行失败
startTimeStamp	String	升级任务开始时间
endTimeStamp	String	升级任务结束时间
expireTimeStamp	String	升级任务过期时间（当前仅升级前检查任务适用）

表 4-338 LineStatus

参数	参数类型	描述
startPoint	**Point** object	线路起点
endPoint	**Point** object	线路终点
critical	String	表示是否为关键线路（关键线路未执行无法取消升级流程）

表 4-339 Point

参数	参数类型	描述
taskType	String	集群升级任务类型

请求示例

开启升级集群至v1.28版本的流程

```
POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgradeworkflows

{
  "kind" : "WorkFlowTask",
  "apiVersion" : "v3",
  "spec" : {
    "targetVersion" : "v1.23"
  }
}
```

响应示例

状态码： 201

表示在指定集群下创建升级流程成功。

```
{
  "kind" : "WorkFlowTask",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "5ddfddfe-87db-11ec-b5e5-0255ac111914"
  },
  "spec" : {
    "clusterID" : "b4b9e60f-8aa2-11ee-af09-0255ac10004f",
    "clusterVersion" : "v1.17.17-r0",
    "targetVersion" : "v1.19.16-r80"
  },
  "status" : {
    "pointStatuses" : [ {
      "taskType" : "PreCheck"
    }, {
      "taskType" : "Snapshot"
    }, {
      "taskType" : "Cluster"
    }, {
      "taskType" : "PostCheck"
    } ],
    "lineStatuses" : [ {
      "startPoint" : {
        "taskType" : "PreCheck"
      },
      "endPoint" : {
        "taskType" : "Cluster"
      }
    }, {
      "startPoint" : {
        "taskType" : "Cluster"
      },
      "endPoint" : {
        "taskType" : "PostCheck"
      }
    } ]
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

开启升级集群至v1.28版本的流程

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class CreateAutopilotUpgradeWorkFlowSolution {
```

```
    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        CreateAutopilotUpgradeWorkFlowRequest request = new CreateAutopilotUpgradeWorkFlowRequest();
        request.withClusterId("{cluster_id}");
        CreateUpgradeWorkFlowRequestBody body = new CreateUpgradeWorkFlowRequestBody();
        WorkFlowSpec specbody = new WorkFlowSpec();
        specbody.withTargetVersion("v1.23");
        body.withSpec(specbody);
        body.withApiVersion("v3");
        body.withKind("WorkFlowTask");
        request.withBody(body);
        try {
            CreateAutopilotUpgradeWorkFlowResponse response = client.createAutopilotUpgradeWorkFlow(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

开启升级集群至v1.28版本的流程

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)
```



```
client = CceClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(CceRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = CreateAutopilotUpgradeWorkFlowRequest()
    request.cluster_id = "{cluster_id}"
    specbody = WorkFlowSpec(
        target_version="v1.23"
    )
    request.body = CreateUpgradeWorkFlowRequestBody(
        spec=specbody,
        api_version="v3",
        kind="WorkFlowTask"
    )
    response = client.create_autopilot_upgrade_work_flow(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

开启升级集群至v1.28版本的流程

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAutopilotUpgradeWorkFlowRequest{}
    request.ClusterId = "{cluster_id}"
    specbody := &model.WorkFlowSpec{
        TargetVersion: "v1.23",
    }
    request.Body = &model.CreateUpgradeWorkFlowRequestBody{
        Spec: specbody,
        ApiVersion: "v3",
```

```
        Kind: "WorkFlowTask",
    }
    response, err := client.CreateAutopilotUpgradeWorkFlow(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	表示在指定集群下创建升级流程成功。

错误码

请参见**错误码**。

4.3.15 获取 UpgradeWorkFlows 列表

功能介绍

获取历史集群升级引导任务列表

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgradeworkflows

表 4-340 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-341 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-342 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“List”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
items	**UpgradeWorkFlow** object	升级工作流列表

表 4-343 UpgradeWorkFlow

参数	参数类型	描述
kind	String	API类型，固定值“WorkFlowTask”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**Metadata** object	升级流程的元数据信息
spec	**WorkFlowSpec** object	集合类的元素类型，您对集群升级流程主体都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	**WorkFlowStatus** object	集合类的元素类型，用于记录本次集群升级流程的当前状态信息，包含了集群升级流程的各个流程的执行状态

表 4-344 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-345 WorkFlowSpec

参数	参数类型	描述
clusterID	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效
clusterVersion	String	本次集群升级的当前版本
targetVersion	String	本次集群升级的目标版本

表 4-346 WorkFlowStatus

参数	参数类型	描述
phase	String	集群升级流程的执行状态： Init: 表示该升级流程中还未有任何任务开始运行 Running: 表示该升级流程中已有任务开始执行 Pending: 表示该升级流程中有任务执行失败 Success: 表示该升级流程中所有任务都已执行成功 Cancel: 表示该升级流程已被取消
pointStatuses	Array of **PointStatus** objects	升级流程中的各个任务项的执行状态
lineStatuses	Array of **LineStatus** objects	表示该升级流程的任务执行线路

表 4-347 PointStatus

参数	参数类型	描述
taskType	String	集群升级任务类型： Cluster: 集群升级任务 PreCheck: 集群升级预检查任务 Rollback: 集群升级回归任务 Snapshot: 集群升级快照任务 PostCheck: 集群升级后检查任务
taskID	String	升级任务项ID
status	String	集群升级状态： Init: 任务初始状态 Queuing: 任务已进入执行队列 Running: 任务开始执行 Success: 任务执行成功 Failed: 任务执行失败
startTimeStamp	String	升级任务开始时间
endTimeStamp	String	升级任务结束时间
expireTimeStamp	String	升级任务过期时间（当前仅升级前检查任务适用）

表 4-348 LineStatus

参数	参数类型	描述
startPoint	**Point** object	线路起点
endPoint	**Point** object	线路终点
critical	String	表示是否为关键线路（关键线路未执行无法取消升级流程）

表 4-349 Point

参数	参数类型	描述
taskType	String	集群升级任务类型

请求示例

无

响应示例

状态码： 200

获取历史集群升级引导任务列表成功。


```
{
  "apiVersion" : "v3",
  "kind" : "List",
  "items" : {
    "kind" : "WorkFlowTask",
    "apiVersion" : "v3",
    "metadata" : {
      "uid" : "730f5577-38ef-448c-b4a7-c6878fbefdda",
      "creationTimestamp" : "2023-11-24 08:39:15.894417 +0000 UTC",
      "updateTimestamp" : "2023-11-25 02:57:25.718567 +0000 UTC"
    },
    "spec" : {
      "clusterID" : "b4b9e60f-8aa2-11ee-af09-0255ac10004f",
      "clusterVersion" : "v1.17.17-r0",
      "targetVersion" : "v1.19.16-r80"
    },
    "status" : {
      "phase" : "Cancel",
      "pointStatuses" : [ {
        "taskType" : "PreCheck"
      }, {
        "taskType" : "Snapshot"
      }, {
        "taskType" : "Cluster"
      }, {
        "taskType" : "PostCheck"
      } ],
      "lineStatuses" : [ {
        "startPoint" : {
          "taskType" : "PreCheck"
        },
        "endPoint" : {
          "taskType" : "Cluster"
        }
      }, {
        "startPoint" : {
          "taskType" : "Cluster"
        },
        "endPoint" : {
          "taskType" : "PostCheck"
        }
      } ]
    }
  }
}
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
```

```
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotUpgradeWorkFlowsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotUpgradeWorkFlowsRequest request = new ListAutopilotUpgradeWorkFlowsRequest();
        request.withClusterId("{cluster_id}");
        try {
            ListAutopilotUpgradeWorkFlowsResponse response = client.listAutopilotUpgradeWorkFlows(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"
```



```
    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotUpgradeWorkFlowsRequest()
        request.cluster_id = "{cluster_id}"
        response = client.list_autopilot_upgrade_work_flows(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListAutopilotUpgradeWorkFlowsRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ListAutopilotUpgradeWorkFlows(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	获取历史集群升级引导任务列表成功。

错误码

请参见**错误码**。

4.3.16 获取指定集群升级引导任务详情

功能介绍

该API用于通过升级引导任务ID获取任务的详细信息。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgradeworkflows/{upgrade_workflow_id}

表 4-350 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。
upgrade_workflow_id	是	String	集群升级任务引导流程ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-351 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-352 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“WorkFlowTask”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**Metadata** object	升级流程的元数据信息
spec	**WorkFlowSpec** object	集合类的元素类型，您对集群升级流程主体都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	**WorkFlowStatus** object	集合类的元素类型，用于记录本次集群升级流程的当前状态信息，包含了集群升级流程的各个流程的执行状态

表 4-353 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-354 WorkFlowSpec

参数	参数类型	描述
clusterID	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效
clusterVersion	String	本次集群升级的当前版本
targetVersion	String	本次集群升级的目标版本

表 4-355 WorkFlowStatus

参数	参数类型	描述
phase	String	集群升级流程的执行状态： Init: 表示该升级流程中还未有任何任务开始运行 Running: 表示该升级流程中已有任务开始执行 Pending: 表示该升级流程中有任务执行失败 Success: 表示该升级流程中所有任务都已执行成功 Cancel: 表示该升级流程已被取消
pointStatuses	Array of **PointStatus** objects	升级流程中的各个任务项的执行状态
lineStatuses	Array of **LineStatus** objects	表示该升级流程的任务执行线路

表 4-356 PointStatus

参数	参数类型	描述
taskType	String	集群升级任务类型： Cluster: 集群升级任务 PreCheck: 集群升级预检查任务 Rollback: 集群升级回归任务 Snapshot: 集群升级快照任务 PostCheck: 集群升级后检查任务
taskID	String	升级任务项ID

参数	参数类型	描述
status	String	集群升级状态： Init: 任务初始状态 Queuing: 任务已进入执行队列 Running: 任务开始执行 Success: 任务执行成功 Failed: 任务执行失败
startTimeStamp	String	升级任务开始时间
endTimeStamp	String	升级任务结束时间
expireTimeStamp	String	升级任务过期时间（当前仅升级前检查任务适用）

表 4-357 LineStatus

参数	参数类型	描述
startPoint	**Point** object	线路起点
endPoint	**Point** object	线路终点
critical	String	表示是否为关键线路（关键线路未执行无法取消升级流程）

表 4-358 Point

参数	参数类型	描述
taskType	String	集群升级任务类型

请求示例

无

响应示例

状态码： 200

表示获取指定集群升级引导任务详情成功

```
{
  "kind" : "WorkFlowTask",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "c271e39e-1a6e-4d3d-8fa8-2a36329c68d1",
```

```
    "creationTimestamp" : "2023-11-25 06:32:34.923248 +0000 UTC",
    "updateTimestamp" : "2023-11-25 07:49:30.281911 +0000 UTC"
  },
  "spec" : {
    "clusterID" : "b4b9e60f-8aa2-11ee-af09-0255ac10004f",
    "clusterVersion" : "v1.17.17-r0",
    "targetVersion" : "v1.19.16-r80"
  },
  "status" : {
    "phase" : "Pending",
    "pointStatuses" : [ {
      "taskType" : "PreCheck",
      "taskID" : "f61e008c-1600-41c0-9bde-121de5a30660",
      "status" : "Success",
      "startTimeStamp" : "2023-11-25 07:20:04.592972 +0000 UTC",
      "endTimeStamp" : "2023-11-25 07:21:05.518966 +0000 UTC",
      "expireTimeStamp" : "2023-11-25 08:21:05.518966 +0000 UTC"
    }, {
      "taskType" : "Snapshot"
    }, {
      "taskType" : "Cluster",
      "taskID" : "6d799ff6-3afe-4242-80b4-6f0a0fa746cb",
      "status" : "Failed",
      "startTimeStamp" : "2023-11-25 07:49:30.283459 +0000 UTC",
      "endTimeStamp" : "2023-11-25 07:58:35.507243 +0000 UTC"
    }, {
      "taskType" : "PostCheck"
    } ],
    "lineStatuses" : [ {
      "startPoint" : {
        "taskType" : "PreCheck"
      },
      "endPoint" : {
        "taskType" : "Cluster"
      }
    }, {
      "startPoint" : {
        "taskType" : "Cluster"
      },
      "endPoint" : {
        "taskType" : "PostCheck"
      }
    } ]
  }
}
```


状态码

状态码	描述
200	表示获取指定集群升级引导任务详情成功

错误码

请参见**错误码**。

4.3.17 更新指定集群升级引导任务状态

功能介绍

该API用于更新指定集群升级引导任务状态，当前仅适用于取消升级流程

调用该API时升级流程引导任务状态不能为进行中(running) 已完成(success) 已取消(cancel),升级子任务状态不能为running(进行中) init(已初始化) pause(任务被暂停) queue(队列中)

调用方法

请参见**如何调用API**。

URI

PATCH /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/operation/upgradeworkflows/{upgrade_workflow_id}

表 4-359 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。
upgrade_workflow_id	是	String	集群升级任务引导流程ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-360 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-361 请求 Body 参数

参数	是否必选	参数类型	描述
status	否	**status** object	更新后workflow的状态（当前仅支持Cancel）

表 4-362 status

参数	是否必选	参数类型	描述
phase	否	String	集群升级流程的执行状态： Init: 表示该升级流程中还未有任何任务开始运行 Running: 表示该升级流程中已有任务开始执行 Pending: 表示该升级流程中有任务执行失败 Success: 表示该升级流程中所有任务都已执行成功 Cancel: 表示该升级流程已被取消

