

KubeSphere 文档 Express v1.0

KubeSphere 简介

产品介绍

[KubeSphere](#) 是在目前主流容器调度平台 [Kubernetes](#) 之上构建的企业级分布式多租户容器管理平台，提供简单易用的操作界面以及向导式操作方式，在降低用户使用容器调度平台学习成本的同时，极大减轻开发、测试、运维的日常工作的复杂度。除此之外，平台已经整合并优化了多个适用于容器场景的功能模块，以帮助企业轻松应对多租户、工作负载和集群管理、服务与网络管理、应用管理、镜像仓库管理和存储管理等业务场景，下一个版本将支持服务治理、CI/CD、监控日志、大数据、人工智能以及 LDAP 集成等复杂业务场景，KubeSphere 提供了在生产环境集群部署的全栈化容器部署与管理平台。

产品优势

设计愿景

众所周知，开源项目 Kubernetes 已经成为事实上的编排平台的领导者，是下一代分布式架构的王者，其在自动化部署、扩展性、以及管理容器化的应用已经体现出独特的优势。然而，很多人学习 Kubernetes，就会发现有点不知所措，因为 Kubernetes 本身有许多组件并且还有一些组件需要自行安装和部署，比如存储和网络目前 Kubernetes 仅提供的是开源的解决方案或项目，可能在某种程度上难以安装，维护和操作，对于用户来说学习成本和门槛都不是一件易事。

如果无论如何都得将应用部署在云上运行，为什么不让 KubeSphere 为您运行 Kubernetes 且更好地管理运行的资源呢？这样您就可以继续运行应用程序和工作负载并专注于这些更重要的业务。因为通过 KubeSphere 来快速创建 Kubernetes 集群、部署应用、添加服务、CI/CD、集群扩容、微服务治理、日志记录和资源监控，以及利用 KubeSphere 的其他诸多强大功能是多么容易。换句话说，Kubernetes 是一个很棒的开源项目（或被认为是一个框架），但是 KubeSphere 是一款非常专业的企业级平台产品，专注于解决用户在复杂业务场景中的痛点，提供更好更专业的用户体验。

最重要的是，KubeSphere 在存储和网络方面提供了最优的解决方案，比如存储除了支持开源的 Ceph RBD 和 GlusterFS 之外，还提供 [青云云平台的块存储](#) 作为 Kubernetes 的持久化存储，通过集成的 QingCloud CSI 和 NeonSAN CSI 插件，即可使用青云提供的高性能块存储作为存储卷挂载至工作负载，为企业应用和数据提供更稳定安全的存储。

为什么选择 KubeSphere？

KubeSphere 为企业用户提供高性能可伸缩的容器应用管理服务，旨在帮助企业完成新一代互联网技术驱动下的数字化转型，加速业务的快速迭代与交付，以满足企业日新月异的业务需求。

优势	说明
灵活便捷的存储配置方案	针对 Kubernetes 在存储配置上的复杂性，KubeSphere 支持动态分配存储卷，屏蔽底层存储差异性，极大地简化分配和回收存储卷流程。并且提供青云块存储与 Kubernetes 存储对接的解决方案，如目前的 Flex Volume, QingCloud CSI 插件和即将支持的 NeonSAN CSI 插件。
弹性伸缩	支持部署后集群节点扩容以及 Pod 动态的横向伸缩，保证集群和资源的高可用和可靠性。
统一门户	无基础设施依赖，无 Kubernetes 依赖，整合多种云平台，纳管多源 Kubernetes 集群。
安全第一位	多租户、细粒度安全架构设计，并可集成企业中心化用户中心系统。
极简体验，向导UI	面向开发、测试、运维友好的 UI，向导式用户体验，降低 Kubernetes 学习成本的设计理念。
松耦合功能模块设计	除提供基于原生 k8s 的管理功能之外，用户可以使用 KubeSphere 集成的诸如镜像仓库、应用仓库、监控、日志模块，也可通过配置的方式集成自建的相关服务。
整体化容器平台解决方案	高级版将贯通应用开发、管理、CI/CD、服务治理、发布上线和流量管控等一系列流程。
可选的商业网络和存储解决方案	除开源解决方案外，如用户对网络和存储有更高要求，可选用青云作为底层平台，可以使用性价比更高的网络和存储解决方案。

产品规划

Community Edition (社区版) => Express Edition (易捷版) => Advanced Edition (高级版)

产品功能

KubeSphere 为用户提供了一个具备极致体验的 Web 控制台，让您能够像使用任何其他互联网产品一样，快速上手各项功能与服务。KubeSphere 目前集成了应用负载、服务与网络、应用管理、资源管理和平台管理共五大模块，以下从专业的角度为您详解各个模块的功能服务：

功能	说明
应用负载管理	对 kubernetes 中的多种 workload 提供向导式管理界面，包括 Deployments, Daemon Sets, Stateful Sets，并提供 HPA 支持。
服务与网络管理	基于原生 API，对 k8s 中的服务 (Service)、应用路由 (ingress) 等功能提供向导式管理界面，快速将应用暴露以供用户访问。高级版将集成 istio 中的 微服务治理、熔断、灰度发布、限流、智能路由等功能提供向导式管理界面。 如果部署在青云平台之上，可以使用插件对接青云的负载均衡器。
应用管理	后端使用开源的 OpenPitrix 服务，为用户提供应用全生命周期管理功能，包括：应用仓库管理、应用拓扑图、APM、应用变更和发布、应用上下线审批、版本控制、鲁棒性测试等。
资源管理	提供存储、主机、集群以及配额管理。存储既支持主流开源存储解决方案，也可对接青云的块存储和 NeonSAN。可批量添加主机，且对主机平台及系统弱依赖。并且支持镜像仓库管理、镜像复制、权限管理、镜像安全扫描。

