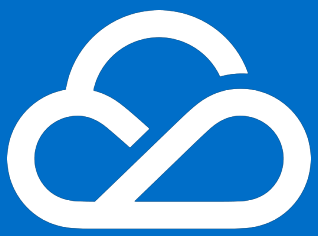
腾讯云容器服务

应用模板

产品文档



腾讯云

【版权声明】

©2013-2017 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

【商标声明】

腾讯云

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

文档声明 2
应用模板 4
应用模板概述 4
应用模板示例-nginx单服务应用 5
应用模板示例-guestbook应用 12
创建应用模板 21
删除应用模板 29
更新应用模板 30
查看应用模板 33
语法说明 - 基础语法 35
语法说明 - 扩展语法 43
语法说明 - 变量设置 47
语法说明 - 限制说明 51
应用模板内容操作指引 55

版权所有：腾讯云计算（北京）有限责任公司

应用模板

应用模板概述

在腾讯云容器服务中可以将多个服务的部署信息，通过YAML形式的描述文本保存到应用模板中。通过应用模板和
不同环境
下的配置，可以快速的在不同环境中部署应用中的服务。更多关于应用管理的内容可以参考应用管理概述。

应用模板主要操作

- 创建应用模板
- 删除应用模板
- 更新应用模板
- 查看应用模板

编排语法能力说明

应用模板中依赖于编排语法对服务的部署信息进行说明。编排语法本质对服务部署信息描述的一套规则，用户按照这个规则来描述服务的部署信息，编排引擎对描述信息进行解析后，按照解析得到的参数对服务进行部署。

腾讯云容器服务应用模板的编排语法底层支持kubernetes的原生编排语法，关于kubernetes的编排语的详细说明可以参考基础语法。在kubernetes的编排语
法的基础上支持变量替换，变量替换的详细说明可以
参考变量替换。同时考虑到容器服务平台本身的能力，提供容器服务平台相关的扩展语法，具体的关于扩展语法的说明可以参考扩展语法。
出于
功能稳定
闭环的考虑，容器平台对kubernetes的编排语法做了一定的限制。具体的限制说明可以参考限制说明。

应用模板示例-nginx单服务应用

nginx是最基础的一个服务，在应用中也算是最基础的一个应用，有点类似于"Hello World"示例。应用模板的示例，我们从"Hello World"的nginx应用开始。

步骤一: 新建应用模板

在应用模板列表中，点击新建按钮。

步骤二: 应用模板编辑

2.1 添加nginx服务**

应用模板中增加服务，是通过导入服务相对应的模板内容实现的。导入服务模板内容包括两种可选的方式: 1. 从控制台导入 2. YAML文件导入

> 注意：
> 可以根据场景选择使用其中的任意一种方式

更多关于服务导入的说明可以参考[应用模板内容操作指引][4]。

导入方法1： 控制台导入服务

点击UI导入服务按钮，会弹出导入服务的控制台，在导入服务控制台填写相应参数。

在

nginx

服务中设置的参数包括：

设置服务的基本信息：

1. 填写服务名称

   nginx

2. 填写服务描述

   nginx服务

设置服务的数据卷信息：

未使用数据盘，无。

设置镜像参数：

1. 在设置容器运行参数中的镜像参数：
   容器名称设置为

   nginx

   镜像名称设置为

   nginx

   版本号选择为

   latest

设置服务的实例数：

1. 服务的实例数设置为1

设置服务的访问方式：

1. 服务的访问方式设置为集群内访问
2. 服务的访问端口：容器端口和服务端口都设置成80

更多关于参数设置内容可以参考 [服务创建][7]操作的相关文档。

填写参数后，点击导入服务的

完成

按钮，控制台自动导入服务的模板内容。

导入方法2： YAML文件导入

如果已经存在服务对应的模板内容的YAML文件，可以直接将模板内容导入到编辑框中。具体的步骤如下：

步骤一 创建对应的服务：
点击"新增空服务"号按钮，新增一个服务。服务名称设置为

nginx

。

步骤二 导入模板内容：

可以将下面YAML文件中的内容，直接拷贝到编辑框中，导入服务的模板内容。

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
  description: nginx服务
  creationTimestamp: null
  name: nginx
  namespace: '{{.NAMESPACE}}'
spec:
  replicas: 1
  revisionHistoryLimit: 5
  selector: {}
  strategy: {}
  template:
  metadata:
  creationTimestamp: null
  spec:
  containers:
  - image: nginx:latest
  imagePullPolicy: Always
  name: nginx
```

```
  resources:
  requests:
  cpu: 200m
  securityContext:
  privileged: false
  serviceAccountName: ""
  volumes: null
status: {}
---
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  name: nginx
  namespace: '{{.NAMESPACE}}'
spec:
  ports:
  - name: tcp-80-80-ogxxh
  nodePort: 0
  port: 80
  protocol: TCP
  targetPort: 80
  selector: {}
  type: LoadBalancer
status:
  loadBalancer: {}
```

在示例中，使用了

NAMESPACE

将

namespace

填写完之后，如下图所示：

步骤三: 完成应用模板编辑，并查看

在步骤二中，完成了应用模板的编辑。点击

完成

按钮，保存应用模板。这样应用模板就创建完成，可以在应用模板列表查看。

接下来可以使

用创建的模板，进行应用服务部署。关于如何使用应用模板进行应用部署可以参考创建应用。关于

nginx

这个应用模板具体部署应用的过程可以参考应用模板示例-Nginx单服务应用。

版权所有：腾讯云计算（北京）有限责任公司

应用模板示例-guestbook应用

Guestbook留言板是一个比较典型的web应用服务，由一个frontend前端服务和redis-master和redis-slave两个后端存储服务组成。用户通过web前端提交数据，写入到redis-master上，然后通过读取同步到redis-slave上的数据展示给用户。

为了实现快速的在不同集群或者集群的不同namespace中部署Guestbook应用，可以先将Gustbook部署相关的配置保存到应用模板中，然后使用应用模板快速部署对应的应用。

本示例将介绍如何将创建Gustbook的应用模板。

步骤一: 创建应用模板

在应用模板列表中，点击新建按钮。

步骤二: 编辑应用模板

2.1 填写应用模板名称

2.2 创建frontend服务

(1) 点击图中

新增空服务

按钮，新增一个服务。服务名称设置为frontend。

(2) 在模板内容的编辑框中，填写

frontend

服务的模板内容。可以直接拷贝下面的内容导编辑框中。

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: frontend
  namespace: {{.NAMESPACE}}
spec:
  replicas: {{.FRONTEND_REPLICAS}}
  template:
  spec:
  containers:
  - env:
  - name: GET_HOSTS_FROM
  value: dns
  image: ccr.ccs.tencentyun.com/library/gb-frontend:{{.FRONTEND_VERSION}}
  imagePullPolicy: Always
  name: php-redis
  resources:
  requests:
  cpu: 100m
  securityContext:
  privileged: false
  dnsPolicy: ClusterFirst
  restartPolicy: Always
---
apiVersion: v1
kind: Service
metadata:
  name: frontend
  namespace: {{.NAMESPACE}}
spec:
  ports:
  - name: tcp-80-80-fwdy6
  port: 80
  protocol: TCP
```

```
  targetPort: 80
  sessionAffinity: None
  type: LoadBalancer
```