响应参数

状态码： 200

表 4-363 响应 Body 参数

参数	参数类型	描述
kind	String	API类型，固定值“WorkFlowTask”，该值不可修改。
apiVersion	String	API版本，固定值“v3”，该值不可修改。
metadata	**Metadata** object	升级流程的元数据信息
spec	**WorkFlowSpec** object	集合类的元素类型，您对集群升级流程主体都在spec中给出。CCE通过spec的描述来创建或更新对象。
status	**WorkFlowStatus** object	集合类的元素类型，用于记录本次集群升级流程的当前状态信息，包含了集群升级流程的各个流程的执行状态

表 4-364 Metadata

参数	参数类型	描述
uid	String	唯一id标识
name	String	资源名称
labels	Map<String,String>	资源标签，key/value对格式，接口保留字段，填写不会生效

参数	参数类型	描述
annotations	Map<String,String>	资源注解，由key/value组成
updateTimestamp	String	更新时间
creationTimestamp	String	创建时间

表 4-365 WorkFlowSpec

参数	参数类型	描述
clusterID	String	集群ID，资源唯一标识，创建成功后自动生成，填写无效
clusterVersion	String	本次集群升级的当前版本
targetVersion	String	本次集群升级的目标版本

表 4-366 WorkFlowStatus

参数	参数类型	描述
phase	String	集群升级流程的执行状态： Init: 表示该升级流程中还未有任何任务开始运行 Running: 表示该升级流程中已有任务开始执行 Pending: 表示该升级流程中有任务执行失败 Success: 表示该升级流程中所有任务都已执行成功 Cancel: 表示该升级流程已被取消
pointStatuses	Array of **PointStatus** objects	升级流程中的各个任务项的执行状态
lineStatuses	Array of **LineStatus** objects	表示该升级流程的任务执行线路

表 4-367 PointStatus

参数	参数类型	描述
taskType	String	集群升级任务类型： Cluster: 集群升级任务 PreCheck: 集群升级预检查任务 Rollback: 集群升级回归任务 Snapshot: 集群升级快照任务 PostCheck: 集群升级后检查任务
taskID	String	升级任务项ID
status	String	集群升级状态： Init: 任务初始状态 Queuing: 任务已进入执行队列 Running: 任务开始执行 Success: 任务执行成功 Failed: 任务执行失败
startTimeStamp	String	升级任务开始时间
endTimeStamp	String	升级任务结束时间
expireTimeStamp	String	升级任务过期时间（当前仅升级前检查任务适用）

表 4-368 LineStatus

参数	参数类型	描述
startPoint	**Point** object	线路起点
endPoint	**Point** object	线路终点
critical	String	表示是否为关键线路（关键线路未执行无法取消升级流程）

表 4-369 Point

参数	参数类型	描述
taskType	String	集群升级任务类型

请求示例

取消升级流程

```
PATCH /autopilot/v3/projects/47eb1d64cbeb45cfa01ae20af4f4b563/clusters/
f9960c6b-8e60-11ee-9754-0255ac100b05/operation/upgradeworkflows/
d0b7e319-8172-424c-86ea-543cd23f9756

{
  "status" : {
    "phase" : "Cancel"
  }
}
```

响应示例

状态码： 200

表示更新集群升级引导任务状态成功

```
{
  "kind" : "WorkFlowTask",
  "apiVersion" : "v3",
  "metadata" : {
    "uid" : "c271e39e-1a6e-4d3d-8fa8-2a36329c68d1",
    "creationTimestamp" : "2023-11-25 06:32:34.923248 +0000 UTC",
    "updateTimestamp" : "2023-11-25 07:49:30.281911 +0000 UTC"
  },
  "spec" : {
    "clusterID" : "b4b9e60f-8aa2-11ee-af09-0255ac10004f",
    "clusterVersion" : "v1.17.17-r0",
    "targetVersion" : "v1.19.16-r80"
  },
  "status" : {
    "phase" : "Cancel",
    "pointStatuses" : [ {
      "taskType" : "PreCheck",
      "taskID" : "f61e008c-1600-41c0-9bde-121de5a30660",
      "status" : "Success",
      "startTimeStamp" : "2023-11-25 07:20:04.592972 +0000 UTC",
      "endTimeStamp" : "2023-11-25 07:21:05.518966 +0000 UTC",
      "expireTimeStamp" : "2023-11-25 08:21:05.518966 +0000 UTC"
    }, {
      "taskType" : "Snapshot"
    }, {
      "taskType" : "Cluster",
      "taskID" : "6d799ff6-3afe-4242-80b4-6f0a0fa746cb",
      "status" : "Failed",
      "startTimeStamp" : "2023-11-25 07:49:30.283459 +0000 UTC",
      "endTimeStamp" : "2023-11-25 07:58:35.507243 +0000 UTC"
    }, {
      "taskType" : "PostCheck"
    } ],
    "lineStatuses" : [ {
      "startPoint" : {
        "taskType" : "PreCheck"
      },
      "endPoint" : {
        "taskType" : "Cluster"
      }
    }, {
      "startPoint" : {
        "taskType" : "Cluster"
      },
      "endPoint" : {
        "taskType" : "PostCheck"
      }
    } ]
```

```
 }
}
```

SDK 代码示例

SDK代码示例如下。

Java

取消升级流程

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class UpgradeAutopilotWorkFlowUpdateSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        UpgradeAutopilotWorkFlowUpdateRequest request = new UpgradeAutopilotWorkFlowUpdateRequest();
        request.withClusterId("{cluster_id}");
        request.withUpgradeWorkflowId("{upgrade_workflow_id}");
        UpgradeWorkFlowUpdateRequestBody body = new UpgradeWorkFlowUpdateRequestBody();
        UpgradeWorkFlowUpdateRequestBodyStatus statusbody = new UpgradeWorkFlowUpdateRequestBodyStatus();
        statusbody.withPhase(UpgradeWorkFlowUpdateRequestBodyStatus.PhaseEnum.fromValue("Cancel"));
        body.withStatus(statusbody);
        request.withBody(body);
        try {
            UpgradeAutopilotWorkFlowUpdateResponse response = client.upgradeAutopilotWorkFlowUpdate(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
```

```
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

取消升级流程

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpgradeAutopilotWorkFlowUpdateRequest()
        request.cluster_id = "{cluster_id}"
        request.upgrade_workflow_id = "{upgrade_workflow_id}"
        statusbody = UpgradeWorkFlowUpdateRequestBodyStatus(
            phase="Cancel"
        )
        request.body = UpgradeWorkFlowUpdateRequestBody(
            status=statusbody
        )
        response = client.upgrade_autopilot_work_flow_update(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

取消升级流程

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
```

```
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpgradeAutopilotWorkFlowUpdateRequest{}
    request.ClusterId = "{cluster_id}"
    request.UpgradeWorkflowId = "{upgrade_workflow_id}"
    phaseStatus:= model.GetUpgradeWorkFlowUpdateRequestBodyStatusPhaseEnum().CANCEL
    statusbody := &model.UpgradeWorkFlowUpdateRequestBodyStatus{
        Phase: &phaseStatus,
    }
    request.Body = &model.UpgradeWorkFlowUpdateRequestBody{
        Status: statusbody,
    }
    response, err := client.UpgradeAutopilotWorkFlowUpdate(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示更新集群升级引导任务状态成功

错误码

请参见**错误码**。

4.4 配额管理（Autopilot）

4.4.1 查询 CCE 服务下的资源配额

功能介绍

该API用于查询CCE服务下的资源配额。

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v3/projects/{project_id}/quotas

表 4-370 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-371 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-372 响应 Body 参数

参数	参数类型	描述
quotas	Array of **QuotaResource** objects	资源

表 4-373 QuotaResource

参数	参数类型	描述
quotaKey	String	资源类型
quotaLimit	Integer	配额值
used	Integer	已创建的资源个数
unit	String	单位
regionId	String	局点ID。若资源不涉及此参数，则不返回该参数。
availabilityZoneId	String	可用区ID。若资源不涉及此参数，则不返回该参数。

请求示例

无

响应示例

状态码： 200

表示获取资源配额成功。

```
{
  "quotas" : [ {
    "quotaKey" : "autopilot_cluster",
    "quotaLimit" : 20,
    "used" : 13,
    "unit" : "count",
    "regionId" : "cn-north-7"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotQuotasSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
```

```
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotQuotasRequest request = new ShowAutopilotQuotasRequest();
        try {
            ShowAutopilotQuotasResponse response = client.showAutopilotQuotas(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotQuotasRequest()
        response = client.show_autopilot_quotas(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
```



```
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotQuotasRequest{}
    response, err := client.ShowAutopilotQuotas(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	表示获取资源配额成功。

错误码

请参见**错误码**。

4.5 标签管理（Autopilot）

4.5.1 批量添加指定集群的资源标签

功能介绍

该API用于批量添加指定集群的资源标签。

说明

- 每个集群支持最多20个资源标签。
- 此接口为幂等接口：创建时，如果创建的标签已经存在（key/value均相同视为重复），默认处理成功；key相同，value不同时会覆盖原有标签。

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/tags/create

表 4-374 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-375 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-376 请求 Body 参数

参数	是否必选	参数类型	描述
tags	是	Array of **ResourceTag** objects	待创建的集群资源标签列表。单集群资源标签总数上限为20。

表 4-377 ResourceTag

参数	是否必选	参数类型	描述
key	否	String	Key值。 • 不能为空且首尾不能包含空格，最多支持128个字符 • 可用UTF-8格式表示的汉字、字母、数字和空格 • 支持部分特殊字符：_.:=+-@ • 不能以"_sys_"开头
value	否	String	Value值。 • 可以为空但不能缺省，最多支持255个字符 • 可用UTF-8格式表示的汉字、字母、数字和空格 • 支持部分特殊字符：_.:/=+-@

响应参数

无

请求示例

批量添加指定集群的资源标签

```
POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/tags/create

{
  "tags" : [ {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key2",
    "value" : "value3"
  } ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

批量添加指定集群的资源标签

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;

import java.util.List;
import java.util.ArrayList;

public class BatchCreateAutopilotClusterTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        BatchCreateAutopilotClusterTagsRequest request = new BatchCreateAutopilotClusterTagsRequest();
        request.withClusterId("{cluster_id}");
        BatchCreateClusterTagsRequestBody body = new BatchCreateClusterTagsRequestBody();
        List<ResourceTag> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new ResourceTag()
                .withKey("key1")
                .withValue("value1")
        );
        listbodyTags.add(
            new ResourceTag()
                .withKey("key2")
                .withValue("value3")
        );
        body.withTags(listbodyTags);
        request.withBody(body);
        try {
            BatchCreateAutopilotClusterTagsResponse response = client.batchCreateAutopilotClusterTags(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
```

```
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

批量添加指定集群的资源标签

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchCreateAutopilotClusterTagsRequest()
        request.cluster_id = "{cluster_id}"
        listTagsbody = [
            ResourceTag(
                key="key1",
                value="value1"
            ),
            ResourceTag(
                key="key2",
                value="value3"
            )
        ]
        request.body = BatchCreateClusterTagsRequestBody(
            tags=listTagsbody
        )
        response = client.batch_create_autopilot_cluster_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

批量添加指定集群的资源标签

```
package main
```



```
import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchCreateAutopilotClusterTagsRequest{}
    request.ClusterId = "{cluster_id}"
    keyTags:= "key1"
    valueTags:= "value1"
    keyTags1:= "key2"
    valueTags1:= "value3"
    var listTagsbody = []model.ResourceTag{
        {
            Key: &keyTags,
            Value: &valueTags,
        },
        {
            Key: &keyTags1,
            Value: &valueTags1,
        },
    }
    request.Body = &model.BatchCreateClusterTagsRequestBody{
        Tags: listTagsbody,
    }
    response, err := client.BatchCreateAutopilotClusterTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见**错误码**。

4.5.2 批量删除指定集群的资源标签

功能介绍

该API用于批量删除指定集群的资源标签。

说明

- 此接口为幂等接口：删除时，如果删除的标签key不存在，默认处理成功。

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/tags/delete

表 4-378 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-379 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-380 请求 Body 参数

参数	是否必选	参数类型	描述
tags	是	Array of **ResourceDeleteTag** objects	待删除的集群资源标签列表。

表 4-381 ResourceDeleteTag

参数	是否必选	参数类型	描述
key	否	String	Key值。 • 不能为空，最多支持128个字符 • 可用UTF-8格式表示的汉字、字母、数字和空格 • 支持部分特殊字符：_.:/=+-@ • 不能以"_sys_"开头

响应参数

无

请求示例

批量删除指定集群的资源标签

```
POST /autopilot/v3/projects/{project_id}/clusters/{cluster_id}/tags/delete

{
  "tags" : [ {
    "key" : "key1"
  }, {
    "key" : "key2"
  } ]
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

批量删除指定集群的资源标签

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;

import java.util.List;
import java.util.ArrayList;

public class BatchDeleteAutopilotClusterTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        BatchDeleteAutopilotClusterTagsRequest request = new BatchDeleteAutopilotClusterTagsRequest();
        request.withClusterId("{cluster_id}");
        BatchDeleteClusterTagsRequestBody body = new BatchDeleteClusterTagsRequestBody();
        List<ResourceDeleteTag> listbodyTags = new ArrayList<>();
        listbodyTags.add(
            new ResourceDeleteTag()
                .withKey("key1")
        );
        listbodyTags.add(
            new ResourceDeleteTag()
                .withKey("key2")
        );
        body.withTags(listbodyTags);
        request.withBody(body);
        try {
            BatchDeleteAutopilotClusterTagsResponse response = client.batchDeleteAutopilotClusterTags(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
```