功能	说明
平台管理	提供基于角色的细粒度权限多租户管理，平台服务间进行加密通信；提供操作审计日志；可对宿主机以及容器镜像进行安全扫描并发现漏洞。

名词解释

了解和使用 KubeSphere 管理平台，会涉及到以下的基本概念：

KubeSphere	Kubernetes
项目	Namespace，为 Kubernetes 集群提供虚拟的隔离作用，详细参考 Namespace 。
部署	Deployment，表示用户对 Kubernetes 集群的一次更新操作，详细参考 Deployment 。
有状态副本集	StatefulSet，用来管理有状态应用，可以保证部署和 scale 的顺序，详细参考 StatefulSet 。
守护进程集	DaemonSet，保证在每个 Node 上都运行一个容器副本，常用来部署一些集群的日志、监控或者其他系统管理应用，详细参考 Daemonset 。
服务	Service，一个 Kurnetee 服务是一个最小的对象，类似 Pod，和其它的终端对象一样，详细参考 Service 。
应用路由	Ingress，是授权入站连接到达集群服务的规则集合。可通过 Ingress 配置提供外部可访问的 URL、负载均衡、SSL、基于名称的虚拟主机等，详细参考 Ingress 。
镜像仓库	Image Registries，镜像仓库用于存放 Docker 镜像，Docker 镜像用于部署容器服务，详细参考 Images 。
存储卷	PersistentVolumeClaim (PVC)，满足用户对于持久化存储的需求，用户将 Pod 内需要持久化的数据挂载至存储卷，实现删除 Pod 后，数据仍保留在存储卷内。Kubesphere 推荐使用动态分配存储，当集群管理员配置存储类型后，集群用户可一键式分配和回收存储卷，无需关心存储底层细节。详细参考 Volume 。
存储类型	StorageClass，为管理员提供了描述存储“Class (类) ”的方法，包含 Provisioner、ReclaimPolicy 和 Parameters 。详细参考 StorageClass 。

KubeSphere	Kubernetes
主机	Node, Kubernetes 集群中的计算能力由 Node 提供, Kubernetes 集群中的 Node 是所有 Pod 运行所在的工作主机, 可以是物理机也可以是虚拟机。详细参考 Nodes 。

KubeSphere 安装指南

简介

[KubeSphere](#) 是在目前主流容器调度平台 [Kubernetes](#) 之上构建的 **企业级分布式多租户容器管理平台**，为用户提供简单易用的操作界面以及向导式操作方式，KubeSphere 提供了在生产环境集群部署的全栈化容器部署与管理平台。

部署 KubeSphere

KubeSphere 部署支持 `all-in-one` 和 `multi-node` 两种部署模式，KubeSphere Installer 采用 [Ansible](#) 对部署目标机器及部署流程进行集中化管理配置。采用预配置模板，可以在部署前通过对相关配置文件进行自定义实现对部署过程的预配置，以适应不同的 IT 环境，帮助您快速部署 KubeSphere。

- KubeSphere 集群中由于部署服务的不同，分为管理节点和计算节点两个角色。
- 当进行 `all-in-one` 模式进行单节点部署时，这个节点既是管理节点，也是计算节点。
- 当进行 `multi-node` 模式部署多节点集群时，可在配置文件中设置集群角色。
- 由于部署过程中需要更新操作系统和从镜像仓库拉取镜像，因此必须能够访问外网。无外网环境需先下载离线安装包安装。
- 如果是新装系统，在 Software Selection 界面需要把 OpenSSH Server 选上。

All-in-One 模式

`All-in-One` 模式即单节点部署，仅建议您用来测试或熟悉部署流程和了解 KubeSphere 功能特性，在正式使用环境建议使用 `multi-node` 模式，请参考下文的 `multi-node` 模式。

第一步：准备主机（单节点）

您可以参考以下节点规格准备一台符合要求的主机节点开始 `all-in-one` 模式的部署。

操作系统	最小配置	推荐配置
Ubuntu 16.04 LTS 64bit	CPU: 8 核 内存: 12 G 磁盘: 40 G	CPU: 16 核 内存: 32 G 磁盘: 100 G
CentOS 7.4 64bit	CPU: 8 核 内存: 12 G 磁盘: 40 G	CPU: 16 核 内存: 32 G 磁盘: 100 G

第二步: 准备安装包 (单节点)

1. 下载 [KubeSphere Installer](#), 跳转到下载页面后也可以通过 `curl -O url` 或 `wget url` 命令获取 Installer。

KubeSphere 版本	支持系统 (将支持更多系统)
Dev 版	Ubuntu 16.04 LTS 64bit, CentOS 7.4 64bit
Stable (Alpha 版)	Ubuntu 16.04 LTS 64bit
Offline 版	Ubuntu 16.04.4 LTS 64bit, Ubuntu 16.04.5 LTS 64bit