2.3 创建redis-master服务

(1) 和

frontend

服务创建过程类似，新建一个

redis-master

服务。

(2) 填写

redis-master

服务的模板内容。

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: redis-master
  namespace: {{.NAMESPACE}}
spec:
  template:
  spec:
  containers:
  - image: ccr.ccs.tencentyun.com/library/redis:{{.REDIS_MASTER_VERSION}}
  imagePullPolicy: Always
```

```
  name: master
  resources:
  limits:
  memory: 100Mi
  requests:
  cpu: 100m
  memory: 100Mi
  securityContext:
  privileged: false
  dnsPolicy: ClusterFirst
  restartPolicy: Always
---
apiVersion: v1
kind: Service
metadata:
  name: redis-master
  namespace: {{.NAMESPACE}}
spec:
  ports:
  - name: tcp-6379-6379-6d3d9
  port: 6379
  protocol: TCP
  targetPort: 6379
  sessionAffinity: None
  type: ClusterIP
```

2.4 创建redis-slave服务

(1) 和

frontend

服务创建过程类似，新建一个

redis-slave

服务。

(2) 填写

redis-slave

服务的模板内容。

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: redis-slave
  namespace: {{.NAMESPACE}}
spec:
  template:
  spec:
  containers:
  - env:
  - name: GET_HOSTS_FROM
  value: dns
  image: ccr.ccs.tencentyun.com/library/gb-redisslave :{{.REDIS_SLAVE_VERSION}}
  imagePullPolicy: Always
  name: slave
  resources:
  limits:
  memory: 100Mi
  requests:
  cpu: 100m
  memory: 100Mi
  securityContext:
  privileged: false
  dnsPolicy: ClusterFirst
```

版权所有：腾讯云计算（北京）有限责任公司

```
  restartPolicy: Always
---
apiVersion: v1
kind: Service
metadata:
  name: redis-slave
  namespace: {{.NAMESPACE}}
spec:
  externalName: ""
  ports:
  - name: tcp-6379-6379-av5fd
  port: 6379
  protocol: TCP
  targetPort: 6379
  sessionAffinity: None
  type: ClusterIP
```

创建后的服务如下图所示：

2.5 导出配置项，并填写配置项内容

在应用模板汇总使用了多个变量，需要在配置项中为变量设置默认值。具体的做法如下。

填写配置中配置项的内容。在本示例中，配置项的默认值如下。(可以根据需要进行修改)

NAMESPACE: default

FRONTEND_REPLICAS: 2

FRONTEND_VERSION: v4

REDIS_MASTER_VERSION: e2e

REDIS_SLAVE_VERSION: v1

步骤三: 完成应用模板编辑，并查看

在步骤二中，完成了应用模板的编辑。点击

完成

按钮，保存应用模板。

这样应用模板就创建完成，可以在应用模板列表查看。

接下来可以使

用创建的模板，进行应用服务部署。关于如何使用应用模板进行应用部署可以参考创建应用。关于

Guestbook

这个应用模板具体部署应用的过程可以参考应用模板示例-Guestbook应用。

创建应用模板

□## 新建应用模板

登录 腾讯云容器服务控制台 。在应用管理->应用模板页面点击新建

应用模板增加服务

应用模板中增加服务，是通过导入服务相对应的模板内容实现的。导入服务模板内容包括两种可选的方式: 1. 从控制台导入 2. YAML文件导入

> 注意：
> 可以根据场景选择使用其中的任意一种方式

更多关于导入模板内容的说明，可以参考应用模板内容操作指引

导入方法1： 控制台导入服务：

点击导入服务按钮，弹出服务导入的控制台

导入后结果：

模板内容　模板可以通过从UI导入服务或新增空服务并手动编写来创建多个服务的YAML描述，详情可查看 应用模板操作指引

服务名	操作	内容
mariadb	删除	

```
37          cbsDiskId: '{{.ReleaseCBS_mariadb_vol}}'
38          fsType: ext4
39  status: {}
40  ---
41  apiVersion: v1
42  kind: Service
43  metadata:
44    creationTimestamp: null
45    name: mariadb
46    namespace: '{{.NAMESPACE}}'
47  spec:
48    ports:
49    - name: tcp-3306-3306-nh6kj
50      nodePort: 0
51      port: 3306
52      protocol: TCP
53      targetPort: 3306
54    selector: {}
55    type: ClusterIP
```

新增空服务　从UI导入服务

默认配置项　以上服务配置中，若存在自定义变量，可以手动输入为变量赋值，也可使用已有的配置项来为变量赋值，更多参见 变量设置说明

注：变量替换类型的配置项用于字符串替换，文件挂载类型的配置项用于将指定内容挂载到容器

变量名	变量值	类型
NAMESPACE	default	变量替换
ROOT_PASSWORD_VALUE		变量替换
ReleaseCBS_mariadb_vol		变量替换

使用已有配置项赋值

在导入服务控制台，填写对应的参数后，点击确认，则自动生成服务的模板内容。更多关于控制台导入模板内容的说明，可以参考应用模板内容操作指引-控制台导入服务。

导入方法2： YAML文件导入服务：

点击服务添加按钮，添加服务

将在服务编辑中，编辑服务的内容。可以直接拷贝模板内容到服务编辑框。下面是一个nginx示例的模板内容。更多关于YAML文件导入模板内容的说明，可以参考应用模板内容操作指引-YAML文件导入模板内容。