```
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

批量删除指定集群的资源标签

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchDeleteAutopilotClusterTagsRequest()
        request.cluster_id = "{cluster_id}"
        listTagsbody = [
            ResourceDeleteTag(
                key="key1"
            ),
            ResourceDeleteTag(
                key="key2"
            )
        ]
        request.body = BatchDeleteClusterTagsRequestBody(
            tags=listTagsbody
        )
        response = client.batch_delete_autopilot_cluster_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

批量删除指定集群的资源标签

```
package main
```

```
import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchDeleteAutopilotClusterTagsRequest{}
    request.ClusterId = "{cluster_id}"
    keyTags:= "key1"
    keyTags1:= "key2"
    var listTagsbody = []model.ResourceDeleteTag{
        {
            Key: &keyTags,
        },
        {
            Key: &keyTags1,
        },
    }
    request.Body = &model.BatchDeleteClusterTagsRequestBody{
        Tags: listTagsbody,
    }
    response, err := client.BatchDeleteAutopilotClusterTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见**错误码**。

4.6 模板管理（Autopilot）

4.6.1 上传模板

功能介绍

上传模板

调用方法

请参见**如何调用API**。

URI

POST /autopilot/v2/charts

请求参数

表 4-382 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-383 FormData 参数

参数	是否必选	参数类型	描述
parameters	否	String	上传模板的配置参数，示例如下： "{"override":true,"skip_lint":true,"source":"package"}" • skip_lint: 是否验证上传的模板 • override: 是否覆盖已存在的模板 • visible: 模板是否可见

参数	是否必选	参数类型	描述
content	是	File	模板包文件

响应参数

状态码： 201

表 4-384 响应 Body 参数

参数	参数类型	描述
id	String	模板ID
name	String	模板名称
values	String	模板值
translate	String	模板翻译资源
instruction	String	模板介绍
version	String	模板版本
description	String	模板描述
source	String	模板的来源
icon_url	String	模板的图标链接
public	Boolean	是否公开模板
chart_url	String	模板的链接
create_at	String	创建时间
update_at	String	更新时间

请求示例

```
POST /autopilot/v2/charts

{
  "parameters" : "{\"override\":true,\"skip_lint\":true,\"source\":\"package\"}",
  "content" : "chart-file.tgz"
}
```

响应示例

状态码： 201

Created

```
{
  "id" : "e99a7e86-afdd-11eb-aca3-0255ac100b0e",
  "name" : "neo4j",
  "values" : "{\"acceptLicenseAgreement\":\"no\",\"affinity\":{},\"authEnabled\":true,\"clusterDomain
```

\":\"cluster.local\",\"core\":{\"initContainers\":[],\"numberOfServers\":3,\"persistentVolume\":{\"enabled
\":true,\"mountPath\":\"/data\",\"size\":\"10Gi\"},\"sidecarContainers\":[]},\"defaultDatabase\":\"neo4j
\",\"image\":\"neo4j\",\"imagePullPolicy\":\"IfNotPresent\",\"imageTag\":\"4.0.3-enterprise\",\"name
\":\"neo4j\",\"nodeSelector\":{},\"podDisruptionBudget\":{},\"readReplica\":{\"autoscaling\":{\"enabled
\":false,\"maxReplicas\":3,\"minReplicas\":1,\"targetAverageUtilization\":70},\"initContainers\":
[],\"numberOfServers\":0,\"resources\":{},\"sidecarContainers\":[]},\"resources\":{},\"testImage
\":\"markhneedham/k8s-kubectl\",\"testImageTag\":\"master\",\"tolerations\":[],\"useAPOC\":\"true\"}",
  "translate" : "",
  "instruction" : "README.md",
  "version" : "3.0.1",
  "description" : "DEPRECATED Neo4j is the world's leading graph database",
  "source" : "",
  "icon_url" : "https://info.neo4j.com/rs/773-GON-065/images/neo4j_logo.png",
  "public" : false,
  "chart_url" : "neo4j-3.0.1.tgz",
  "create_at" : "2021-05-08T08:53:13Z",
  "update_at" : "2021-05-08T08:53:13Z"
}

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class UploadAutopilotChartSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        UploadAutopilotChartRequest request = new UploadAutopilotChartRequest();
        UploadAutopilotChartRequestBody bodybody = new UploadAutopilotChartRequestBody();
        bodybody.withParameters("{"override":true,"skip_lint":true,"source":"package"}")
            .withContent("chart-file.tgz");
        body.withBody(bodybody);
        request.withBody(listbodyBody);
        try {
            UploadAutopilotChartResponse response = client.uploadAutopilotChart(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
```



```
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UploadAutopilotChartRequest()
        bodybody = UploadAutopilotChartRequestBody(
            parameters="{"override":true,"skip_lint":true,"source":"package"}",
            content="chart-file.tgz"
        )
        request.body = listBodybody
        response = client.upload_autopilot_chart(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
```

```
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UploadAutopilotChartRequest{}
    parametersBody:= "{"override":true,"skip_lint":true,"source":"package"}"
    bodybody := &model.UploadAutopilotChartRequestBody{
        Parameters: &parametersBody,
        Content: "chart-file.tgz",
    }
    request.Body = listBodybody
    response, err := client.UploadAutopilotChart(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	Created

错误码

请参见**错误码**。

4.6.2 获取模板列表

功能介绍

获取模板列表

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v2/charts

请求参数

表 4-385 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-386 响应 Body 参数

参数	参数类型	描述
[数组元素]	Array of **ChartResp** objects	模板列表

表 4-387 ChartResp

参数	参数类型	描述
id	String	模板ID
name	String	模板名称
values	String	模板值
translate	String	模板翻译资源
instruction	String	模板介绍
version	String	模板版本
description	String	模板描述
source	String	模板的来源
icon_url	String	模板的图标链接
public	Boolean	是否公开模板
chart_url	String	模板的链接
create_at	String	创建时间

参数	参数类型	描述
update_at	String	更新时间

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotChartsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotChartsRequest request = new ListAutopilotChartsRequest();
        try {
            ListAutopilotChartsResponse response = client.listAutopilotCharts(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
```

```
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotChartsRequest()
        response = client.list_autopilot_charts(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
```

```
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

    request := &model.ListAutopilotChartsRequest{}
    response, err := client.ListAutopilotCharts(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.3 获取模板实例列表

功能介绍

获取模板实例列表

调用方法

请参见**如何调用API**。

URI

GET /autopilot/cam/v3/clusters/{cluster_id}/releases

表 4-388 路径参数

参数	是否必选	参数类型	描述
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

表 4-389 Query 参数

参数	是否必选	参数类型	描述
chart_id	否	String	模板ID
namespace	否	String	模板对应的命名空间

请求参数

表 4-390 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-391 响应 Body 参数

参数	参数类型	描述
[数组元素]	Array<Array<**ReleaseResp**>>	OK

表 4-392 ReleaseResp

参数	参数类型	描述
chart_name	String	模板名称
chart_public	Boolean	是否公开模板
chart_version	String	模板版本
cluster_id	String	集群ID
cluster_name	String	集群名称
create_at	String	创建时间

参数	参数类型	描述
description	String	模板实例描述
name	String	模板实例名称
namespace	String	模板实例所在的命名空间
parameters	String	模板实例参数
resources	String	模板实例需要的资源
status	String	模板实例状态 ● DEPLOYED：已部署，表示模板实例处于正常状态。 ● DELETED：已删除，表示模板实例已经被删除。 ● FAILED：失败，表示模板实例部署失败。 ● DELETING：删除中，表示模板实例正处于删除过程中。 ● PENDING_INSTALL：待安装，表示模板正在等待安装。 ● PENDING_UPGRADE：待升级，表示模板正在等待升级。 ● PENDING_ROLLBACK：待回滚，表示模板正在等待回滚。 ● UNKNOWN：未知，表示模板状态异常，可尝试手动删除后重新安装。
status_description	String	模板实例状态描述
update_at	String	更新时间
values	String	模板实例的值
version	Integer	模板实例版本

请求示例

无

响应示例

状态码： 200

OK

```
[ {
  "chart_name" : "magento-mysql",
  "chart_public" : false,
  "chart_version" : "1.0.0",
  "cluster_id" : "a870253f-5dc7-11ee-bf71-0255ac100b03",
```

```
  "cluster_name" : "sfs-turbo-test",
  "create_at" : "2023-11-14T20:30:57+08:00",
  "description" : "Initial install underway",
  "name" : "testwww",
  "namespace" : "monitoring",
  "parameters" : "",
  "resources" : "",
  "status" : "PENDING_INSTALL",
  "status_description" : "Initial install underway",
  "update_at" : "2023-11-14T20:30:57+08:00",
  "values" : "{\"basic\":{\"admin_password\":\"******\",\"admin_username\":\"username\",\"app_name
\":\"magento\",\"mysql_database\":\"magento\",\"mysql_name\":\"mysql\",\"mysql_password\":\"******
\",\"mysql_port\":3306,\"mysql_root_password\":\"******\",\"mysql_user\":\"magento\",\"storage_class\":\"csi-
nas\",\"storage_mode\":\"ReadWriteMany\",\"storage_size\":\"10G\"},\"global\":{\"magento_EIP
\":\"100.100.100.100\",\"magento_EPORT\":32080,\"namespace\":\"default\"},\"image\":{\"magento_image
\":\"example.com/everest/magento:latest\",\"mysql_image\":\"example.com/everest/mysql:5.7.14\"}}",
  "version" : 1
} ]
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ListAutopilotReleasesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ListAutopilotReleasesRequest request = new ListAutopilotReleasesRequest();
        request.withClusterId("{cluster_id}");
        try {
            ListAutopilotReleasesResponse response = client.listAutopilotReleases(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
```

```
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListAutopilotReleasesRequest()
        request.cluster_id = "{cluster_id}"
        response = client.list_autopilot_releases(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()
```

```
    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListAutopilotReleasesRequest{}
    request.ClusterId = "{cluster_id}"
    response, err := client.ListAutopilotReleases(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.4 创建模板实例

功能介绍

创建模板实例

调用方法

请参见**如何调用API**。

URI

POST /autopilot/cam/v3/clusters/{cluster_id}/releases

表 4-393 路径参数

参数	是否必选	参数类型	描述
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-394 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-395 请求 Body 参数

参数	是否必选	参数类型	描述
chart_id	是	String	模板ID
description	否	String	模板实例描述
name	是	String	模板实例名称
namespace	是	String	模板实例所在的命名空间
version	是	String	模板实例版本号
parameters	否	**ReleaseReqBodyParams** object	模板实例参数
values	是	**values** object	模板实例的值

表 4-396 ReleaseReqBodyParams

参数	是否必选	参数类型	描述
dry_run	否	Boolean	开启后，仅验证模板参数，不进行安装
name_template	否	String	实例名称模板
no_hooks	否	Boolean	安装时是否禁用hooks
replace	否	Boolean	是否替换同名实例
recreate	否	Boolean	是否重建实例
reset_values	否	Boolean	更新时是否重置values

参数	是否必选	参数类型	描述
release_version	否	Integer	回滚实例的版本
include_hooks	否	Boolean	更新或者删除时启用hooks

表 4-397 values

参数	是否必选	参数类型	描述
imagePullPolicy	否	String	镜像拉取策略
imageTag	否	String	镜像标签

响应参数

状态码： 201

表 4-398 响应 Body 参数

参数	参数类型	描述
chart_name	String	模板名称
chart_public	Boolean	是否公开模板
chart_version	String	模板版本
cluster_id	String	集群ID
cluster_name	String	集群名称
create_at	String	创建时间
description	String	模板实例描述
name	String	模板实例名称
namespace	String	模板实例所在的命名空间
parameters	String	模板实例参数
resources	String	模板实例需要的资源

参数	参数类型	描述
status	String	模板实例状态 ● DEPLOYED：已部署，表示模板实例处于正常状态。 ● DELETED：已删除，表示模板实例已经被删除。 ● FAILED：失败，表示模板实例部署失败。 ● DELETING：删除中，表示模板实例正处于删除过程中。 ● PENDING_INSTALL：待安装，表示模板正在等待安装。 ● PENDING_UPGRADE：待升级，表示模板正在等待升级。 ● PENDING_ROLLBACK：待回滚，表示模板正在等待回滚。 ● UNKNOWN：未知，表示模板状态异常，可尝试手动删除后重新安装。
status_description	String	模板实例状态描述
update_at	String	更新时间
values	String	模板实例的值
version	Integer	模板实例版本