2. 获取 KubeSphere 安装包后, 以 Alpha 版本的安装包为例, 执行以下命令解压安装包。若下载的是 Dev 或 Offline 版本, 则替换为 Dev 或 Offline 对应的包名和目录名。

```
$ tar -zxvf kubesphere-all-express-1.0.0-alpha.tar.gz
```

3. 进入 “`kubesphere-all-express-1.0.0-alpha`” 文件夹

```
$ cd kubesphere-all-express-1.0.0-alpha
```

第三步: 安装 KubeSphere (单节点)

说明:

- 通常情况您不需要修改任何配置, 直接安装即可。
- 若您需要自定义配置文件的安装参数, 如网络、存储等相关内容需在 `conf/vars.yaml` 配置文件中指定或修改。
- 由于 Kubernetes 集群的 Cluster IP 子网网段默认是 10.233.0.0/18, Pod 的子网网段默认是 10.233.64.0/18, 因此部署 KubeSphere 的节点 IP 地址范围不应与以上两个网段有重複, 若遇到地址范围冲突可在配置文件 `conf/vars.yaml` 修改 `kube_service_addresses` 或 `kube_pods_subnet` 的参数。
- 网络: 默认插件 `calico`。
- 支持存储类型: QingCloud-CSI (Dev 版支持)、GlusterFS、CephRBD、local-storage, 存储配置相关的详细信息请参考 [存储配置说明](#)。
- All-in-One 默认会用 local storage 作为存储类型, 由于 local storage 不支持动态分配, installer 会预先创建 8 个可用的 10G PV 供使用, 若存储空间不足则需要手动创建 Persistent Volume (PV), 参见 [Local Volume 使用方法](#)。

KubeSphere 部署过程中将会自动化地进行环境和文件监测、平台依赖软件的安装、Kubernetes 和 etcd 的自动化部署, 以及存储的自动化配置。Installer 默认安装的 Kubernetes 版本是 v1.10.5, 目前已支持 v1.11.2, 如需安装 v1.11.2 可在配置文件 `conf/vars.yaml` 中修改 `kube_version` 的参数为 v1.11.2, 再执行安装, 安装成功后可通过 KubeSphere 控制台右上角点击关于查看安装的版本。KubeSphere 安装包将会自动安装一些依赖软件, 如 Ansible (v2.4+), Python-netaddr (v0.7.18+), Jinja (v2.9+)。

注意, 若下载的是离线安装包, 安装仅支持默认的 Kubernetes v1.10.5, 不支持修改版本号。

参考以下步骤开始 all-in-one 部署:

1. 进入 `scripts` 目录

```
$ cd scripts
```

2. 执行 `install.sh` 脚本:

```
$ ./install.sh
```

3. 输入数字 1 选择第一种即 all-in-one 模式开始部署：

```
#####
# KubeSphere Installer Menu
#####
* 1) All-in-one
* 2) Multi-node
* 3) Cluster-scaling
* 4) Quit
#####
https://kubesphere.io/          2018-07-27
#####
Please input an option: 1
```

4. 测试 KubeSphere 单节点部署是否成功：

(1) 待 install.sh 执行完后，当看到如下 “Successful” 界面，则说明 KubeSphere 安装成功。

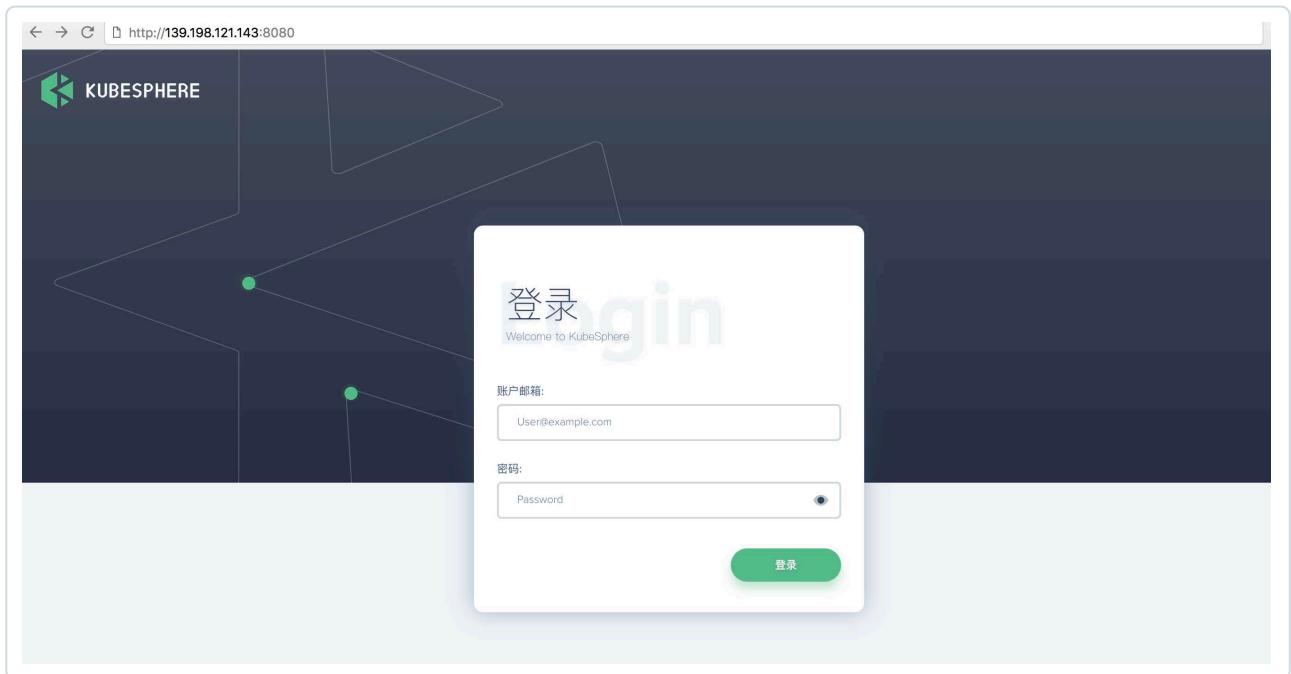
```
PLAY RECAP ****
KubeSphere : ok=69 changed=68 unreachable=0 failed=0
Successful!
#####
KubeSphere is running!
Master IP: 192.168.100.10
ks-console-nodeport: 32117
ks-apiserver-nodeport: 32002
#####
```