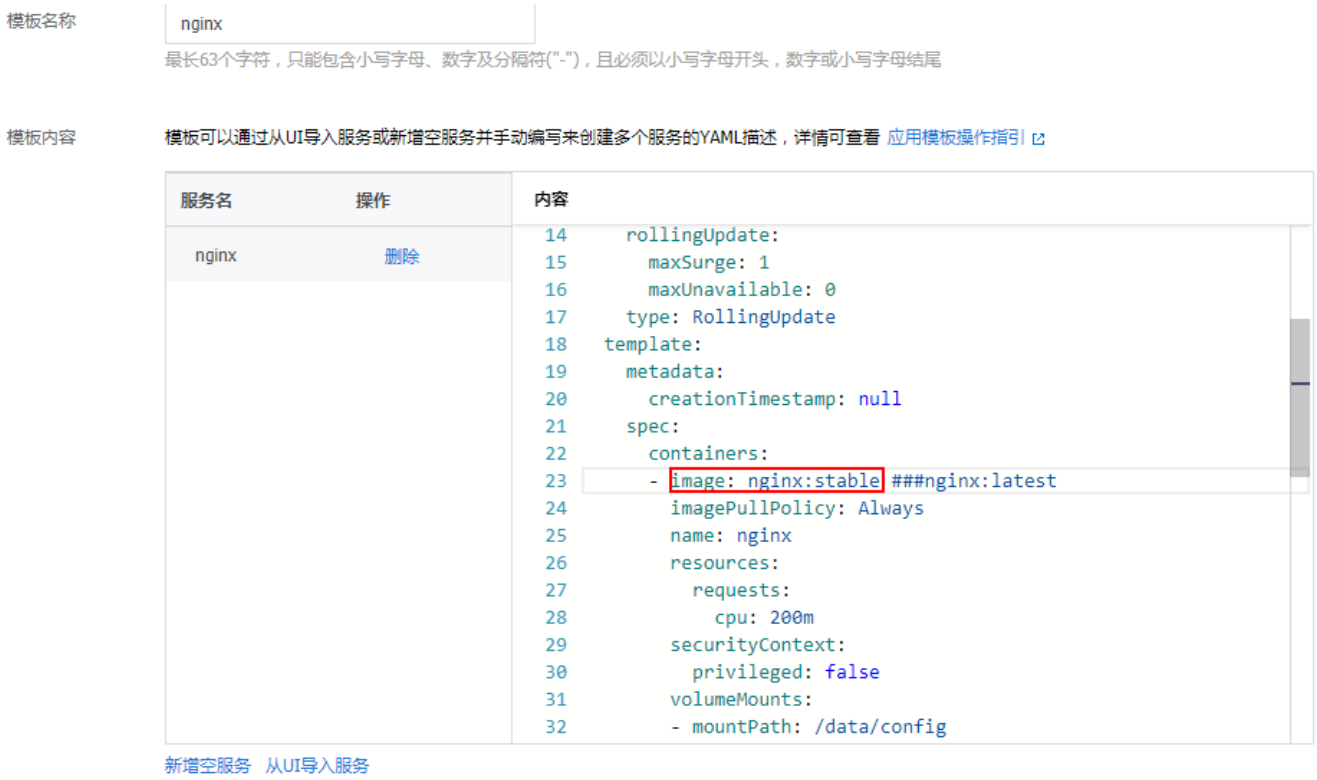
```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
  description: nginx服务
  creationTimestamp: null
  name: nginx
  namespace: '{{.NAMESPACE}}'
spec:
```

```
  replicas: 1
  revisionHistoryLimit: 5
  selector: {}
  strategy: {}
  template:
  metadata:
  creationTimestamp: null
  spec:
  containers:
  - image: nginx:latest
  imagePullPolicy: Always
  name: nginx
  resources:
  requests:
  cpu: 200m
  securityContext:
  privileged: false
  serviceAccountName: ""
  volumes: null
status: {}
---
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  name: nginx
  namespace: '{{.NAMESPACE}}'
spec:
  ports:
  - name: tcp-80-80-ai6zd
  nodePort: 0
  port: 80
  protocol: TCP
  targetPort: 80
```

```
  selector: {}
  type: LoadBalancer
status:
  loadBalancer: {}
```

应用模板内容编辑

在应用模板内容的编辑框中，可以编辑和修改对应服务的内容。例如：可以在模板内容中修改镜像的版本，从"latest"修改为"stable"(镜像的版本从最新版修改为稳定版)。



模板内容满足一定的编排语法规则，更多关于语法规则的说明可以参考编排语法能力说明。

应用模板配置项设置

在模板内容中，如果有些参数在不同环境下是不相同的，或者需要进行频繁的修改。可以将这部分参数转换成一个可配置的变量。例如在上面的示例中，可以将镜像版本设置为变量

IMAGE_VERSION

，在配置项中设置

IMAGE_VERSION

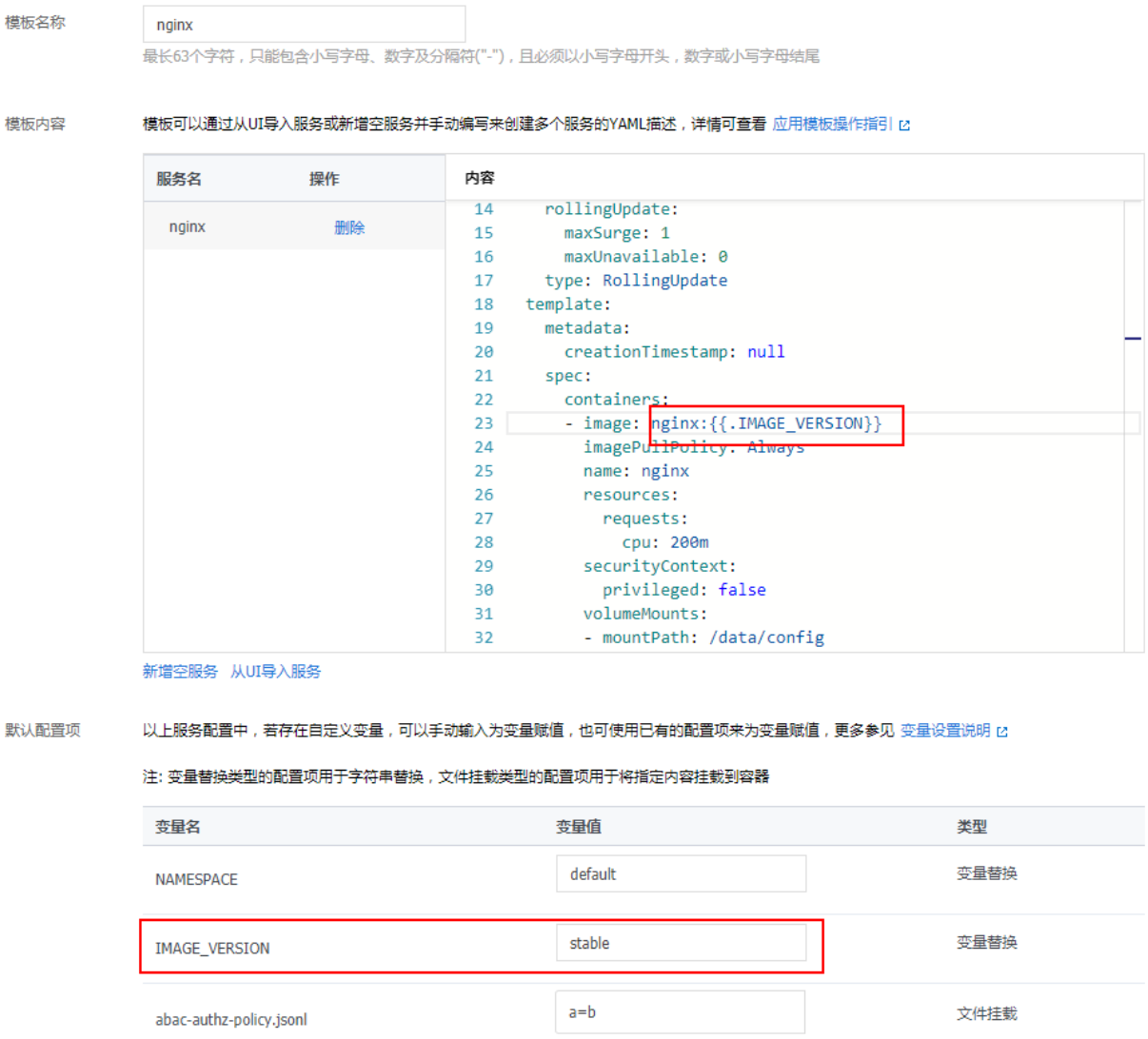
的值为"stable"。

具体的操作步骤如下:

步骤一： 编辑模板内容，将参数用变量替换
如下图所示将模板内容中镜像的版本这个参数使用

IMAGE_VERSION

变量替换。变量形式为"{{.xxxx}}"。更多关于变量设置的说明，可以参考变量设置。



步骤二： 设置配置项中变量的值

在配置项中设置

IMAGE_VERSION

的值为"stable"。这样在使用应用模板部署应用时，如果使用应用模板中的默认配置项，则会在部署时使用"stable"作为镜像的版本。当然也可以根据不同环境的需求，在配置项中将

IMAGE_VERSION

变量设置成不同的值，从而实现在不同环境下的使用不同的镜像版本进行部署。

应用模板保存

在写好应用模板的内容和默认配置项后，可以点击

完成

按钮，保存新创建的应用模板。

查看创建的模板

在应用模板列表页面，可查看到刚创建的应用模板。


容器服务
概览
应用中心
集群
应用
服务
Ingress
交付中心
镜像仓库
应用模板仓库
我的模板
模板市场
应用模板
+ 新建
模板ID/模板名称
服务数量
tp-7xq0950i
wordpress
2
tp-el5uirgo
singlenginx
1
tp-fdbnzixs
nginx
1
tp-eh6ia6js
php-nginx
1
tp-mev3xoy2
elk-devin
4
tp-51uxeyx8
d2048
1