请求示例

```
POST /autopilot/cam/v3/clusters/{cluster_id}/releases

{
  "name" : "koi-neo4j",
  "project_id" : "0abdd2dce980d4162f8ac006608ee02d",
  "cluster_id" : "7378a198-a3fe-11eb-ad37-0255ac100b07",
  "namespace" : "default",
  "chart_id" : "e99a7e86-afdd-11eb-aca3-0255ac100b0e",
  "description" : "",
  "version" : "3.0.1",
  "values" : {
    "acceptLicenseAgreement" : "no",
    "affinity" : { },
    "authEnabled" : true,
    "clusterDomain" : "cluster.local",
    "core" : {
      "initContainers" : [ ],
      "numberOfServers" : 3,
      "persistentVolume" : {
        "enabled" : true,
        "mountPath" : "/data",
        "size" : "10Gi"
      },
      "sidecarContainers" : [ ]
    },
    "defaultDatabase" : "neo4j",
```

```
    "image" : "neo4j",
    "imagePullPolicy" : "IfNotPresent",
    "imageTag" : "4.0.3-enterprise",
    "name" : "neo4j",
    "nodeSelector" : { },
    "podDisruptionBudget" : { },
    "readReplica" : {
      "autoscaling" : {
        "enabled" : false,
        "maxReplicas" : 3,
        "minReplicas" : 1,
        "targetAverageUtilization" : 70
      },
      "initContainers" : [ ],
      "numberOfServers" : 0,
      "resources" : { },
      "sidecarContainers" : [ ]
    },
    "resources" : { },
    "testImage" : "markhneedham/k8s-kubectl",
    "testImageTag" : "master",
    "tolerations" : [ ],
    "useAPOC" : "true"
  }
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class CreateAutopilotReleaseSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
```



```
        CreateAutopilotReleaseRequest request = new CreateAutopilotReleaseRequest();
        request.withClusterId("{cluster_id}");
        CreateReleaseReqBody body = new CreateReleaseReqBody();
        CreateReleaseReqBodyValues valuesbody = new CreateReleaseReqBodyValues();
        valuesbody.withImagePullPolicy("IfNotPresent")
            .withImageTag("4.0.3-enterprise");
        body.withValues(valuesbody);
        body.withVersion("3.0.1");
        body.withNamespace("default");
        body.withName("koi-neo4j");
        body.withDescription("");
        body.withChartId("e99a7e86-afdd-11eb-aca3-0255ac100b0e");
        request.withBody(body);
        try {
            CreateAutopilotReleaseResponse response = client.createAutopilotRelease(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateAutopilotReleaseRequest()
        request.cluster_id = "{cluster_id}"
        valuesbody = CreateReleaseReqBodyValues(
            image_pull_policy="IfNotPresent",
            image_tag="4.0.3-enterprise"
        )
        request.body = CreateReleaseReqBody(
            values=valuesbody,
            version="3.0.1",
            namespace="default",
            name="koi-neo4j",
            description="",
```



```
            chart_id="e99a7e86-afdd-11eb-aca3-0255ac100b0e"
        )
        response = client.create_autopilot_release(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateAutopilotReleaseRequest{}
    request.ClusterId = "{cluster_id}"
    imagePullPolicyValues:= "IfNotPresent"
    imageTagValues:= "4.0.3-enterprise"
    valuesbody := &model.CreateReleaseReqBodyValues{
        ImagePullPolicy: &imagePullPolicyValues,
        ImageTag: &imageTagValues,
    }
    descriptionCreateReleaseReqBody:= ""
    request.Body = &model.CreateReleaseReqBody{
        Values: valuesbody,
        Version: "3.0.1",
        Namespace: "default",
        Name: "koi-neo4j",
        Description: &descriptionCreateReleaseReqBody,
        ChartId: "e99a7e86-afdd-11eb-aca3-0255ac100b0e",
    }
    response, err := client.CreateAutopilotRelease(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```


更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	Created

错误码

请参见**错误码**。

4.6.5 更新模板

功能介绍

更新模板

调用方法

请参见**如何调用API**。

URI

PUT /autopilot/v2/charts/{chart_id}

表 4-399 路径参数

参数	是否必选	参数类型	描述
chart_id	是	String	模板的ID

请求参数

表 4-400 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-401 FormData 参数

参数	是否必选	参数类型	描述
parameters	否	String	更新模板的配置参数，示例如下： "{"override":true,"skip_lint":true,"source":"package"}" • skip_lint: 是否验证上传的模板 • override: 是否覆盖已存在的模板 • visible: 模板是否可见
content	是	File	模板包文件

响应参数

状态码： 200

表 4-402 响应 Body 参数

参数	参数类型	描述
id	String	模板ID
name	String	模板名称
values	String	模板值
translate	String	模板翻译资源
instruction	String	模板介绍
version	String	模板版本
description	String	模板描述
source	String	模板的来源
icon_url	String	模板的图标链接
public	Boolean	是否公开模板
chart_url	String	模板的链接
create_at	String	创建时间
update_at	String	更新时间

请求示例

```
PUT /autopilot/v2/charts/{chart_id}
```

```
{
  "parameters" : "{\"override\":true,\"skip_lint\":true,\"source\":\"package\"}",
  "content" : "chart-file.tgz"
}
```

响应示例

状态码： 200

OK

```
{
  "id" : "e99a7e86-afdd-11eb-aca3-0255ac100b0e",
  "name" : "neo4j",
  "values" : "{\"acceptLicenseAgreement\":\"no\",\"affinity\":{},\"authEnabled\":true,\"clusterDomain
\":\"cluster.local\",\"core\":{\"initContainers\":[],\"numberOfServers\":3,\"persistentVolume\":{\"enabled
\":true,\"mountPath\":\"/data\",\"size\":\"10Gi\"},\"sidecarContainers\":[]},\"defaultDatabase\":\"neo4j
\",\"image\":\"neo4j\",\"imagePullPolicy\":\"IfNotPresent\",\"imageTag\":\"4.0.3-enterprise\",\"name
\":\"neo4j\",\"nodeSelector\":{},\"podDisruptionBudget\":{},\"readReplica\":{\"autoscaling\":{\"enabled
\":false,\"maxReplicas\":3,\"minReplicas\":1,\"targetAverageUtilization\":70},\"initContainers\":
[],\"numberOfServers\":0,\"resources\":{},\"sidecarContainers\":[]},\"resources\":{},\"testImage
\":\"markhneedham/k8s-kubectl\",\"testImageTag\":\"master\",\"tolerations\":[],\"useAPOC\":\"true\"}",
  "translate" : "",
  "instruction" : "README.md",
  "version" : "3.0.1",
  "description" : "DEPRECATED Neo4j is the world's leading graph database",
  "source" : "",
  "icon_url" : "https://example.com/images/neo4j_logo.png",
  "public" : false,
  "chart_url" : "neo4j-3.0.1.tgz",
  "create_at" : "2021-05-08T08:53:12Z",
  "update_at" : "2021-05-08T08:53:12Z"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class UpdateAutopilotChartSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);
```

```
        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateAutopilotChartRequest request = new UpdateAutopilotChartRequest();
        request.withChartId("{chart_id}");
        UpdateAutopilotChartRequestBody bodybody = new UpdateAutopilotChartRequestBody();
        bodybody.withParameters("{"override":true,"skip_lint":true,"source":"package"}")
            .withContent("chart-file.tgz");
        body.withBody(bodybody);
        request.withBody(listbodyBody);
        try {
            UpdateAutopilotChartResponse response = client.updateAutopilotChart(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateAutopilotChartRequest()
        request.chart_id = "{chart_id}"
        bodybody = UpdateAutopilotChartRequestBody(
            parameters="{"override":true,"skip_lint":true,"source":"package"}",
            content="chart-file.tgz"
        )
        request.body = listBodybody
        response = client.update_autopilot_chart(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateAutopilotChartRequest{}
    request.ChartId = "{chart_id}"
    parametersBody:= "{"override":true,"skip_lint":true,"source":"package"}"
    bodybody := &model.UpdateAutopilotChartRequestBody{
        Parameters: &parametersBody,
        Content: "chart-file.tgz",
    }
    request.Body = listBodybody
    response, err := client.UpdateAutopilotChart(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.6 删除模板

功能介绍

删除模板

调用方法

请参见**如何调用API**。

URI

DELETE /autopilot/v2/charts/{chart_id}

表 4-403 路径参数

参数	是否必选	参数类型	描述
chart_id	是	String	模板的ID

请求参数

表 4-404 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-405 响应 Body 参数

参数	参数类型	描述
-	String	

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class DeleteAutopilotChartSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteAutopilotChartRequest request = new DeleteAutopilotChartRequest();
        request.withChartId("{chart_id}");
        try {
            DeleteAutopilotChartResponse response = client.deleteAutopilotChart(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
```

```
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteAutopilotChartRequest()
        request.chart_id = "{chart_id}"
        response = client.delete_autopilot_chart(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteAutopilotChartRequest{}
    request.ChartId = "{chart_id}"
    response, err := client.DeleteAutopilotChart(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
```



```
  }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.7 更新指定模板实例

功能介绍

更新指定模板实例

调用方法

请参见**如何调用API**。

URI

PUT /autopilot/cam/v3/clusters/{cluster_id}/namespace/{namespace}/releases/{name}

表 4-406 路径参数

参数	是否必选	参数类型	描述
name	是	String	模板实例名称
namespace	是	String	模板实例所在的命名空间
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-407 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

表 4-408 请求 Body 参数

参数	是否必选	参数类型	描述
chart_id	是	String	模板ID
action	是	String	更新操作，升级为upgrade，回退为rollback
parameters	是	**ReleaseReqBodyParams** object	模板实例参数
values	是	**values** object	模板实例的值

表 4-409 ReleaseReqBodyParams

参数	是否必选	参数类型	描述
dry_run	否	Boolean	开启后，仅验证模板参数，不进行安装
name_template	否	String	实例名称模板
no_hooks	否	Boolean	安装时是否禁用hooks
replace	否	Boolean	是否替换同名实例
recreate	否	Boolean	是否重建实例
reset_values	否	Boolean	更新时是否重置values
release_version	否	Integer	回滚实例的版本
include_hooks	否	Boolean	更新或者删除时启用hooks

表 4-410 values

参数	是否必选	参数类型	描述
imagePullPolicy	否	String	镜像拉取策略
imageTag	否	String	镜像标签

响应参数

状态码： 200

表 4-411 响应 Body 参数

参数	参数类型	描述
chart_name	String	模板名称
chart_public	Boolean	是否公开模板
chart_version	String	模板版本
cluster_id	String	集群ID
cluster_name	String	集群名称
create_at	String	创建时间
description	String	模板实例描述
name	String	模板实例名称
namespace	String	模板实例所在的命名空间
parameters	String	模板实例参数
resources	String	模板实例需要的资源

参数	参数类型	描述
status	String	模板实例状态 ● DEPLOYED：已部署，表示模板实例处于正常状态。 ● DELETED：已删除，表示模板实例已经被删除。 ● FAILED：失败，表示模板实例部署失败。 ● DELETING：删除中，表示模板实例正处于删除过程中。 ● PENDING_INSTALL：待安装，表示模板正在等待安装。 ● PENDING_UPGRADE：待升级，表示模板正在等待升级。 ● PENDING_ROLLBACK：待回滚，表示模板正在等待回滚。 ● UNKNOWN：未知，表示模板状态异常，可尝试手动删除后重新安装。
status_description	String	模板实例状态描述
update_at	String	更新时间
values	String	模板实例的值
version	Integer	模板实例版本

请求示例

```
PUT /autopilot/cam/v3/clusters/{cluster_id}/namespace/{namespace}/releases/{name}

{
  "chart_id" : "af4b699e-018c-11ec-b8b0-0255ac100b05",
  "action" : "upgrade",
  "parameters" : {
    "dry_run" : false,
    "name_template" : "string",
    "no_hooks" : false,
    "replace" : false,
    "recreate" : false,
    "reset_values" : false,
    "release_version" : 1,
    "include_hooks" : false
  },
  "values" : {
    "imagePullPolicy" : "IfNotPresent",
    "imageTag" : "v2"
  }
}
```