(2) 以上可以看到两个 nodeport，也可以通过命令 `kubectl get svc -n kubesphere-system` 查看端口号。在 Kubernetes 中 nodeport 是提供给集群外部客户访问该 Service 入口的一种方式，Kubernetes 中的 nodeport 一般是高位 30000 – 32767。您可以通过浏览器，使用集群中任一节点的 IP 地址和端口号即 `<NodeIP>:ks-console-nodeport` 访问 KubeSphere 控制台，如上图

`http://192.168.100.10:32117`。由于服务的常用访问端口通常是低位，因此也可以通过公网 IP 并将高

位端口转发后访问控制台，如：<http://139.198.121.143:8080>，即可进入 KubeSphere 登录界面，能看到如下用户界面说明 KubeSphere 能够正常访问和使用：

注：若公网 IP 有防火墙，请在防火墙添加规则放行对应的端口，外部才能够访问。



KubeSphere 部署成功后，可以使用以下的用户名和密码登录 KubeSphere 控制台体验：

Account: admin@kubesphere.io

Password: passw0rd

关于如何使用请参考 [《KubeSphere 用户指南》](#)。

Multi–Node 模式

Multi–Node 即多节点集群部署，部署前建议您选择集群中任意一个节点作为一台任务执行机 (taskbox)，为准备部署的集群中其他节点执行部署的任务，且 taskbox 应能够与待部署的其他节点进行 ssh 通信。

第一步: 准备主机 (多节点)

您可以参考以下节点规格 准备 至少 2 台 符合要求的主机节点开始 multi-node 模式的部署。

操作系统	最小配置	推荐配置
ubuntu 16.04 LTS 64bit	CPU: 8 核 内存: 12 G 磁盘: 40 G	CPU: 16 核 内存: 32 G 磁盘: 100 G
CentOS 7.4 64bit	CPU: 8 核 内存: 12 G 磁盘: 40 G	CPU: 16 核 内存: 32 G 磁盘: 100 G

以下用一个示例介绍 multi-node 模式部署多节点，此示例准备了 3 台主机，以主机名为 master 的节点作为任务执行机 taskbox，各节点主机名可由用户自定义。假设主机信息如下所示：

主机IP	主机名	集群角色
192.168.0.10	master	master, etcd, node
192.168.0.20	node1	node
192.168.0.30	node2	node

集群架构：单 master 单 etcd 多 node



etcd 作为一个高可用键值存储系统，etcd 节点个数至少需要 1 个，部署多个 etcd 能够使集群更可靠，etcd 节点个数建议设置为 **奇数个**，在当前 KubeSphere Express 版本暂支持单个 etcd 节点，将会在下一个 Advanced Edition 版本中支持 etcd 多节点部署。

第二步: 准备安装包 (多节点)

1. 下载 [KubeSphere Installer](#)，跳转到下载页面后也可以通过 `curl -O url` 或 `wget url` 命令获取 Installer。

KubeSphere 版本	支持系统 (将支持更多系统)
Dev 版	Ubuntu 16.04 LTS 64bit, CentOS 7.4 64bit
Stable (Alpha 版)	Ubuntu 16.04 LTS 64bit
Offline 版	Ubuntu 16.04.4 LTS 64bit, Ubuntu 16.04.5 LTS 64bit

2. 获取 KubeSphere 安装包后，以 Alpha 版本的安装包为例，执行以下命令解压安装包。若下载的是 Dev 或 Offline 版本，则替换为 Dev 或 Offline 对应的包名和目录名。

```
$ tar -zxvf kubesphere-all-express-1.0.0-alpha.tar.gz
```

3. 进入“kubesphere-all-express-1.0.0-alpha”文件夹

```
$ cd kubesphere-all-express-1.0.0-alpha
```

4. 编辑主机配置文件 `conf/hosts.ini`，为了对待部署目标机器及部署流程进行集中化管理配置，集群中各个节点在主机配置文件 `hosts.ini` 中应参考如下配置，以 Alpha 版本的主机配置文件为例。以下示例在 Ubuntu 16.04 上使用 `ubuntu` 用户安装，每台机器信息占一行，不能分行。