删除应用模板

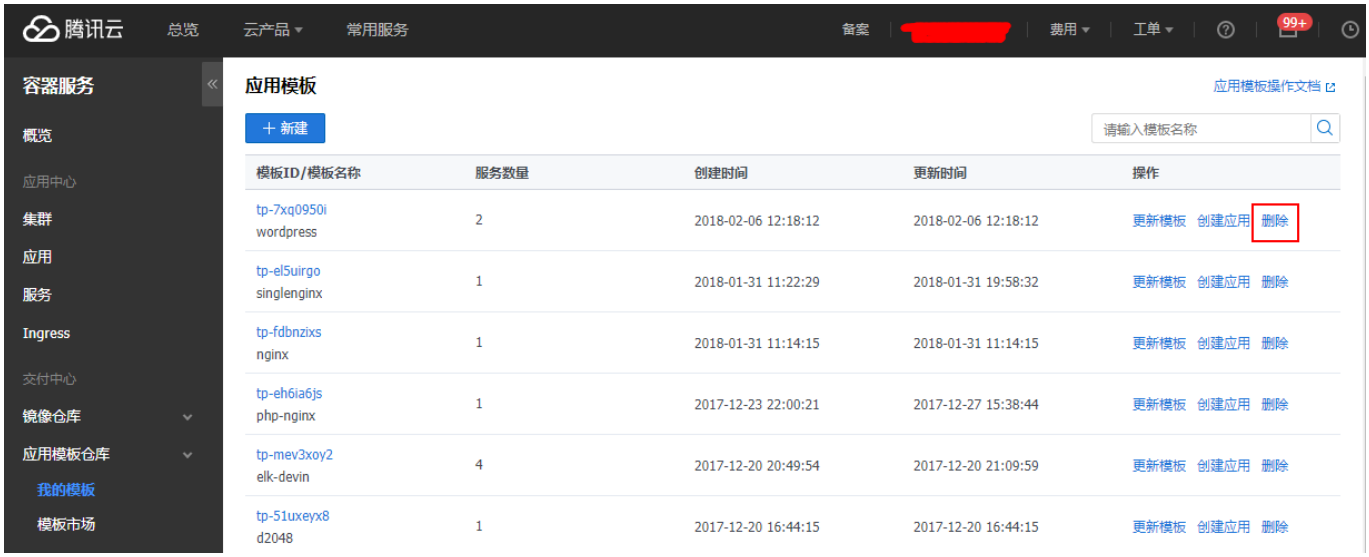
□## 删除应用模板

登录 腾讯云容器服务控制台

。在应用管理->应用模板页面查看应用模板列表。点击右侧的删除按钮，删除指定的应用模板。

注意：

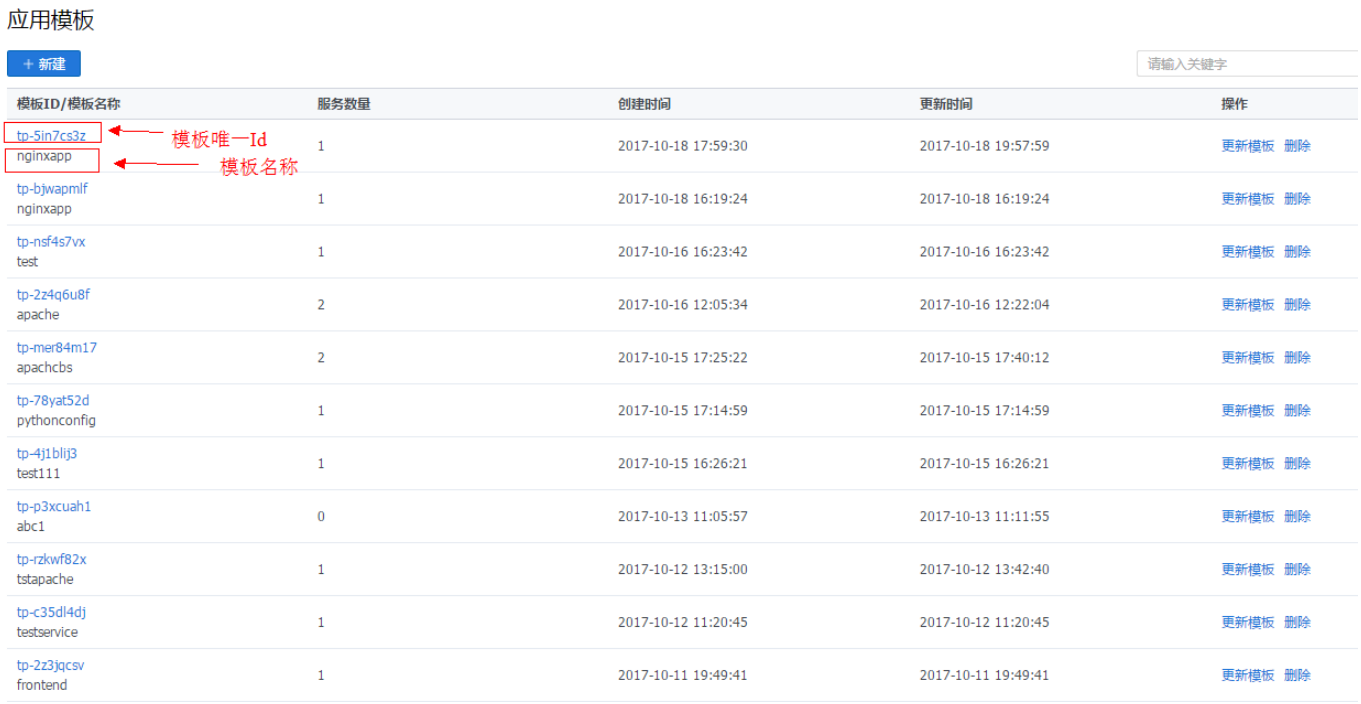
删除应用模板并不会影响已经使用该应用模板部署的应用。



更新应用模板

应用模板列表

登录 腾讯云容器服务控制台 。在应用管理->应用模板页面可以查看该用户下所有的应用模板。



应用模板过滤搜索

在应用模板列表搜索框中，输入模板名称可以过滤搜索出包含特定名称的模板。例如： 在搜索框中输入

nginx

，过滤出名称中包含

nginx

的模板。

应用模板详情

在应用模板页面点击应用模板ID，可以查看应用模板详情。

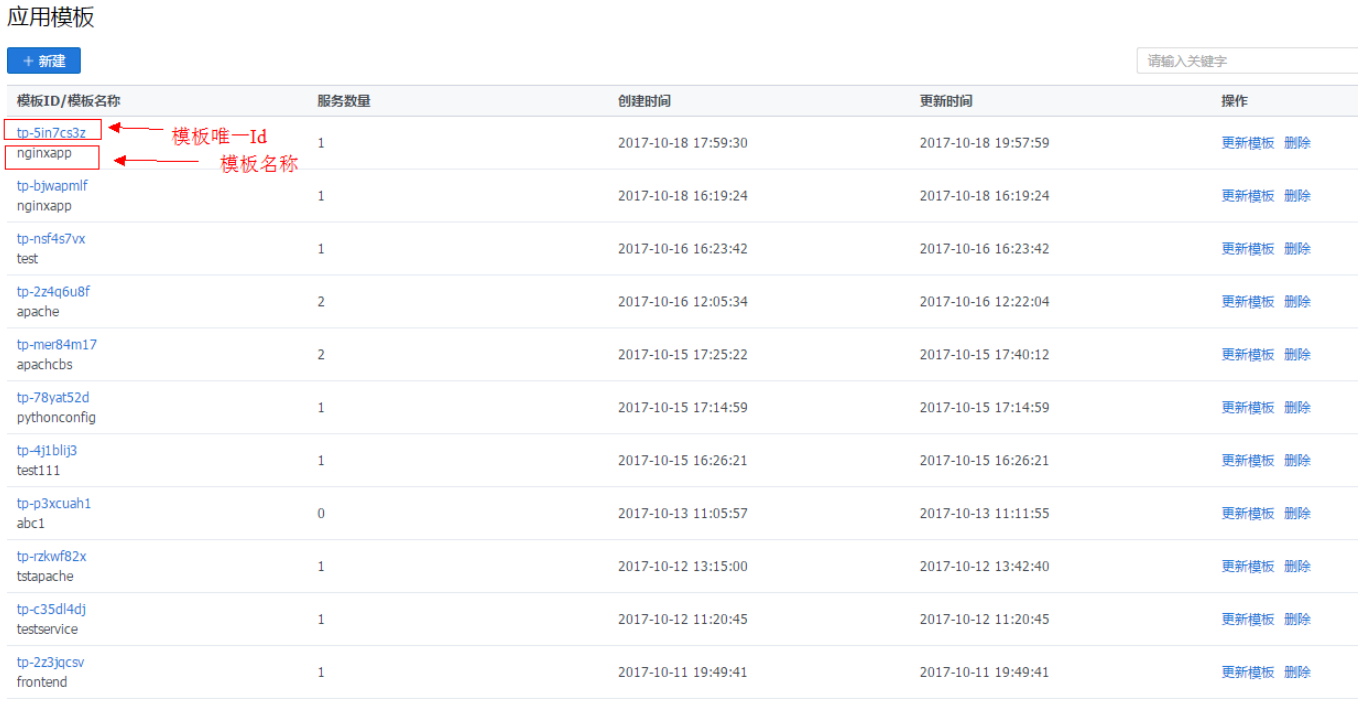
应用模板详情中分为模板内容部分和配置项部分。其中模板内容包括每个服务的YAML格式的描述文本。配置项部分展示模板默认的配置项内容，也为YAML格式。


返回 | tp-5in7cs3z 详情
模板名称 nginxapp
模板内容
nginx
1 apiVersion: extensions/v1beta1
2 kind: Deployment
3 metadata:
4 annotations:
5 description: 更新模板示例
6 creationTimestamp: null
7 name: nginx
8 namespace: '{{.NAMESPACE}}'
9 spec:
10 replicas: 2
11 revisionHistoryLimit: 5
12 selector: {}
13 strategy: {}
14 template:
15 metadata:
16 creationTimestamp: null
17 spec:
18 containers:
19 - image: nginx:latest
20 imagePullPolicy: Always
21 name: nginx
配置项
1 NAMESPACE: default
2