响应示例

状态码： 200

OK


```
{
  "chart_name" : "magento-mysql",
  "chart_public" : false,
  "chart_version" : "1.0.0",
  "cluster_id" : "a870253f-5dc7-11ee-bf71-0255ac100b03",
  "cluster_name" : "sfs-turbo-test",
  "create_at" : "2023-11-14T20:30:57+08:00",
  "description" : "Initial install underway",
  "name" : "testwww",
  "namespace" : "monitoring",
  "parameters" : "",
  "resources" : "",
  "status" : "PENDING_INSTALL",
  "status_description" : "Initial install underway",
  "update_at" : "2023-11-14T20:30:57+08:00",
  "values" : "{\"basic\":{\"admin_password\":\"******\",\"admin_username\":\"username\",\"app_name
\":\"magento\",\"mysql_database\":\"magento\",\"mysql_name\":\"mysql\",\"mysql_password\":\"******
\",\"mysql_port\":3306,\"mysql_root_password\":\"******\",\"mysql_user\":\"magento\",\"storage_class\":\"csi-
nas\",\"storage_mode\":\"ReadWriteMany\",\"storage_size\":\"10G\"},\"global\":{\"magento_EIP
\":\"100.100.100.100\",\"magento_EPORT\":32080,\"namespace\":\"default\"},\"image\":{\"magento_image
\":\"example.com/everest/magento:latest\",\"mysql_image\":\"example.com/everest/mysql:5.7.14\"}}",
  "version" : 1
}
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class UpdateAutopilotReleaseSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        UpdateAutopilotReleaseRequest request = new UpdateAutopilotReleaseRequest();
        request.withName("{name}");
        request.withNamespace("{namespace}");
        request.withClusterId("{cluster_id}");
        UpdateReleaseReqBody body = new UpdateReleaseReqBody();
```

```
        UpdateReleaseReqBodyValues valuesbody = new UpdateReleaseReqBodyValues();
        valuesbody.withImagePullPolicy("IfNotPresent")
            .withImageTag("v2");
        ReleaseReqBodyParams parametersbody = new ReleaseReqBodyParams();
        parametersbody.withDryRun(false)
            .withNameTemplate("string")
            .withNoHooks(false)
            .withReplace(false)
            .withRecreate(false)
            .withResetValues(false)
            .withReleaseVersion(1)
            .withIncludeHooks(false);
        body.withValues(valuesbody);
        body.withParameters(parametersbody);
        body.withAction(UpdateReleaseReqBody.ActionEnum.fromValue("upgrade"));
        body.withChartId("af4b699e-018c-11ec-b8b0-0255ac100b05");
        request.withBody(body);
        try {
            UpdateAutopilotReleaseResponse response = client.updateAutopilotRelease(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateAutopilotReleaseRequest()
        request.name = "{name}"
        request.namespace = "{namespace}"
        request.cluster_id = "{cluster_id}"
        valuesbody = UpdateReleaseReqBodyValues(
            image_pull_policy="IfNotPresent",
            image_tag="v2"
        )
```



```
        parametersbody = ReleaseReqBodyParams(
            dry_run=False,
            name_template="string",
            no_hooks=False,
            replace=False,
            recreate=False,
            reset_values=False,
            release_version=1,
            include_hooks=False
        )
        request.body = UpdateReleaseReqBody(
            values=valuesbody,
            parameters=parametersbody,
            action="upgrade",
            chart_id="af4b699e-018c-11ec-b8b0-0255ac100b05"
        )
        response = client.update_autopilot_release(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateAutopilotReleaseRequest{}
    request.Name = "{name}"
    request.Namespace = "{namespace}"
    request.ClusterId = "{cluster_id}"
    imagePullPolicyValues:= "IfNotPresent"
    imageTagValues:= "v2"
    valuesbody := &model.UpdateReleaseReqBodyValues{
        ImagePullPolicy: &imagePullPolicyValues,
        ImageTag: &imageTagValues,
    }
    dryRunParameters:= false
    nameTemplateParameters:= "string"
    noHooksParameters:= false
```



```
    replaceParameters:= false
    recreateParameters:= false
    resetValuesParameters:= false
    releaseVersionParameters:= int32(1)
    includeHooksParameters:= false
    parametersbody := &model.ReleaseReqBodyParams{
        DryRun: &dryRunParameters,
        NameTemplate: &nameTemplateParameters,
        NoHooks: &noHooksParameters,
        Replace: &replaceParameters,
        Recreate: &recreateParameters,
        ResetValues: &resetValuesParameters,
        ReleaseVersion: &releaseVersionParameters,
        IncludeHooks: &includeHooksParameters,
    }
    request.Body = &model.UpdateReleaseReqBody{
        Values: valuesbody,
        Parameters: parametersbody,
        Action: model.GetUpdateReleaseReqBodyActionEnum().UPGRADE,
        ChartId: "af4b699e-018c-11ec-b8b0-0255ac100b05",
    }
    response, err := client.UpdateAutopilotRelease(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.8 获取模板

功能介绍

获取模板

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v2/charts/{chart_id}

表 4-412 路径参数

参数	是否必选	参数类型	描述
chart_id	是	String	模板的ID

请求参数

表 4-413 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-414 响应 Body 参数

参数	参数类型	描述
id	String	模板ID
name	String	模板名称
values	String	模板值
translate	String	模板翻译资源
instruction	String	模板介绍
version	String	模板版本
description	String	模板描述
source	String	模板的来源
icon_url	String	模板的图标链接
public	Boolean	是否公开模板
chart_url	String	模板的链接
create_at	String	创建时间
update_at	String	更新时间

请求示例

无

响应示例

状态码： 200

OK


```
{
  "id" : "e99a7e86-afdd-11eb-aca3-0255ac100b0e",
  "name" : "neo4j",
  "values" : "{\"acceptLicenseAgreement\":\"no\",\"affinity\":{},\"authEnabled\":true,\"clusterDomain
\":\"cluster.local\",\"core\":{\"initContainers\":[],\"numberOfServers\":3,\"persistentVolume\":{\"enabled
\":true,\"mountPath\":\"/data\",\"size\":\"10Gi\"},\"sidecarContainers\":[]},\"defaultDatabase\":\"neo4j
\",\"image\":\"neo4j\",\"imagePullPolicy\":\"IfNotPresent\",\"imageTag\":\"4.0.3-enterprise\",\"name
\":\"neo4j\",\"nodeSelector\":{},\"podDisruptionBudget\":{},\"readReplica\":{\"autoscaling\":{\"enabled
\":false,\"maxReplicas\":3,\"minReplicas\":1,\"targetAverageUtilization\":70},\"initContainers\":
[],\"numberOfServers\":0,\"resources\":{},\"sidecarContainers\":[]},\"resources\":{},\"testImage
\":\"markhneedham/k8s-kubectl\",\"testImageTag\":\"master\",\"tolerations\":[],\"useAPOC\":\"true\"}",
  "translate" : "",
  "instruction" : "README.md",
  "version" : "3.0.1",
  "description" : "DEPRECATED Neo4j is the world's leading graph database",
  "source" : "",
  "icon_url" : "https://info.neo4j.com/rs/773-GON-065/images/neo4j_logo.png",
  "public" : false,
  "chart_url" : "neo4j-3.0.1.tgz",
  "create_at" : "2021-05-08T08:53:13Z",
  "update_at" : "2021-05-08T08:53:13Z"
}
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotChartSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);
```

```
        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotChartRequest request = new ShowAutopilotChartRequest();
        request.withChartId("{chart_id}");
        try {
            ShowAutopilotChartResponse response = client.showAutopilotChart(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotChartRequest()
        request.chart_id = "{chart_id}"
        response = client.show_autopilot_chart(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
```

```
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotChartRequest{}
    request.ChartId = "{chart_id}"
    response, err := client.ShowAutopilotChart(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.9 删除指定模板实例

功能介绍

删除指定模板实例

调用方法

请参见**如何调用API**。

URI

DELETE /autopilot/cam/v3/clusters/{cluster_id}/namespace/{namespace}/releases/{name}

表 4-415 路径参数

参数	是否必选	参数类型	描述
name	是	String	模板实例名称
namespace	是	String	模板实例所在的命名空间
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-416 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-417 响应 Body 参数

参数	参数类型	描述
-	String	

请求示例

无

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class DeleteAutopilotReleaseSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        DeleteAutopilotReleaseRequest request = new DeleteAutopilotReleaseRequest();
        request.withName("{name}");
        request.withNamespace("{namespace}");
        request.withClusterId("{cluster_id}");
        try {
            DeleteAutopilotReleaseResponse response = client.deleteAutopilotRelease(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *
```

```
if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteAutopilotReleaseRequest()
        request.name = "{name}"
        request.namespace = "{namespace}"
        request.cluster_id = "{cluster_id}"
        response = client.delete_autopilot_release(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteAutopilotReleaseRequest{}
    request.Name = "{name}"
    request.Namespace = "{namespace}"
    request.ClusterId = "{cluster_id}"
    response, err := client.DeleteAutopilotRelease(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
```



```
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.10 获取指定模板实例

功能介绍

获取指定模板实例

调用方法

请参见**如何调用API**。

URI

GET /autopilot/cam/v3/clusters/{cluster_id}/namespace/{namespace}/releases/{name}

表 4-418 路径参数

参数	是否必选	参数类型	描述
name	是	String	模板实例名称
namespace	是	String	模板实例所在的命名空间
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-419 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-420 响应 Body 参数

参数	参数类型	描述
chart_name	String	模板名称
chart_public	Boolean	是否公开模板
chart_version	String	模板版本
cluster_id	String	集群ID
cluster_name	String	集群名称
create_at	String	创建时间
description	String	模板实例描述
name	String	模板实例名称
namespace	String	模板实例所在的命名空间
parameters	String	模板实例参数
resources	String	模板实例需要的资源

参数	参数类型	描述
status	String	模板实例状态 ● DEPLOYED：已部署，表示模板实例处于正常状态。 ● DELETED：已删除，表示模板实例已经被删除。 ● FAILED：失败，表示模板实例部署失败。 ● DELETING：删除中，表示模板实例正处于删除过程中。 ● PENDING_INSTALL：待安装，表示模板正在等待安装。 ● PENDING_UPGRADE：待升级，表示模板正在等待升级。 ● PENDING_ROLLBACK：待回滚，表示模板正在等待回滚。 ● UNKNOWN：未知，表示模板状态异常，可尝试手动删除后重新安装。
status_description	String	模板实例状态描述
update_at	String	更新时间
values	String	模板实例的值
version	Integer	模板实例版本

请求示例

无

响应示例

状态码： 200

OK


```
{
  "chart_name" : "magento-mysql",
  "chart_public" : false,
  "chart_version" : "1.0.0",
  "cluster_id" : "a870253f-5dc7-11ee-bf71-0255ac100b03",
  "cluster_name" : "sfs-turbo-test",
  "create_at" : "2023-11-14T20:30:57+08:00",
  "description" : "Initial install underway",
  "name" : "testwww",
  "namespace" : "monitoring",
  "parameters" : "",
  "resources" : "",
  "status" : "PENDING_INSTALL",
  "status_description" : "Initial install underway",
  "update_at" : "2023-11-14T20:30:57+08:00",
  "values" : "{\"basic\":{\"admin_password\":\"******\",\"admin_username\":\"username\",\"app_name
```

\":\"magento\",\"mysql_database\":\"magento\",\"mysql_name\":\"mysql\",\"mysql_password\":\"******\",\"mysql_port\":3306,\"mysql_root_password\":\"******\",\"mysql_user\":\"magento\",\"storage_class\":\"csi-nas\",\"storage_mode\":\"ReadWriteMany\",\"storage_size\":\"10G\"},\"global\":{\"magento_EIP\":\"100.100.100.100\",\"magento_EPORT\":32080,\"namespace\":\"default\"},\"image\":{\"magento_image\":\"example.com/everest/magento:latest\",\"mysql_image\":\"example.com/everest/mysql:5.7.14\"}}",
  "version" : 1
}

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotReleaseSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotReleaseRequest request = new ShowAutopilotReleaseRequest();
        request.withName("{name}");
        request.withNamespace("{namespace}");
        request.withClusterId("{cluster_id}");
        try {
            ShowAutopilotReleaseResponse response = client.showAutopilotRelease(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```


Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotReleaseRequest()
        request.name = "{name}"
        request.namespace = "{namespace}"
        request.cluster_id = "{cluster_id}"
        response = client.show_autopilot_release(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
```

```
        Build())

    request := &model.ShowAutopilotReleaseRequest{}
    request.Name = "{name}"
    request.Namespace = "{namespace}"
    request.ClusterId = "{cluster_id}"
    response, err := client.ShowAutopilotRelease(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.11 下载模板

功能介绍

下载模板

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v2/charts/{chart_id}/archive

表 4-421 路径参数

参数	是否必选	参数类型	描述
chart_id	是	String	模板的ID

请求参数

表 4-422 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-423 响应 Body 参数

参数	参数类型	描述
-	File	

请求示例

无

响应示例

状态码： 200

OK

```
"chart-file.tgz"
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;
```

```
public class DownloadAutopilotChartSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        DownloadAutopilotChartRequest request = new DownloadAutopilotChartRequest();
        request.withChartId("{chart_id}");
        try {
            DownloadAutopilotChartResponse response = client.downloadAutopilotChart(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DownloadAutopilotChartRequest()
        request.chart_id = "{chart_id}"
        response = client.download_autopilot_chart(request)
        print(response)
```

```
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DownloadAutopilotChartRequest{}
    request.ChartId = "{chart_id}"
    response, err := client.DownloadAutopilotChart(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.12 获取模板 Values

功能介绍

获取模板Values

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v2/charts/{chart_id}/values

表 4-424 路径参数

参数	是否必选	参数类型	描述
chart_id	是	String	模板的ID

请求参数

表 4-425 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-426 响应 Body 参数

参数	参数类型	描述
values	Map<String,Object>	values.yaml中的数据，数据结构以具体的模板为准。

请求示例

无

响应示例

状态码： 200

OK

```
{
  "values" : {
    "basic" : {
      "admin_password" : "******",
      "admin_username" : "username"
    },
    "global" : {
      "magento_EIP" : "127.0.0.1",
      "magento_EPORT" : 32080,
      "namespace" : "demo"
    },
    "image" : {
      "magento_image" : "example.com/demo/magento:latest",
      "mysql_image" : "example.com/demo/mysql:5.7.14"
    }
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotChartValuesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotChartValuesRequest request = new ShowAutopilotChartValuesRequest();
        request.withChartId("{chart_id}");
        try {
            ShowAutopilotChartValuesResponse response = client.showAutopilotChartValues(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
```

```
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotChartValuesRequest()
        request.chart_id = "{chart_id}"
        response = client.show_autopilot_chart_values(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
```



```
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotChartValuesRequest{}
    request.ChartId = "{chart_id}"
    response, err := client.ShowAutopilotChartValues(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.13 查询指定模板实例历史记录

功能介绍

查询指定模板实例历史记录

调用方法

请参见**如何调用API**。

URI

GET /autopilot/cam/v3/clusters/{cluster_id}/namespace/{namespace}/releases/{name}/history