Alpha 版示例：

[all]

```
maser ansible_connection=local local_release_dir={{ansible_env.HOME}}/releases ansible_user=ubuntu ansible_become=yes  
node1 ansible_host=192.168.0.20 ip=192.168.0.20 ansible_user=ubuntu ansible_become=yes ansible_become_pass=  
node2 ansible_host=192.168.0.30 ip=192.168.0.30 ansible_user=ubuntu ansible_become=yes ansible_become_pass=
```

[kube-master]

```
master
```

[kube-node]

```
master  
node1  
node2
```

[etcd]

```
master
```

[k8s-cluster:children]

```
kube-node  
kube-master
```

说明：

- [all] 中需要修改集群中各个节点的内网 IP 和主机 ubuntu 用户密码。主机名为 “master”的节点作为 taskbox 仅需要将 `ansible_become_pass` 替换为当前任务执行机的 ubuntu 用户登录密码, [all] 中其它参数比如 node1 和 node2 需要分别替换 `ansible_host` 和 `ip` 为当前 node1 和 node2 的内网 IP, node1 和 node2 的 `ansible_become_pass` 即替换为各自主机的 ubuntu 用户登录密码。
- “master” 节点作为 taskbox, 用来执行整个集群的部署任务, 同时 “master” 节点在 kubernetes 集群中也作为 master 和 etcd, 应将主机名 “master” 填入 [kube-master] 和 [etcd] 部分。
- 主机名为 “master”, “node1”, “node2”的节点, 作为 kubernetes 集群的 node 节点, 应填入 [kube-node] 部分。

参数解释:

- `ansible_host`: 集群中将要连接的主机名
 - `ip`: 集群中将要连接的主机 IP
 - `ansible_user`: 默认使用的 SSH 登录用户名
 - `ansible_become`: 是否允许权限升级 (yes/no)
 - `ansible_become_user`: 权限升级用户 (root)
 - `ansible_become_pass`: 待连接主机的密码.
- 若下载的是 Dev 或 Offline 版本的安装包, 安装包中 `conf/hosts.ini` 的 [all] 部分参数如 `ansible_host`、`ip`、`ansible_become_pass` 和 `ansible_ssh_pass` 需替换为您实际部署环境中各节点对应的参数。注意 [all] 中参数的配置方式分为 root 和 非 root 用户, 非 root 用户的配置方式在安装包的 `conf/hosts.ini` 的注释部分已给出示例, 请根据实际的用户身份修改配置参数。

5. Multi–Node 模式进行多节点部署时, 您需要预先准备好对应的存储端, 再参考 [存储配置说明](#) 配置集群的存储类型。网络、存储等相关内容需在 `conf/vars.yaml` 配置文件中指定或修改。

说明:

- 根据配置文件按需修改相关配置项, 未做修改将以默认参数执行。
- 由于 Kubernetes 集群的 Cluster IP 子网网段默认是 10.233.0.0/18, Pod 的子网网段默认是 10.233.64.0/18, 因此部署 KubeSphere 的节点 IP 地址范围不应与以上两个网段有重复, 若遇到地址范围冲突可在配置文件 `conf/vars.yaml` 修改 `kube_service_addresses` 或 `kube_pods_subnet` 的参数。

- 网络：默认插件 `calico`。
- 支持存储类型：QingCloud-CSl（Dev 版支持）、GlusterFS、CephRBD 等，存储配置相关的详细信息请参考 [存储配置说明](#)
- 通常情况您需要配置持久化存储，multi-node 不支持 local storage，因此把 local storage 的配置修改为 false，然后配置持久化存储如 QingCloud-CSl、GlusterFS、CephRBD 等。如下所示为配置 QingCloud-CSl（`qy_access_key_id`、`qy_secret_access_key` 和 `qy_zone` 应替换为您实际环境的参数）。
- 安装 KubeSphere 后，如果需要对集群节点扩容，可参考 [集群节点扩容说明](#)。

示例：

```
# Local volume provisioner deployment(Only all-in-one)
local_volume_provisioner_enabled: false
local_volume_provisioner_storage_class: local
local_volume_is_default_class: false

# QingCloud-CSI
qy_csi_enabled: true
qy_csi_is_default_class: true
# Access key pair can be created in QingCloud console
qy_access_key_id: ACCESS_KEY_ID
qy_secret_access_key: ACCESS_KEY_SECRET
# Zone should be the same as Kubernetes cluster
qy_zone: ZONE
# QingCloud IaaS platform service url.
qy_host: api.qingcloud.com
qy_port: 443
qy_protocol: https
qy_uri: /iaas
qy_connection_retries: 3
qy_connection_timeout: 30
# The type of volume in QingCloud IaaS platform.
# 0 represents high performance volume.
# 3 represents super high performance volume.
# 1 or 2 represents high capacity volume depending on cluster's zone.
qy_type: 0
qy_maxSize: 500
qy_minSize: 10
qy_stepSize: 10
qy_fsType: ext4
```

第三步：安装 KubeSphere（多节点）

KubeSphere 多节点部署会自动化地进行环境和文件监测、平台依赖软件的安装、Kubernetes 和 etcd 集群的自动化部署，以及存储的自动化配置。Installer 默认安装的 Kubernetes 版本是 v1.10.5，目前已