查看应用模板

应用模板列表

登录 腾讯云容器服务控制台 。在应用管理->应用模板页面可以该用户下所有的应用模板。



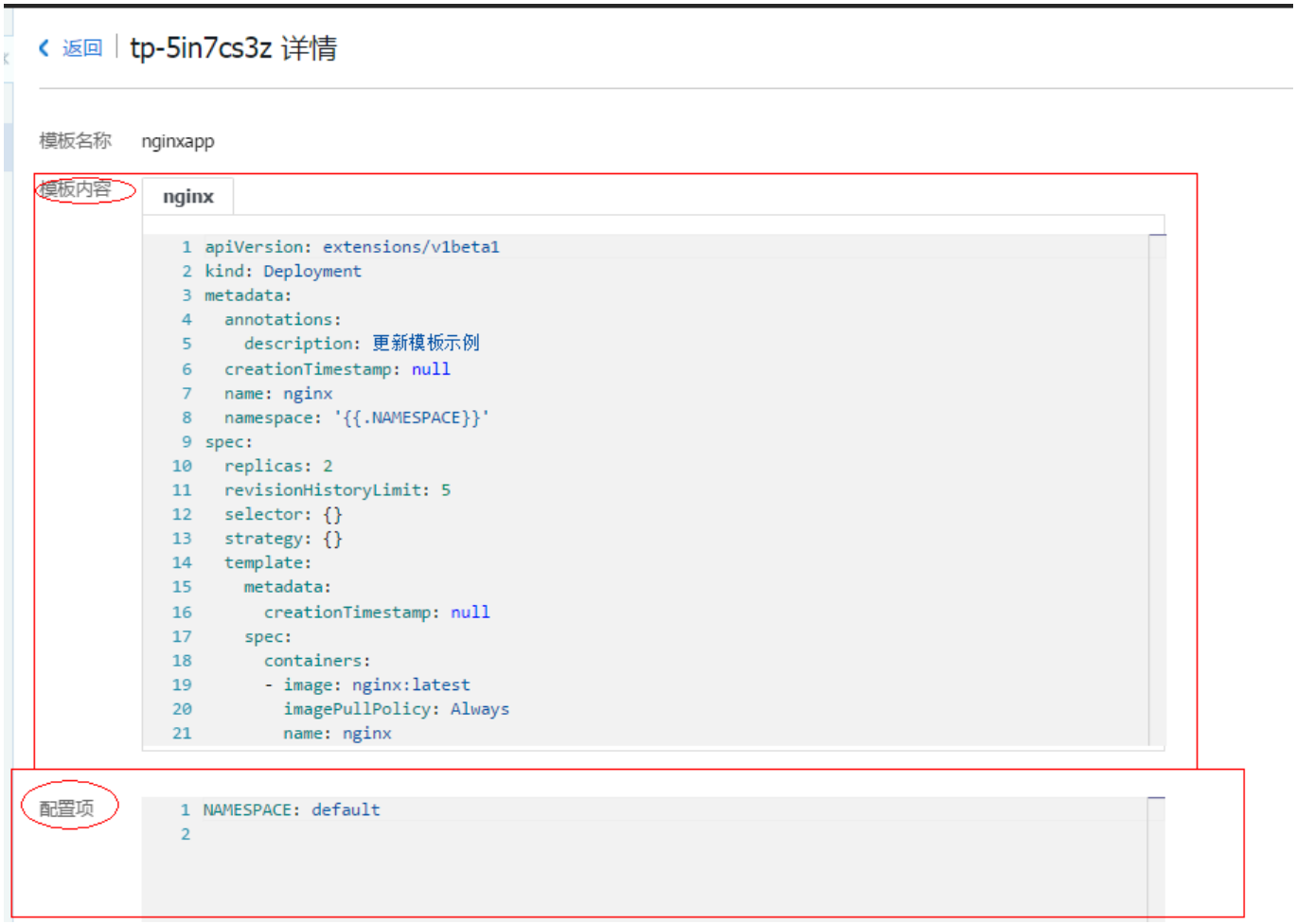
应用模板过滤搜索

在应用模板列表搜索框中，输入模板名称可以过滤搜索出包含特定名称的模板。

应用模板详情

在应用模板页面点击应用模板ID，可以查看应用模板详情。

应用模板详情展示如下，分为模板内容部分和配置项部分。



语法说明 - 基础语法

腾讯云容器服务底层基于Kubernetes编排引擎。在容器服务的应用模板中原生的支持Kubernetes的语法。本文将对Kubernetes的常用的语法做介绍。

CPU/Memory限制设置

Kubernetes使用Limit和Request对容器使用的资源进行限制。更多详细内容可以参数官方文档。

对于CPU和内存的设置，示例如下：

```
resources:
limits:
memory: 128Mi
cpu: 0
requests:
cpu: 200m
memory: 128Mi
```

在上面的示例中，通过设置resources中limits和requests参数中CPU和Memory的值，来限制容器CPU和memory的使用。上面的示例中，CPU在调度时被分配的CPU为0.2核，CPU限制使用的最大量无上限。内存在调度时被分配的128M，内存最大使用量被限制为128M。
在CPU资源的限制中，可以使用单位

m

,1m=0.001核。在内存的限制中，可
以使用M，Mi，G，Gi。1M=10001000Byte,1Mi=10241024Byte,1G=100010001000
Byte,1Gi=102410241024Byte。

命名空间设置

Kubernetes使用namespace参数来设置服务所在的命名空间。例如设置命名空间为wordpress

```
  namespace: wordpress
```

备注：
1、在同一个服务不同资源下，命名空间必须设置成一样。同一个应用中不同服务，也必须设置成一样。
2、如果namespace参数不填写，会默认使用default命名空间。

关于更多关于namespace的作用可以参考Namespace使用指引

实例数设置

Kubernetes使用replicas参数来设置实例的数量，如下所示:

```
spec:
  replicas: 2
```

通过设置replicas参数，把容器实例数设置为2。

镜像参数设置

Kubernetes使用image参数来设置容器使用的镜像。指定使用容器镜像busybox，版本号为latest示例如下：

```
image: busybox:latest
```

备注：如果不指定镜像的版本，会默认使用latest作为镜像的版本

服务类型设置

Kubernetes支持多种不同的服务访问类型，常用的服务访问类型包括: 负载均衡访问，集群内访问，节点端口访问，不设置访问方式。对于负载均衡访问，在容器服务中又分为外网负载均衡访问和VPC内负载均衡访问。Kubernetes通过Service资源中的

type

参数来指定服务的访问类型。

外网负载均衡访问

```
apiVersion: v1
kind: Service
metadata:
  labels:
  qcloud-app: nginx
  name: nginx
spec:
  ports:
  - name: tcp-80-80-pfbp1
  port: 80
  protocol: TCP
  targetPort: 80
  selector:
  qcloud-app: nginx
  type: LoadBalancer
```

如上面的例子所示，通过设置

type

为LoadBalancer将服务的访问类型指定为外网负载均衡访问。

VPC内负载均衡访问

```
apiVersion: v1
kind: Service
metadata:
```

```
  annotations:
  service.kubernetes.io/qcloud-loadbalancer-internal-subnetid: 'subnet-xxxxxx'
  labels:
  qcloud-app: nginx
  name: nginx
spec:
  ports:
  - name: tcp-80-80-pfbp1
  port: 80
  protocol: TCP
  targetPort: 80
  selector:
  qcloud-app: nginx
  type: LoadBalancer
```

VPC内访问的负载均衡，除了需要将通过设置

type

为LoadBalancer，还需要设置一个

service.kubernetes.io/qcloud-loadbalancer-internal-subnetid

的

annotations

，其值设置成负载均衡IP所在的子网(该子网必须在容器集群节点所在vpc内)。

集群内访问

```
apiVersion: v1
kind: Service
```

```
metadata:
  labels:
  qcloud-app: nginx
  name: nginx
spec:
  ports:
  - name: tcp-80-80-tlap9
  nodePort: 0
  port: 80
  protocol: TCP
  targetPort: 80
  selector:
  qcloud-app: nginx
  type: ClusterIP
```

如上面的例子所示，通过设置

type

为ClusterIP将服务的访问类型指定为集群内通过ClusterIP访问。对于外网负载均衡和内网负载均衡也会默认分配ClusterIP，也可以通过ClusterIP在集群内访问。

不设定访问方式

如果服务中没有

Service

资源，则不设定服务的访问方式。

备注： 容器服务中暂时未开放节点端口的这种访问方式，在外网负载均衡访问，VPC内负载均衡访问和集群内访问三种访问方式中都会默认为服务分配节点端口。在Ingress中也会通过注册配节点端口，来注册服务访问，如果Ingress中配置了对该服务的访问，将服务访问方式设定为"不设定访问方式"会到时候服务不可用。