表 4-427 路径参数

参数	是否必选	参数类型	描述
name	是	String	模板实例名称
namespace	是	String	模板实例所在的命名空间
cluster_id	是	String	集群ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-428 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-429 响应 Body 参数

参数	参数类型	描述
[数组元素]	Array<Array<**ReleaseResp**>>	OK

表 4-430 ReleaseResp

参数	参数类型	描述
chart_name	String	模板名称
chart_public	Boolean	是否公开模板
chart_version	String	模板版本
cluster_id	String	集群ID
cluster_name	String	集群名称

参数	参数类型	描述
create_at	String	创建时间
description	String	模板实例描述
name	String	模板实例名称
namespace	String	模板实例所在的命名空间
parameters	String	模板实例参数
resources	String	模板实例需要的资源
status	String	模板实例状态 ● DEPLOYED：已部署，表示模板实例处于正常状态。 ● DELETED：已删除，表示模板实例已经被删除。 ● FAILED：失败，表示模板实例部署失败。 ● DELETING：删除中，表示模板实例正处于删除过程中。 ● PENDING_INSTALL：待安装，表示模板正在等待安装。 ● PENDING_UPGRADE：待升级，表示模板正在等待升级。 ● PENDING_ROLLBACK：待回滚，表示模板正在等待回滚。 ● UNKNOWN：未知，表示模板状态异常，可尝试手动删除后重新安装。
status_description	String	模板实例状态描述
update_at	String	更新时间
values	String	模板实例的值
version	Integer	模板实例版本

请求示例

无

响应示例

状态码： 200

OK

```
[ {
  "chart_name" : "magento-mysql",
  "chart_public" : false,
```

```
  "chart_version" : "1.0.0",
  "cluster_id" : "a870253f-5dc7-11ee-bf71-0255ac100b03",
  "cluster_name" : "sfs-turbo-test",
  "create_at" : "2023-11-14T20:30:57+08:00",
  "description" : "Initial install underway",
  "name" : "testwww",
  "namespace" : "monitoring",
  "parameters" : "",
  "resources" : "",
  "status" : "PENDING_INSTALL",
  "status_description" : "Initial install underway",
  "update_at" : "2023-11-14T20:30:57+08:00",
  "values" : "{\"basic\":{\"admin_password\":\"******\",\"admin_username\":\"username\",\"app_name
\":\"magento\",\"mysql_database\":\"magento\",\"mysql_name\":\"mysql\",\"mysql_password\":\"******
\",\"mysql_port\":3306,\"mysql_root_password\":\"******\",\"mysql_user\":\"magento\",\"storage_class\":\"csi-
nas\",\"storage_mode\":\"ReadWriteMany\",\"storage_size\":\"10G\"},\"global\":{\"magento_EIP
\":\"100.100.100.100\",\"magento_EPORT\":32080,\"namespace\":\"default\"},\"image\":{\"magento_image
\":\"example.com/everest/magento:latest\",\"mysql_image\":\"example.com/everest/mysql:5.7.14\"}}",
  "version" : 1
} ]
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotReleaseHistorySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
                .build();
        ShowAutopilotReleaseHistoryRequest request = new ShowAutopilotReleaseHistoryRequest();
        request.withName("{name}");
        request.withNamespace("{namespace}");
        request.withClusterId("{cluster_id}");
        try {
            ShowAutopilotReleaseHistoryResponse response = client.showAutopilotReleaseHistory(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
```



```
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]

    credentials = BasicCredentials(ak, sk)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotReleaseHistoryRequest()
        request.name = "{name}"
        request.namespace = "{namespace}"
        request.cluster_id = "{cluster_id}"
        response = client.show_autopilot_release_history(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
```

```
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotReleaseHistoryRequest{}
    request.Name = "{name}"
    request.Namespace = "{namespace}"
    request.ClusterId = "{cluster_id}"
    response, err := client.ShowAutopilotReleaseHistory(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

4.6.14 获取用户模板配额

功能介绍

获取用户模板配额

调用方法

请参见**如何调用API**。

URI

GET /autopilot/v2/charts/{project_id}/quotas

表 4-431 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方式请参见**如何获取接口URI中参数**。

请求参数

表 4-432 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式）
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

响应参数

状态码： 200

表 4-433 响应 Body 参数

参数	参数类型	描述
quotas	**quotas** object	模板配额

表 4-434 quotas

参数	参数类型	描述
resources	Array of **resources** objects	资源

表 4-435 resources

参数	参数类型	描述
type	String	类型
quota	Integer	配额

参数	参数类型	描述
used	Integer	已使用量

请求示例

无

响应示例

状态码： 200

OK

```
{
  "quotas" : {
    "resources" : [ {
      "type" : "Charts",
      "quota" : 200,
      "used" : 2
    } ]
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.cce.v3.region.CceRegion;
import com.huaweicloud.sdk.cce.v3.*;
import com.huaweicloud.sdk.cce.v3.model.*;


public class ShowAutopilotUserChartsQuotasSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
        String projectId = "{project_id}";

        ICredential auth = new BasicCredentials()
                .withProjectId(projectId)
                .withAk(ak)
                .withSk(sk);

        CceClient client = CceClient.newBuilder()
                .withCredential(auth)
                .withRegion(CceRegion.valueOf("<YOUR REGION>"))
```

```
                .build();
        ShowAutopilotUserChartsQuotasRequest request = new ShowAutopilotUserChartsQuotasRequest();
        try {
            ShowAutopilotUserChartsQuotasResponse response =
client.showAutopilotUserChartsQuotas(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

import os
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcce.v3.region.cce_region import CceRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcce.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.environ["CLOUD_SDK_AK"]
    sk = os.environ["CLOUD_SDK_SK"]
    projectId = "{project_id}"

    credentials = BasicCredentials(ak, sk, projectId)

    client = CceClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(CceRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowAutopilotUserChartsQuotasRequest()
        response = client.show_autopilot_user_charts_quotas(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    cce "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/cce/v3/region"
)
```



```
func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
    projectId := "{project_id}"

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        WithProjectId(projectId).
        Build()

    client := cce.NewCceClient(
        cce.CceClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowAutopilotUserChartsQuotasRequest{}
    response, err := client.ShowAutopilotUserChartsQuotas(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见**API Explorer**的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见**错误码**。

5 使用 Kubernetes API

Kubernetes API 说明

Kubernetes API是通过HTTP提供的基于资源 (RESTful) 的编程接口。它支持通过标准HTTP请求方法（POST、PUT、PATCH、DELETE、GET）进行查询、创建、更新和删除各类集群资源。

CCE支持通过多种方式使用原生**Kubernetes API**：

- **通过集群API Server调用Kubernetes API**：（推荐）直接连接集群API Server，适合大规模调用。
- **通过API网关调用Kubernetes API**：适合小规模调用场景，大规模调用时可能会触发API网关流控。

通过集群 API Server 调用 Kubernetes API

通过Kubernetes集群的API Server可以调用Kubernetes原生API。

步骤1 获取集群证书及API Server。

- 方式一：通过**获取集群证书**API获取，将返回的信息保存至kubeconfig.json文件中，并提取证书、私钥和API Server信息，命令如下。

```
# 获取证书并保存为client.crt
cat ./kubeconfig.json |grep client-certificate-data | awk -F '"' '{print $4}' | base64 -d > ./client.crt
# 获取私钥并保存为client.key
cat ./kubeconfig.json |grep client-key-data | awk -F '"' '{print $4}' | base64 -d > ./client.key
# 获取API Server
cat ./kubeconfig.json |grep server | awk -F '"' '{print $4}'
```

- 方式二：通过CCE控制台的“总览”页面查询API Server地址（内网地址或公网地址），并下载证书（client.crt和client.key文件）。

步骤2 使用集群证书调用Kubernetes原生API。

例如使用curl命令调用接口查看Pod信息，如下所示，其中*192.168.0.198:5443*为集群API Server地址。

```
curl --cacert ./ca.crt --cert ./client.crt --key ./client.key  https://192.168.0.198:5443/api/v1/namespaces/default/pods/
```

更多集群接口请参见**Kubernetes API**。

----结束

通过 API 网关调用 Kubernetes API

Kubernetes原生API，可以通过API网关调用，其URL格式为：**https://{clusterid}.Endpoint/uri**。其中**{clusterid}**为集群ID，**uri**为资源路径，也即API访问的路径。

表 5-1 URL 中的参数说明

参数	描述
{clusterid}	集群ID，创建集群后，调用**获取指定项目下的集群**接口获取。
Endpoint	Web服务入口点的URL，可以从终端节点（Endpoint）中获取。
uri	资源路径，也即API访问路径。从具体接口的URI模块获取，请参见Kubernetes API。

步骤1 获取集群所在区域的Token，获取方式请参见获取Token。

步骤2 获取集群ID。

- 方式一：通过获取集群信息API查询集群uid。
- 方式二：通过CCE控制台的“总览”页面查询。

步骤3 根据URL格式**https://{clusterid}.Endpoint/uri**，确定请求的URL。

- **{clusterid}**：通过**步骤2**获取。
- **Endpoint**：通过**地区和终端节点**获取。

例如CCE服务在“华东-上海一”区域的Endpoint为“cce.cn-east-3.myhuaweicloud.com”

- **uri**：根据需要调用的接口设置，例如需要创建一个Deployment，则请求方法为POST，接口uri为**/apis/apps/v1/namespaces/{namespace}/deployments**，其中{namespace}为集群命名空间名称，本示例为default。

  更多接口请参见**Kubernetes API**。

将上述参数根据URL格式**https://{clusterid}.Endpoint/uri**进行拼接。

则调用接口查看所有Pod信息的URL示例如下：

```
https://07da5*****.cce.cn-east-3.myhuaweicloud.com/apis/apps/v1/namespaces/default/deployments
```

步骤4 使用接口指定的请求方法，并设置请求Header参数。如果接口要求添加Body参数，可参考**Kubernetes API**添加接口对应的结构体。

例如使用curl命令调用创建Deployment接口，请求方法为POST，并添加对应的Body体。

本示例中使用nginx.json文件，创建一个名为nginx的Deployment负载，该工作负载使用nginx:latest镜像并包含两个Pod，每个Pod占用100mCPU、200Mi内存。

```
curl --location --request POST 'https://07da5*****.cce.cn-east-3.myhuaweicloud.com/apis/apps/v1/namespaces/default/deployments' \
--header 'Content-Type: application/json' \
--header 'X-Auth-Token: MIIWvw******' \
--data @nginx.json
```

请求中包含的Header参数如下：

表 5-2 请求 Header 参数

参数	是否必选	参数类型	描述
Content-Type	是	String	消息体的类型（格式），例如application/json
X-Auth-Token	是	String	调用接口的认证方式分为Token和AK/SK两种，如果您使用的Token方式，此参数为必填，请填写Token的值，获取方式请参见**获取token**。

nginx.json文件内容如下：

```
{
    "apiVersion": "apps/v1",
    "kind": "Deployment",
    "metadata": {
        "name": "nginx"
    },
    "spec": {
        "replicas": 2,
        "selector": {
            "matchLabels": {
                "app": "nginx"
            }
        },
        "template": {
            "metadata": {
```



```
            "labels": {
                "app": "nginx"
            }
        },
        "spec": {
            "containers": [
                {
                    "image": "nginx:latest",
                    "name": "container-0",
                    "resources": {
                        "limits": {
                            "cpu": "100m",
                            "memory": "200Mi"
                        },
                        "requests": {
                            "cpu": "100m",
                            "memory": "200Mi"
                        }
                    }
                }
            ],
            "imagePullSecrets": [
                {
                    "name": "default-secret"
                }
            ]
        }
    }
  }
}
```

----结束

相关文档

- **使用Kubernetes API访问集群**
- **Kubernetes官方SDK**（包括Go、Python、Java等语言）

语言	客户端库	样例程序
C	**github.com/kubernetes-client/c**	**浏览**
dotnet	**github.com/kubernetes-client/csharp**	**浏览**
Go	**github.com/kubernetes/client-go/**	**浏览**
Haskell	**github.com/kubernetes-client/haskell**	**浏览**
Java	**github.com/kubernetes-client/java**	**浏览**
JavaScript	**github.com/kubernetes-client/javascript**	**浏览**
Perl	**github.com/kubernetes-client/perl/**	**浏览**
Python	**github.com/kubernetes-client/python/**	**浏览**
Ruby	**github.com/kubernetes-client/ruby/**	**浏览**

6 权限和授权项

如果您需要对您所拥有的云容器引擎（CCE）进行精细的权限管理，您可以使用统一身份认证服务（Identity and Access Management，简称IAM），如果账号已经能满足您的要求，不需要创建独立的IAM用户，您可以跳过本章节，不影响您使用CCE服务的其它功能。

默认情况下，新建的IAM用户没有任何权限，您需要将其加入用户组，并给用户组授予策略或角色，才能使用户组中的用户获得相应的权限，这一过程称为授权。授权后，用户就可以基于已有的权限对云服务进行操作。关于策略的语法结构及示例，请参见**IAM权限管理说明**。

权限根据授权的精细程度，分为**角色**和**策略**。角色以服务为粒度，是IAM最初提供的一种根据用户的工作职能定义权限的粗粒度授权机制。策略以API接口为粒度进行权限拆分，授权更加精细，可以精确到某个操作、资源和条件，能够满足企业对权限最小化的安全管控要求。