支持 v1.11.2，如需安装 v1.11.2 可在配置文件 `conf/vars.yaml` 中修改 `kube_version` 的参数为 v1.11.2，再执行安装，安装成功后可通过 KubeSphere 控制台右上角点击关于查看安装的版本。KubeSphere 安装包将会自动安装一些依赖软件，如 Ansible (v2.4+)，Python–netaddr (v0.7.18+)，Jinja (v2.9+)。

参考以下步骤开始 multi-node 部署：

1. 进入 `scripts` 目录

```
$ cd scripts
```

2. 执行 `install.sh` 脚本：

```
$ ./install.sh
```

3. 输入数字 `2` 选择第二种 Multi-node 模式开始部署：

```
#####
# KubeSphere Installer Menu
#####
* 1) All-in-one
* 2) Multi-node
* 3) Cluster-scaling
* 4) Quit
#####
https://kubesphere.io/      2018-07-27
#####
Please input an option: 2
```

提示：

- 安装程序会提示您是否已经配置过存储，若未配置请输入 “no”，返回目录继续配置存储并参考 [存储配置说明](#)
- Dev 和 Offline 版本的安装包不再需要配置 ssh 免密登录，只提示用户是否配置过存储。
- 若下载的是 Alpha 版本的安装包，taskbox 需配置与待部署集群中所有节点的 [ssh 免密登录](#)

|，若还未配置 ssh 免密登录，在执行 `install.sh` 安装脚本时会提示用户是否已经配置免密登录，输入 “no” 安装程序将会帮您自动配置 ssh 免密登录，如下图所示：

```
#####
# KubeSphere Installer Menu
#####
* 1) All-in-one
* 2) Multi-node
* 3) Cluster-scaling
* 4) Quit
#####
https://kubesphere.io/          2018-07-27
#####
Please input an option: 2
2
Have you configured storage parameters in conf/vars.yml yet? (yes/no)
yes
Password-less SSH communication is necessary, have you configured yet?
If not, it will be created automatically. (yes/no)
no
Generating public/private rsa key pair.
Created directory '/home/ubuntu/.ssh'.
Your identification has been saved in /home/ubuntu/.ssh/id_rsa.
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub.
```

4. 测试 KubeSphere 集群部署是否成功：

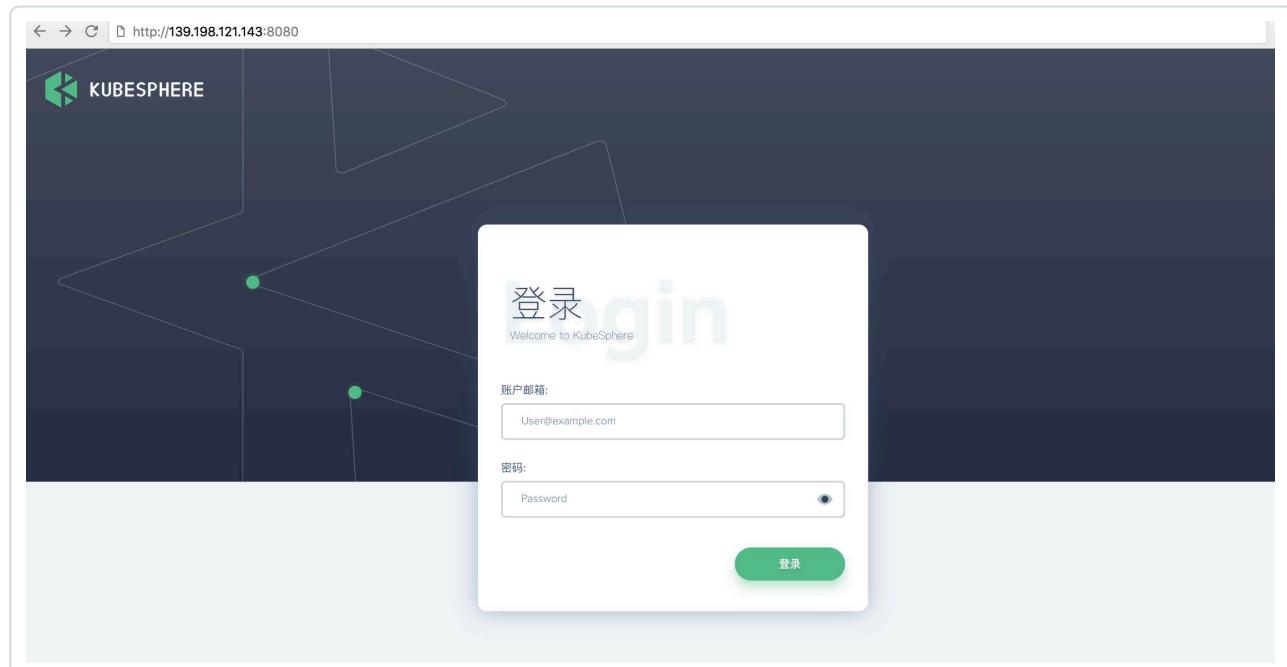
(1) 待 `install.sh` 执行完后，当看到如下 “Successful” 界面，则说明 KubeSphere 安装成功：

```
PLAY RECAP ****
KubeSphere : ok=69 changed=68 unreachable=0 failed=0
Successful!
#####
KubeSphere is running!
Master IP: 192.168.100.10
ks-console-nodeport: 32117
ks-apiserver-nodeport: 32002
#####
```