更多关于服务访问方式的说明可以参考kubernetes 官方文档，也可以参考服务访问方式设置

磁盘挂载设置

Kubernetes支持挂载不类型的目录或者文件到容器指定的目录。其中涉及到两个步骤：挂载磁盘设置和挂载点设置。

例如挂载一个主机上的/home目录到容器中：

(1) 设置挂载的磁盘/home目录

```
volumes:
- hostPath:
path: /home
name: vol
```

(2) 将磁盘挂载到mnt目录

```
volumeMounts:
- mountPath: /mnt
name: vol
```

详细的关于磁盘挂载的说明，可以参考Kubernetes官方文档-volumes

容器平台现还支持挂载CBS盘，NFS，和配置文件。对于不同类型磁盘的挂载，可以参考。

扩展语法--CBS盘的使用

自定义变量--ReleaseConfig

多容器服务

Kubernetes中支持在一个实例中包含多个容器，例如下面服务中一个实例包含一个nginx容器和一个busybox容器。

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
  qcloud-app: nn
  name: nn
  namespace: default
spec:
  replicas: 1
  revisionHistoryLimit: 5
  selector:
  matchLabels:
  qcloud-app: nn
  strategy: {}
  template:
  metadata:
  creationTimestamp: null
  labels:
  qcloud-app: nn
  spec:
  containers:
  - image: nginx:latest
  imagePullPolicy: Always
  name: nginx
  resources:
  requests:
  cpu: 200m
  securityContext:
  privileged: false
  - args:
```

```
  - "36000"
  command:
  - sleep
  image: busybox:latest
  imagePullPolicy: Always
  name: busybox
  resources:
  requests:
  cpu: 200m
  securityContext:
  privileged: false
  serviceAccountName: ""
  volumes: null
status: {}
```

如上面的例子所示，在

containers

参数中设置不同的容器信息，可以使服务实例包含多个容器。

Kubernetes基础编排语法

Kubernetes通过定义API结构体中的每一个参数，来定义编排的描述语言。关于Kubernetes API参数的详细说明可以参考：

v1.7

v1.6

v1.5

更多关于Kubernetes编排描述语言的说明，可以参考API Overview

语法说明 - 扩展语法

为了更好的使用Kubernetes编排引擎，并使得Kubernetes编排引擎和腾讯云Iaas层的基础服务更好的结合。腾讯云容器服务在Kubernetes原生的编排语法的基础上，对编排的语法进行了扩展。下面将对扩展的语法进行介绍。

扩展语法---cbs盘使用

容器服务支持直接在服务实例上挂载CBS盘。具体的做法如下面的示例所示：

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
  qcloud-app: cbs
  name: cbs
  namespace: default
spec:
  replicas: 1
  revisionHistoryLimit: 5
  selector:
  matchLabels:
  qcloud-app: cbs
  strategy: {}
  template:
  metadata:
  creationTimestamp: null
  labels:
  qcloud-app: cbs
  spec:
  containers:
  - args:
  - "360000"
```

```
command:
- sleep
image: busybox:latest
imagePullPolicy: Always
name: busybox
resources:
requests:
cpu: 200m
securityContext:
privileged: true
volumeMounts:
- mountPath: /mnt
name: vol
serviceAccountName: ""
volumes:
- name: vol
qcloudCbs:
cbsDiskId: 'disk_xxxxxxxx'
fsType: ext4
```

在volume盘指定时，指定类型为qcloudCbs，并设置对应的cbsDiskId为需要挂载盘的instanceId。(Cbs盘的instanceId可以在云硬盘页面
查看)。挂载点位置设定与其他类型的磁盘设置一样，如上例所示，将cbs盘挂载到容器的/mnt目录。

备注：由于一块cbs盘只能同时被挂载一次，所以如果在服务中使用了cbs盘则容器实例不能大于1，且不支持滚动更新。

扩展语法---VPC内负载均衡访问

在基础语法中，通过设置

Service

中的

type

字段，能够设置服务的访问方式。在容器服务中定义

annotations

中

service.kubernetes.io/qcloud-loadbalancer-internal-subnetid

来区分是VPC内网访问负载均衡还是外网访问服务均衡。

如果设置了

service.kubernetes.io/qcloud-loadbalancer-internal-subnetid

这个

annotations

则认为是创建VPC内访问的负载均衡器，否则则认为是创建外网的负载均衡器。

具体示例如下所示：

```
apiVersion: v1
kind: Service
metadata:
  annotations:
  service.kubernetes.io/qcloud-loadbalancer-internal-subnetid: 'subnet-xxxxxx'
  labels:
  qcloud-app: nginx
  name: nginx
```

```
spec:
  ports:
  - name: tcp-80-80-pfbp1
  port: 80
  protocol: TCP
  targetPort: 80
  selector:
  qcloud-app: nginx
  type: LoadBalancer
```

其中

subnet-xxxxxx

为集群节点所在VPC内的一个子网。

语法说明 - 变量设置

□## 变量替换说明

应用模板中支持变量替换，变量的结构满足

{{.}}

，在

.

后面加上对应的变量名。需要被替换的变量在模板解析时会被配置文件中配置项的值替换。例如在模板中定义FRONTEND_REPLICAS：

```
spec:
 replicas: {{.FRONTEND_REPLICAS}}
```

在配置项中对变量FRONTEND_REPLICAS设定值：

```
FRONTEND_REPLICAS: 2
```

这样在模板解析时，模板文件中的

{{.FRONTEND_REPLICAS}}

会被替换成配置文件中的值"2"

备注： 变量名称满足正则表达式"[A-Za-z][A-Za-z0-9]*",最大长度为64个字符

自定义变量--ReleaseCBS

在容器服务中，如果需要挂载cbs盘，指定的描述语言如下所示：

```
volumes:
- name: vol
qcloudCbs:
cbsDiskId: 'disk-pr47vtvt'
fsType: ext4


volumeMounts:
- mountPath: /mnt
name: vol
```

(上面这段描述语言表示：将CBS盘disk-pr47vtvt作为vol盘挂载，具体的挂载到容器的/mnt目录。)

由于cbs盘是不能重复挂载的，通过应用模板在不同环境部署应用时，需要指定不同的cbs。所以容器服务提供ReleaseCBS变量来表示CBS盘。在应用部署时，会选择应用可使用的特定的CBS盘进行部署。具体示例如下

```
volumes:
- name: vol
qcloudCbs:
cbsDiskId: '{{.ReleaseCBS_pr47vtvt}}'
fsType: ext4
```

[补充cbs应用部署时cbs盘选择截图]

自定义变量--ReleaseSubnetId

在容器服务中，如果服务的访问方式选择为VPC内访问，则需要在描述文件中指定Lb对应的SubnetId，指定方式为，在service的描述文件中定义如下结果，其中subnet-s1jz1ycx为设定的SubnetId.

```
apiVersion: v1
```

```
kind: Service
metadata:
  annotations:
  service.kubernetes.io/qcloud-loadbalancer-internal-subnetid: 'subnet-s1jz1ycx'
```

由于SubnetId需要在集群所在的VPC(私有网络)下，所以SubnetId在不同集群中很有可能是不相同的。为了方便在应用部署时设置SubnetId，我们在模板中通过ReleaseSubNetId变量对需要设置的subnetId进行标识。下面是一个具体的示例：

```
apiVersion: v1
kind: Service
metadata:
  annotations:
  service.kubernetes.io/qcloud-loadbalancer-internal-subnetid: '{{.ReleaseSubnetId_XXXX}}'
```

在应用部署时，将在配置文件中为ReleaseSubnetId_XXXX变量指定对应的subnetId。

备注：由于应用中的多个服务可能会设置不同的subnetId，所有在ReleaseSubnetId名称后通过"_"(下划线)和服务名称来区分不同服务的subnetId。