说明

如果您要允许或是禁止某个接口的操作权限，请使用策略。

账号具备所有接口的调用权限，如果使用账号下的IAM用户发起API请求时，该IAM用户必须具备调用该接口所需的权限，否则，API请求将调用失败。每个接口所需要的权限，与各个接口所对应的授权项相对应，只有发起请求的用户被授予授权项所对应的策略，该用户才能成功调用该接口。例如，用户要调用接口来查询云服务器列表，那么这个IAM用户被授予的策略中必须包含允许“ecs:servers:list”的授权项，该接口才能调用成功。

IAM 支持的授权项

策略包含系统策略和自定义策略，如果系统策略不满足授权要求，管理员可以创建自定义策略，并通过给用户组授予自定义策略来进行精细的访问控制。策略支持的操作与API相对应，授权项列表说明如下：

- 权限：允许或拒绝某项操作。
- 对应API接口：自定义策略实际调用的API接口。
- 授权项：自定义策略中支持的Action，在自定义策略中的Action中写入授权项，可以实现授权项对应的权限功能。
- 依赖的授权项：部分Action存在对其他Action的依赖，需要将依赖的Action同时写入授权项，才能实现对应的权限功能。

- IAM项目(Project)/企业项目(Enterprise Project)：自定义策略的授权范围，包括IAM项目与企业项目。授权范围如果同时支持IAM项目和企业项目，表示此授权项对应的自定义策略，可以在IAM和企业管理两个服务中给用户组授权并生效。如果仅支持IAM项目，不支持企业项目，表示仅能在IAM中给用户组授权并生效，如果在企业管理中授权，则该自定义策略不生效。关于IAM项目与企业项目的区别，详情请参见：**IAM与企业管理的区别**。

说明

“√”表示支持，“x”表示暂不支持。

云容器引擎（CCE）支持的自定义策略授权项如下所示：

表 6-1 Cluster

权限	对应API接口	授权项（Action）	IAM项目(Project)	企业项目(Enterprise Project)
获取指定项目下的集群	GET /api/v3/projects/{project_id}/clusters	cce:cluster:list	√	√
获取指定的集群	GET /api/v3/projects/{project_id}/clusters/{cluster_id}	cce:cluster:get	√	√
创建集群	POST /api/v3/projects/{project_id}/clusters	cce:cluster:create	√	√
更新指定的集群	PUT /api/v3/projects/{project_id}/clusters/{cluster_id}	cce:cluster:update	√	√
删除集群	DELETE /api/v3/projects/{project_id}/clusters/{cluster_id}	cce:cluster:delete	√	√
升级集群	POST /api/v2/projects/:projectid/clusters/:clusterid/upgrade	cce:cluster:upgrade	√	√
唤醒集群	POST /api/v3/projects/{project_id}/clusters/{cluster_id}/operation/awake	cce:cluster:start	√	√
休眠集群	POST /api/v3/projects/{project_id}/clusters/{cluster_id}/operation/hibernate	cce:cluster:stop	√	√
变更集群规格	POST /api/v2/projects/{project_id}/clusters/:clusterid/resize	cce:cluster:resize	√	√

权限	对应API接口	授权项（Action）	IAM项目(Project)	企业项目(Enterprise Project)
获取集群证书	POST /api/v3/projects/{project_id}/clusters/{cluster_id}/clustercert	cce:cluster:get	√	√

表 6-2 Node

权限	对应API接口	授权项	IAM项目(Project)	企业项目(Enterprise Project)
获取集群下所有节点	GET /api/v3/projects/{project_id}/clusters/{cluster_id}/nodes	cce:node:list	√	√
获取指定的节点	GET /api/v3/projects/{project_id}/clusters/{cluster_id}/nodes/{node_id}	cce:node:get	√	√
创建节点	POST /api/v3/projects/{project_id}/clusters/{cluster_id}/nodes	cce:node:create	√	√ **说明** 使用企业项目授权创建节点需额外添加evs:quota:get的全局权限。
更新指定的节点	PUT /api/v3/projects/{project_id}/clusters/{cluster_id}/nodes/{node_id}	cce:node:update	√	√
删除节点	DELETE /api/v3/projects/{project_id}/clusters/{cluster_id}/nodes/{node_id}	cce:node:delete	√	√

表 6-3 Job

权限	对应API接口	授权项	IAM项目(Project)	企业项目(Enterprise Project)
获取任务信息	GET /api/v3/projects/{project_id}/jobs/{job_id}	cce:job:get	√	√
列出所有任务	GET /api/v2/projects/{project_id}/jobs	cce:job:list	√	√
删除所有任务或删除单个任务	DELETE /api/v2/projects/{project_id}/jobs DELETE /api/v2/projects/{project_id}/jobs/{job_id}	cce:job:delete	√	√

表 6-4 Nodepool

权限	对应API接口	授权项	IAM项目(Project)	企业项目(Enterprise Project)
获取集群下所有节点池	GET /api/v3/projects/{project_id}/clusters/{cluster_id}/nodepools	cce:nodepool:list	√	√
获取节点池	GET /api/v3/projects/{project_id}/clusters/{cluster_id}/nodepools/{nodepool_id}	cce:nodepool:get	√	√
创建节点池	POST /api/v3/projects/{project_id}/clusters/{cluster_id}/nodepools	cce:nodepool:create	√	√
更新节点池信息	PUT /api/v3/projects/{project_id}/clusters/{cluster_id}/nodepools/{nodepool_id}	cce:nodepool:update	√	√
删除节点池	DELETE /api/v3/projects/{project_id}/clusters/{cluster_id}/nodepools/{nodepool_id}	cce:nodepool:delete	√	√

表 6-5 Chart

权限	对应API接口	授权项	IAM项目(Project)	企业项目(Enterprise Project)
更新模板	PUT /v2/charts/{id}	cce:chart:update	√	×
上传模板	POST /v2/charts	cce:chart:upload	√	×
列出所有模板	GET /v2/charts	cce:chart:list	√	×
获取模板信息	GET /v2/charts/{id}	cce:chart:get	√	×
删除模板	DELETE /v2/charts/{id}	cce:chart:delete	√	×

表 6-6 Release

权限	对应API接口	授权项	IAM项目(Project)	企业项目(Enterprise Project)
更新升级模板实例	PUT /v2/releases/{name}	cce:release:update	√	√
列出所有模板实例	GET /v2/releases	cce:release:list	√	√
创建模板实例	POST /v2/releases	cce:release:create	√	√
获取模板实例信息	GET /v2/releases/{name}	cce:release:get	√	√
删除模板实例	DELETE /v2/releases/{name}	cce:release:delete	√	√

表 6-7 Storage

权限	对应API接口	授权项	IAM项目(Project)	企业项目(Enterprise Project)
创建PersistentVolumeClaim	POST /api/v1/namespaces/{namespace}/cloudpersistentvolumeclaims	cce:storage:create	√	√
删除PersistentVolumeClaim	DELETE /api/v1/namespaces/{namespace}/cloudpersistentvolumeclaims/{name}	cce:storage:delete	√	√
列出所有磁盘	GET /storage/api/v1/namespaces/{namespace}/listvolumes	cce:storage:list	√	√

表 6-8 Addon

权限	对应API接口	授权项	IAM项目(Project)	企业项目(Enterprise Project)
创建插件实例	POST /api/v3/addons	cce:addonInstance:create	√	√
获取插件实例	GET /api/v3/addons/{id}?cluster_id={cluster_id}	cce:addonInstance:get	√	√
列出所有插件实例	GET /api/v3/addons?cluster_id={cluster_id}	cce:addonInstance:list	√	√
删除插件实例	DELETE /api/v3/addons/{id}?cluster_id={cluster_id}	cce:addonInstance:delete	√	√
更新升级插件实例	PUT /api/v3/addons/{id}	cce:addonInstance:update	√	√

表 6-9 Quota

权限	对应API接口	授权项	IAM项目(Project)	企业项目(Enterprise Project)
查询配额详情	GET /api/v3/projects/{project_id}/quotas	cce:quota:get	√	√

7 附录

7.1 状态码

状态码如表7-1所示

表 7-1 状态码

状态码	编码	状态说明
100	Continue	继续请求。 这个临时响应用来通知客户端，它的部分请求已经被服务器接收，且仍未被拒绝。
101	Switching Protocols	切换协议。只能切换到更高级的协议。 例如，切换到HTTP的新版本协议。
201	Created	创建类的请求完全成功。
202	Accepted	已经接受请求，但未处理完成。
203	Non-Authoritative Information	非授权信息，请求成功。
204	NoContent	请求完全成功，同时HTTP响应不包含响应体。 在响应OPTIONS方法的HTTP请求时返回此状态码。
205	Reset Content	重置内容，服务器处理成功。
206	Partial Content	服务器成功处理了部分GET请求。
300	Multiple Choices	多种选择。请求的资源可包括多个位置，相应可返回一个资源特征与地址的列表用于用户终端（例如：浏览器）选择。
301	Moved Permanently	永久移动，请求的资源已被永久的移动到新的URI，返回信息会包括新的URI。

状态码	编码	状态说明
302	Found	资源被临时移动。
303	See Other	查看其它地址。 使用GET和POST请求查看。
304	Not Modified	所请求的资源未修改，服务器返回此状态码时，不会返回任何资源。
305	Use Proxy	所请求的资源必须通过代理访问。
306	Unused	已经被废弃的HTTP状态码。
400	BadRequest	非法请求。 建议直接修改该请求，不要重试该请求。
401	Unauthorized	在客户端提供认证信息后，返回该状态码，表明服务端指出客户端所提供的认证信息不正确或非法。
402	Payment Required	保留请求。
403	Forbidden	请求被拒绝访问。 返回该状态码，表明请求能够到达服务端，且服务端能够理解用户请求，但是拒绝做更多的事情，因为该请求被设置为拒绝访问，建议直接修改该请求，不要重试该请求。
404	NotFound	所请求的资源不存在。 建议直接修改该请求，不要重试该请求。
405	MethodNotAllowed	请求中带有该资源不支持的方法。 建议直接修改该请求，不要重试该请求。
406	Not Acceptable	服务器无法根据客户端请求的内容特性完成请求。
407	Proxy Authentication Required	请求要求代理的身份认证，与401类似，但请求者应当使用代理进行授权。
408	Request Time-out	服务器等候请求时发生超时。 客户端可以随时再次提交该请求而无需进行任何更改。
409	Conflict	服务器在完成请求时发生冲突。 返回该状态码，表明客户端尝试创建的资源已经存在，或者由于冲突请求的更新操作不能被完成。
410	Gone	客户端请求的资源已经不存在。 返回该状态码，表明请求的资源已被永久删除。

状态码	编码	状态说明
411	Length Required	服务器无法处理客户端发送的不带Content-Length的请求信息。
412	Precondition Failed	未满足前提条件，服务器未满足请求者在请求中设置的其中一个前提条件。
413	Request Entity Too Large	由于请求的实体过大，服务器无法处理，因此拒绝请求。为防止客户端的连续请求，服务器可能会关闭连接。如果只是服务器暂时无法处理，则会包含一个Retry-After的响应信息。
414	Request-URI Too Large	请求的URI过长（URI通常为网址），服务器无法处理。
415	Unsupported Media Type	服务器无法处理请求附带的媒体格式。
416	Requested range not satisfiable	客户端请求的范围无效。
417	Expectation Failed	服务器无法满足Expect的请求头信息。
422	UnprocessableEntity	请求格式正确，但是由于含有语义错误，无法响应。
429	TooManyRequests	表明请求超出了客户端访问频率的限制或者服务端接收到多于它能处理的请求。建议客户端读取相应的Retry-After首部，然后等待该首部指出的时间后再重试。
500	InternalServerError	表明服务端能被请求访问到，但是不能理解用户的请求。
501	Not Implemented	服务器不支持请求的功能，无法完成请求。
502	Bad Gateway	充当网关或代理的服务器，从远端服务器接收到了一个无效的请求。
503	ServiceUnavailable	被请求的服务无效。 建议直接修改该请求，不要重试该请求。
504	ServerTimeout	请求在给定的时间内无法完成。客户端仅在为请求指定超时（Timeout）参数时会得到该响应。
505	HTTP Version not supported	服务器不支持请求的HTTP协议的版本，无法完成处理。

7.2 错误码

调用接口出错后，将不会返回结果数据。调用方可根据每个接口对应的错误码来定位错误原因。 当调用出错时，HTTP 请求返回一个 4xx 或 5xx 的 HTTP 状态码。返回的

消息体中是具体的错误代码及错误信息。在调用方找不到错误原因时，可以联系客服，并提供错误码，以便尽快帮您解决问题。

错误响应 Body 体格式说明

当接口调用出错时，会返回错误码及错误信息说明，错误响应的Body体格式如下所示。

```
{
   "errorMessage": "The format of message is error",
   "errorCode": "CCE.01400001"
}
```