(2) 以上可以看到两个 nodeport，也可以通过命令 `kubectl get svc -n kubesphere-system` 查看端口号。在 Kubernetes 中 nodeport 是提供给集群外部客户访问该 Service 入口的一种方式，Kubernetes 中的 nodeport 一般是高位 30000 – 32767。您可以通过浏览器，使用集群中任一节点的 IP 地址和端口号即 `<NodeIP>:ks-console-nodeport` 访问 KubeSphere 控制台，如上图

`http://192.168.100.10:32117`。由于服务的常用访问端口通常是低位，因此也可以通过公网 IP 并将高位端口转发后访问控制台，如：`http://139.198.121.143:8080`，即可进入 KubeSphere 登录界面，能看到如下用户界面说明 KubeSphere 能够正常访问和使用：

注：若公网 IP 有防火墙，请在防火墙添加规则放行对应的端口，外部才能够访问。



KubeSphere 部署成功后，可以使用以下的用户名和密码登录 KubeSphere 控制台体验：

Account: admin@kubesphere.io

Password: passw0rd

关于如何使用请参考 [《KubeSphere 用户指南》](#)。

存储配置说明

可使用 [青云云平台块存储](#)、[GlusterFS](#) 或 [CephRBD](#) 作为持久化存储（更多的存储类型持续更新中），使用青云云平台块存储需要安装 [QingCloud-CSI 插件](#)，且需要有操作 [QingCloud IaaS 平台](#) 资源的权限，GlusterFS 和 CephRBD 需提前准备相关存储服务端。

1. [QingCloud-CSI](#) 插件已通过 KubeSphere 测试，仅需在 `vars.yml` 配置对应的参数，则 Installer 会根据配置项自动安装 QingCloud-CSI。
2. KubeSphere 测试过的存储服务端 [Ceph](#) Server 版本为 v0.94.10，[Ceph](#) 服务端集群部署可参考 [部署 Ceph 存储集群](#)，正式环境搭建 Ceph 存储服务集群请参考 [Install Ceph](#)。
3. KubeSphere 测试过的存储服务端 [GlusterFS](#) Server 版本为 v3.7.6，[GlusterFS](#) 服务端部署可参考 [部署 GlusterFS 存储集群](#)，正式环境搭建 GlusterFS 集群请参考 [Install Gluster](#) 或 [Gluster Docs](#) 并且需要安装 [Heketi 管理端](#)，Heketi 版本为 v3.0.0。
4. KubeSphere 安装过程中程序将会根据用户在 `vars.yml` 里选择配置的存储类型如 GlusterFS 或 Ceph RBD，进行自动化地安装对应 Kubernetes 集群所需的 GlusterFS Client 或 Ceph RBD Client，无需手动安装 Client。KubeSphere 自动安装的 GlusterFS Client 版本为 v3.12.10，可通过 `glusterfs -V` 命令查看，RBD Client 版本为 v12.2.5，可用 `rbd -v` 命令查看。
5. Kubernetes 集群中不可同时存在两个默认存储类型，若要指定默认存储类型前请先确保当前集群中无默认存储类型。

在您准备好存储服务端以后，只需要参考以下表中的参数说明，在 `conf` 目录下的 `vars.yml` 中，根据您存储服务端所支持的存储类型，在 `vars.yml` 的 `#QingCloud-CSI`、`# Ceph_rbd deployment`、`# GlusterFS provisioner deployment` 或 `# Local volume provisioner deployment(Only all-in-one)` 部分，参考脚本中的示例修改对应参数，即可完成 Kubernetes 集群存储类型的配置。以下对存储相关配置做简要说明（参数详解请参考 [storage classes](#)）：

Local Volume

Local Volume	Description
local_volume_provisioner_enabled	是否使用 local_volume 作为持久化存储, 是: true; 否: false
local_volume_provisioner_storage_class	storage_class 名称, 默认: local
local_volume_is_default_class	是否设定为默认 storage_class, 是: true; 否: false 注: 系统中存在多种 storage_class 时, 只能设定一种为 default_class

说明: 在您配置好 Local Volume (只有 all-in-one 支持这类存储) 并成功安装 KubeSphere 后, 请参考 [Local Volume 使用方法](#)。

QingCloud–CSI

[QingCloud–CSI](#) 块存储插件实现了 CSI 接口, 并且支持 KubeSphere 能够使用 QingCloud 云平台的存储资源。块存储插件部署后, 用户可创建访问模式 (Access Mode) 为单节点读写 (ReadWriteOnce) 的基于 QingCloud 的超高性能型 (超高性能型硬盘只能用在超高性能型主机) 、性能型 (性能型硬盘只能用在性能型主机) 或容量型硬盘的存储卷并挂载至工作负载。