自定义变量--ReleaseConfig

配置文件是程序运行很重要的一个部分，很多程序往往需要从磁盘的某个位置读取对应的配置文件，于是Kubernetes支持将通过confimap中的某个key挂载到容器指定的目录。更多关于Kubernetes configmap的说明可以参考Kubernetes官方文档。

应用中包含一个属于应用的配置文件，通过自定义变量ReleaseConfig可以将应用的配置文件中key挂载到容器指定目录。

具体的使用示例如下：

```
  volumes:
```

```
- configMap:
Name: '{{.ReleaseConfig}}'
items:
- key: NAMESPCE
mode: 511
path: NAMESPCE
- key: FRONTEND_REPLICAS
mode: 511
path: FRONTEND_REPLICAS
name: data
```

挂载点的设置为：

```
volumeMounts:
- mountPath: /mnt
name: data
```

应用的配置信息为：

NAMESPACE: default
FRONTEND_REPLICAS: 2

这样，配置文件中的key：NAMESPACE和FRONTEND_REPLICAS会被挂载到容器的/mnt目录，挂载的文件名称分别为NAMESPACE和FRONTEND_REPLICAS。文件的内容为配置文件中相应的key所对应的内容

版权所有：腾讯云计算（北京）有限责任公司

语法说明 - 限制说明

出于功能稳定闭环的考虑，容器平台对kubernetes的编排语法做了一定的限制。本文档将介绍这些限制的内容和原因。

支持的资源类型限制

Kubernetes包含多种不同资源类型。容器服务根据用户的需求，在UI界面上暂时只开放了

Deployment

和

Service

这两种最常用资源。(在创建

Deployment

会自动创建

Pod

资源和

replicaset

).

所有在容器服务的应用编排中，暂时只支持

Deployment

和

Service

这两种资源。

资源名称限制

应用模板的每个服务内，

Deployment

和

Service

资源名称必须和服务名称保持一致。

命名空间限制

(1) 应用模板中所有资源必须在一个命名空间内。

(2)

kube-system

命名空间暂时不支持创建服务，所以在应用模板中暂时不支持命名空间为

kube-system

。

Label标签限制

(1) 在应用模板中，可以使用Label标签对服务中的资源进行标记。在容器服务中，

Service

通过

Select Label

标签去寻找对应的

Pod

,从而实现与

Deployment

的关联。

Deployment

关联

Pod

使用的是

Deployment

自身的

Select Label

。所以为了实现

Deployment

和

Service

的管理，增加了

Deployment

和

Service

的

Select Label

必须一致的限制。

(2) 在应用模板中默认为每一个服务提供

qcloud-app

进行标识，提供

qcloud-application-label

标签标记属于哪个应用。这两个标签暂时不支持修改。

Cbs盘使用限制

(1) 由于cbs只能同时被一个容器实例挂载，所有在使用了cbs盘的服务，实例数最大为1

(2) 使用了cbs盘的服务，更新操作只能使用重新创建更新，暂时不支持滚动更新。

应用模板内容操作指引

在应用模板(或者应用)中，通过YAML格式的描述性语言将服务的部署信息保存成模板内容，从而实现多次复用和多环境下部署不同的应用。可以在模板内容区域导入和编辑服务的模板内容。

模板内容支持两种方式导入：1. UI导入服务 2. 新增空服务

> 注意：
> 可以根据场景选择使用其中的任意一种方式

UI导入服务
可以通过在控制台填写服务的相关参数，自动转换成模板内容的YAML描述文件，适用于YAML描述文件不存在的情况。新增空服务
可以直接拷贝已经存在的YAML描述文件内容到模板内容区域，适用于YAML描述文件已经存在的情况。

UI导入服务的模板内容

在新建模板(或者新建应用)页面，点击

UI导入服务

按钮，可以在控制台填写对应的参数，自动转换成服务的模板内容。

服务导入页面参数设置，与服务创建页面参数设置保持一致。
主要参数包括： 服务的基本信息，数据卷设置，运行容器参数设置，实例数量设置，服务访问类型和端口等。

例如在示例中部署一个nginx服务，填写的参数如下：

设置服务的基本信息：

1. 填写服务名称

nginx

2. 填写服务描述

nginx服务

设置服务的数据卷信息：
未使用数据盘，无。

设置镜像参数：

1. 在设置容器运行参数中的镜像参数：
   容器名称设置为

   nginx

   镜像名称设置为

   nginx

   版本号选择为

   latest

设置服务的实例数：

1. 服务的实例数设置为1

设置服务的访问方式：

1. 服务的访问方式设置为集群内访问
2. 服务的访问端口：容器端口和服务端口都设置成80

更多关于参数设置内容可以参考 服务创建操作的相关文档。

填写参数后，点击导入服务的

完成

按钮，控制台自动导入服务的模板内容。

新增空服务导入模板内容

如果已经存在服务对应的模板内容的YAML文件，可以直接将模板内容导入到编辑框中。具体的步骤如下：

步骤一 创建对应的服务：
点击"新增空服务"按钮，新增一个服务。

步骤二 导入模板内容：

可以将下面YAML文件中的内容，直接拷贝到编辑框中，导入服务的模板内容。

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  annotations:
  description: nginx服务
  creationTimestamp: null
  name: nginx
  namespace: default
spec:
  replicas: 1
  revisionHistoryLimit: 5
  selector: {}
  strategy: {}
  template:
  metadata:
```

```
  creationTimestamp: null
  spec:
  containers:
  - image: nginx:latest
  imagePullPolicy: Always
  name: nginx
  resources:
  requests:
  cpu: 200m
  securityContext:
  privileged: false
  serviceAccountName: ""
  volumes: null
status: {}

---
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  name: nginx
  namespace: default
spec:
  ports:
  - name: tcp-80-80-xoq5o
  nodePort: 0
  port: 80
  protocol: TCP
  targetPort: 80
  selector: {}
  type: LoadBalancer
status:
  loadBalancer: {}
```

步骤三 提取模板内容中的变量：

在示例中，使用了

NAMESPACE

将

namespace

参数作为变量进行替换，更多关于变量替换的说明可以参考变量替换。

系统自动提取模板内容中的变量作为配置项。然后在配置项中填写变量的默认值，这里设置成

default

。

编辑模板内容

在模板内容区域，可以直接对模板内容进行编辑。例如在模板内容中将服务实例的副本数修改为

2

。

模板内容的语法规

则，满足Kubernetes定义的语法，更多关于模板内容规则的说明可以参考编排语法能力说明。

模板内容中参数配置化

在模板内容中，如果有些参数在不同环境下是不相同的，或者需要进行频繁的修改。可以将这部分参数转换成一个可配置的变量。例如，上面的服务实例副本数，可能在不同环境下是不相同的，我们可以将它转换成变量

NGINX_REPLICAS

。关于模板中变量的使用说明，可以参考变量设置。

系统提取模板内容中的变量作为配置项。然后在配置项中填写变量的默认值，这里设置成

2

。

在不同的环境部署时，可以根据需要，选择不同的配置项，从而调整服务的实例数量。