其中，errorCode表示错误码，errorMessage表示错误描述信息。

错误码说明

当您调用API时，如果遇到“APIGW”开头的错误码，请参见API网关错误码进行处理。

状态码	错误码	错误信息	描述	处理措施
400	CCE.01400001	Invalid request.	请求体不合法。	请参考返回的message和CCE接口文档修改请求体，或联系技术支持。
400	CCE.01400002	Subnet not found in the VPC.	未在VPC中找到子网。	请确认请求体中的子网是否在对应VPC下。
400	CCE.01400003	IPv6 not supported for the subnet.	子网不支持ipv6。	请使用支持ipv6的子网。
400	CCE.01400004	No available flavors for master nodes.	Master节点无可用规格。	请更换其他可用的集群规格，或联系技术支持。
400	CCE.01400005	Container network CIDR blocks conflict.	容器网络网段冲突。	请参考返回的message检查容器网段。
400	CCE.01400006	Content type not supported.	Content type不合法。	请参考CCE接口文档使用支持的Content type。
400	CCE.01400007	Insufficient cluster quota.	集群配额不足。	请提交工单增加集群配额。
400	CCE.01400008	Insufficient server quota	ECS配额不足。	请提交工单增加ECS配额。
400	CCE.01400009	Insufficient CPU quota.	ECS CPU配额不足。	请提交工单增加ECS CPU配额。

状态码	错误码	错误信息	描述	处理措施
400	CCE.01400010	Insufficient memory quota.	ECS 内存配额不足。	请提交工单增加ECS内存配额。
400	CCE.01400011	Insufficient security group quota.	安全组配额不足。	请提交工单增加安全组配额。
400	CCE.01400012	Insufficient EIP quota.	EIP配额不足。	请提交工单增加EIP配额。
400	CCE.01400013	Insufficient volume quota.	磁盘配额不足。	请参考返回的message，提交工单增加相应的磁盘配额。
400	CCE.01400014	Excessive nodes in the cluster.	节点数超出集群规模限制。	请提交工单申请变更集群规格。
400	CCE.01400015	Version not supported.	不受支持的集群版本。	请参考返回的message，创建支持的集群版本。
400	CCE.01400016	Current cluster type does not support this node flavor.	当前集群类型不支持此节点规格。	请参考返回的message，使用正确的节点规格。
400	CCE.01400017	No available container CIDR block found.	没有找到可用的容器网段。	请参考返回的message，使用正确的容器网段。
400	CCE.01400018	This type of OS cannot be created in this CCE version.	当前CCE版本不支持创建该类型的操作系统。	请参考返回的message，使用支持的操作系统。
400	CCE.01400019	Insufficient resource tenant quota.	资源租户配额不足。	请参考返回的message，或联系技术支持。
400	CCE.01400020	Insufficient VPC quota.	VPC配额不足。	请参考返回的message，或联系技术支持。
400	CCE.01400021	No available flavors for nodes.	节点无可用规格。	请更换其他可用的节点规格，或联系技术支持。

状态码	错误码	错误信息	描述	处理措施
400	CCE.01400022	No available node volumes for nodes.	节点无可用EVS卷规格。	请更换其他可用的EVS卷规格，或联系技术支持。
400	CCE.02400001	Invalid request.	请求体不合法。	请参考返回的message和CCE接口文档修改请求体，或联系技术支持。
400	CCE.03400001	Invalid request.	请求体不合法。	请参考返回的message和CCE接口文档修改请求体，或联系技术支持。
400	CCE.03400002	Missing access key.	缺少Access key。	请确认安装或升级的存储插件版本正确，或联系技术支持。
401	CCE.01401001	Authorization failed.	认证失败。	请参考返回的message，或联系技术支持。
401	CCE.02401001	Authorization failed.	认证失败。	请参考返回的message，或联系技术支持。
401	CCE.03401001	Authorization failed.	认证失败。	请参考返回的message，或联系技术支持。
403	CCE.01403001	Forbidden.	禁止访问。	请参考返回的message，或联系技术支持。
403	CCE.02403001	Forbidden.	禁止访问。	请参考返回的message，或联系技术支持。
403	CCE.03403001	Forbidden.	禁止访问。	请参考返回的message，或联系技术支持。
404	CCE.01404001	Resource not found.	未找到资源。	请确认要访问的资源是否已被删除。
404	CCE.02404001	Resource not found.	未找到资源。	请确认要访问的资源是否已被删除。
404	CCE.03404001	Resource not found.	未找到资源。	请确认要访问的资源是否已被删除。

状态码	错误码	错误信息	描述	处理措施
409	CCE.01409001	The resource already exists.	资源已存在。	请先删除资源后，再进行创建。
409	CCE.01409002	Resource updated with out-of-date version.	要更新的资源版本已过期。	请确认要更新的资源为版本为最新，或联系技术支持。
409	CCE.02409001	The resource already exists.	资源已存在。	请先删除资源后，再进行创建。
409	CCE.03409001	Addon instance has installed.	插件实例已安装。	请先删除插件实例，再进行安装。
429	CCE.01429002	Resource locked by other requests.	资源被其他请求锁定。	请参考返回的message，或联系技术支持。
429	CCE.02429001	The throttling threshold has been reached.	已达到最大请求数量限制。	请减少发送请求的频率，或联系技术支持。
500	CCE.01500001	Internal error.	内部错误。	请参考返回的message，或联系技术支持。
500	CCE.02500001	Internal error.	内部错误。	请参考返回的message，或联系技术支持。
500	CCE.03500001	Internal error.	内部错误。	请参考返回的message，或联系技术支持。

7.3 获取项目 ID

操作场景

在调用接口的时候，部分URL中需要填入项目ID，所以需要获取到项目ID。有如下两种获取方式：

- **调用API获取项目ID**
- **从控制台获取项目ID**

调用 API 获取项目 ID

项目ID可以通过调用**查询指定条件下的项目列表**API获取。

获取项目ID的接口为“GET https://{Endpoint}/v3/projects”，其中{Endpoint}为IAM的终端节点，可以从**终端节点（Endpoint）**获取。接口的认证鉴权请参见**认证鉴权**。

响应示例如下，其中projects下的“id”即为项目ID。

```
{
    "projects": [
        {
            "domain_id": "65382450e8f64ac0870cd180d14e684b",
            "is_domain": false,
            "parent_id": "65382450e8f64ac0870cd180d14e684b",
            "name": "project_name",
            "description": "",
            "links": {
                "next": null,
                "previous": null,
                "self": "https://www.example.com/v3/projects/a4a5d4098fb4474fa22cd05f897d6b99"
            },
            "id": "a4a5d4098fb4474fa22cd05f897d6b99",
            "enabled": true
        }
    ],
    "links": {
        "next": null,
        "previous": null,
        "self": "https://www.example.com/v3/projects"
    }
}
```

从控制台获取项目 ID

从控制台获取项目ID的步骤如下：

1. 登录管理控制台。
2. 鼠标悬停在右上角的用户名，选择下拉列表中的“我的凭证”。
   在“API凭证”页面的项目列表中查看项目ID。

图 7-1 查看项目 ID

7.4 获取账号 ID

在调用接口的时候，部分URL中需要填入账号ID（domain-id），所以需要先在管理控制台上获取到账号ID。账号ID获取步骤如下：

1. 注册并登录管理控制台。
2. 单击用户名，在下拉列表中单击“我的凭证”。
   在“API凭证”页面的项目列表中查看账号ID。

图 7-2 获取账号 ID

7.5 如何获取接口 URI 中参数

项目 ID（project_id）

project_id即项目ID，可以通过控制台或API接口获取，具体请参见**获取项目ID**。

集群 ID（cluster_id）

步骤1 登录CCE控制台，在左侧导航栏中选择“集群管理”。

步骤2 单击所创建集群的名称，进入集群详情页面，获取集群ID。

图 7-3 获取 cluster_id

----结束

节点 ID（node_id）

步骤1 登录CCE控制台，在左侧导航栏中选择“集群管理”。

步骤2 单击所创建集群的名称，并在左侧选择“节点管理”，将光标移动到节点名称上，查看对应的节点ID。

图 7-4 获取 node_id

----结束

任务 ID（job_id）

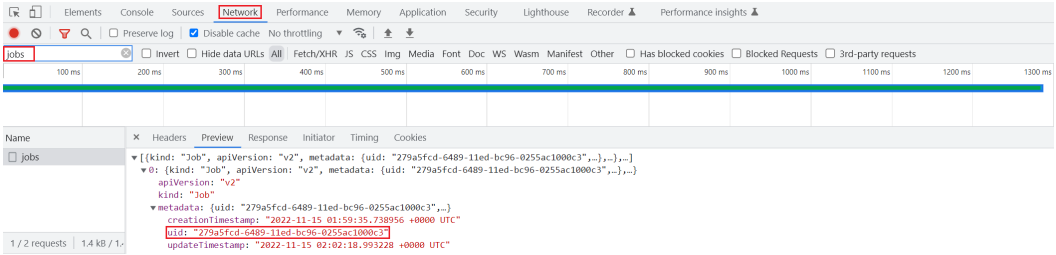
步骤1 登录CCE控制台，在左侧导航栏中选择“集群管理”。此处以集群管理为例，获取正在创建中的集群job_id。

图 7-5 创建集群

步骤2 获取job_id。

1. 以Chrome浏览器为例，F12打开浏览器Console，单击“Network”。
2. 单击CCE控制台中的“操作记录”，查看集群操作记录详情。
3. 在浏览器Console的“Filter”栏里输入“jobs”，过滤出jobs列表，单击该名称并选择“Preview”页签，在左侧列表选择本次操作对应的job，其中uid字段即为job的uid。

图 7-6 获取 job_id



----结束

7.6 创建 VPC 和子网

背景信息

在创建集群之前，您需要创建虚拟私有云（VPC），为CCE服务提供一个安全、隔离的网络环境。

如果用户已有VPC，可重复使用，不需多次创建。

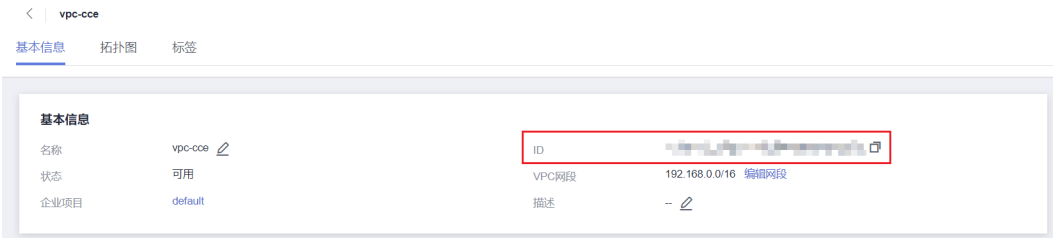
创建 VPC

步骤1 登录管理控制台，选择“网络 > 虚拟私有云 VPC”。

步骤2 在虚拟私有云控制台，单击右上角的“创建虚拟私有云”，按照提示完成创建。

步骤3 创建完成后返回虚拟私有云列表，单击创建的VPC名称，在详情页获取VPC的ID，后续**创建集群**时需要使用。

图 7-7 获取 VPC 的 ID



----结束

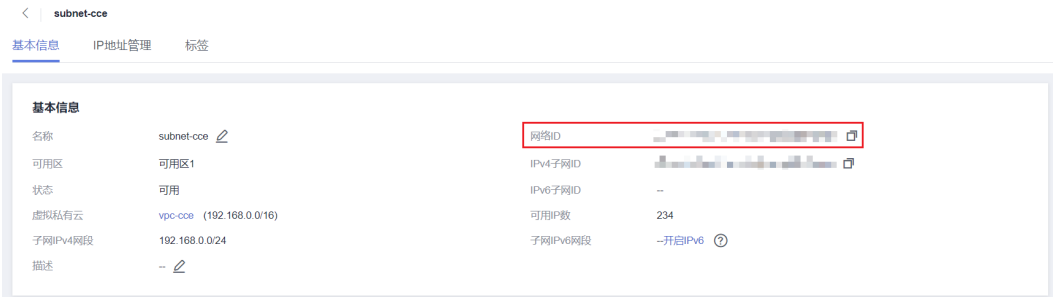
创建子网

步骤1 登录管理控制台，选择“网络 > 虚拟私有云 VPC”。

步骤2 在“虚拟私有云”列表页面，单击左侧导航栏中“虚拟私有云 > 子网”，单击右上角“创建子网”。

步骤3 按照页面提示完成子网创建，并单击子网的名称，获取子网的“网络ID”，后续**创建集群**时需要使用。

图 7-8 获取子网的网络 ID



----结束

7.7 创建密钥对

背景信息

在创建集群之前，您需要创建密钥对，用于登录工作节点时的身份验证。

如果用户已有密钥对，可重复使用，不需多次创建。

操作步骤

步骤1 登录管理控制台，选择“计算 > 弹性云服务器”。

步骤2 在左侧导航树中，选择“密钥对”。

步骤3 单击“创建密钥对”，并按照提示完成创建，详情请参见**密钥对**。

步骤4 创建完成后，系统生成密钥文件，自动保存在系统默认目录下。

----结束

7.8 通过控制台可视化生成 API 参数

在使用API创建集群或节点时，如果请求中的API参数组合不正确，将会导致接口调用失败。您可以通过控制台可视化生成API参数，根据选项配置自动生成正确的参数组合。

生成创建集群的 API 参数

步骤1 登录**CCE控制台**。

步骤2 在“集群管理”页面右上角单击“购买集群”。

步骤3 参考**购买Autopilot集群**，根据自身需求配置集群参数。

步骤4 完成配置后，在“确认配置”页面，查看根据配置生成的API数据，您可以通过下载或复制进行使用。

图 7-9 生成创建集群的 API 参数

步骤5 使用生成的API数据作为Body体，调用创建集群接口，详情请参见**创建集群**。

----结束