QingCloud–CSI	Description
qy_csi_enabled	是否使用 QingCloud–CSI 作为持久化存储, 是: true; 否: false
qy_csi_is_default_class	是否设定为默认 storage_class, 是: true; 否: false 注: 系统中存在多种 storage_class 时, 只能设定一种为 default_class
qy_access_key_id , qy_secret_access_key	通过 青云控制台 的右上角账户图标选择 API 密钥 创建密钥获得
qy_zone	zone 应与 Kubernetes 集群所在区相同, CSI 插件将会操作此区的存储卷资源。例如: zone 可以设置为 sh1a (上海一区-A) 、sh1b (上

QingCloud-CSI	Description
	海1区-B) 、 pek2 (北京2区) 、 pek3 (北京3区) 、 pek3a (北京3区-A) 、 gd1 (广东1区) 、 gd2a (广东2区-A) 、 ap1 (亚太1区) 、 ap2a (亚太2区-A) 、
qy_host	QingCloud 云平台 api 地址, 例如 api.qingcloud.com (若对接私有云则以下值都需要根据实际情况填写)
qy_port	API 请求的端口, 默认 https 端口 (443)
qy_protocol	网络协议, 默认 https 协议
qy_uri	URI 路径, 默认值 iaas
qy_connection_retries	API 连接重试时间 (默认 3 秒)
qy_connection_timeout	API 连接超时时间 (默认 30 秒)
qy_type	<p>QingCloud 云平台硬盘的类型</p> <ul style="list-style-type: none"> * 性能型是 0 * 容量型是 2 * 超高性能型是 3 * 企业级分布式块存储 NeonSAN 是 5 * 基础型是 100 * SSD 企业型是 200 <p>详情见 QingCloud 官方文档</p>
qy_maxSize, qy_minSize	限制存储卷类型的存储卷容量范围, 单位为 GiB
qy_stepSize	设置用户所创建存储卷容量的增量, 单位为 GiB
qy_fsType	存储卷的文件系统, 支持 ext3, ext4, xfs. 默认为 ext4

Ceph RBD

Ceph_RBD	Description
ceph_rbd_enabled	是否使用 ceph_RBD 作为持久化存储, 是: true; 否: false
ceph_rbd_storage_class	storage_class 名称
ceph_rbd_is_default_class	是否设定为默认 storage_class, 是: true; 否: false

Ceph_RBD	Description
	注：系统中存在多种 storage_class 时，只能设定一种为 default_class
ceph_rbd_monitors	根据 Ceph RBD 服务端配置填写，可参考 Kubernetes 官方文档
ceph_rbd_admin_id	能够在存储池中创建 的客户端 ID， 默认: admin， 可参考 Kubernetes 官方文档
ceph_rbd_admin_secret	Admin_id 的 secret， 安装程序将会自动在 kube-system 项目内创建此 secret， 可参考 Kubernetes 官方文档
ceph_rbd_pool	可使用的 CephRBD 存储池， 可参考 Kubernetes 官方文档
ceph_rbd_user_id	用于映射 RBD 的 ceph 客户端 ID 默认: admin， 可参考 Kubernetes 官方文档
ceph_rbd_user_secret	User_id 的 secret， 需注意在所使用 rbd image 的项目内都需创建此 Secret， 可参考 Kubernetes 官方文档
ceph_rbd_fsType	kubernetes 支持的 fsType， 默认: ext4， 可参考 Kubernetes 官方文档
ceph_rbd_imageFormat	CephRBD 格式， 默认: "1"， 可参考 Kubernetes 官方文档
ceph_rbd_imageFeatures	当 ceph_rbd_imageFormat 字段不为 1 时需填写此字段， 可参考 Kubernetes 官方文档

注： 存储类型中创建 secret 所需 ceph secret 如 `ceph_rbd_admin_secret` 和 `ceph_rbd_user_secret` 可在 ceph 服务端通过以下命令获得：

```
$ ceph auth get-key client.admin
```

GlusterFS

GlusterFS（需提供 heketi 所管理的 glusterfs 集群）	Description
glusterfs_provisioner_enabled	是否使用 GlusterFS 作为持久化存储， 是: true； 否:

GlusterFS (需提供 heketi 所管理的 glusterfs 集群)	Description
	false
glusterfs_provisioner_storage_class	storage_class 名称
glusterfs_is_default_class	是否设定为默认 storage_class, 是: true; 否: false 注: 系统中存在多种 storage_class 时, 只能设定一种为 default_class
glusterfs_provisioner_restauthenabled	Gluster 启用对 REST 服务器的认证, 参数配置请参考 Kubernetes 官方文档
glusterfs_provisioner_resturl	Heketi 服务端 url, 参数配置请参考 Kubernetes 官方文档
glusterfs_provisioner_clusterid	Gluster 集群 id, 登录 heketi 服务端输入 heketi-cli cluster list 得到 Gluster 集群 id, 参数配置请参考 Kubernetes 官方文档
glusterfs_provisioner_restuser	能够在 Gluster pool 中创建 volume 的 Heketi 用户, 参数配置请参考 Kubernetes 官方文档
glusterfs_provisioner_secretName	secret 名称, 安装程序将会在 kube-system 项目内自动创建此 secret, 参数配置请参考 Kubernetes 官方文档
glusterfs_provisioner_gidMin	glusterfs_provisioner_storage_class 中 GID 的最小值, 参数配置请参考 Kubernetes 官方文档
glusterfs_provisioner_gidMax	glusterfs_provisioner_storage_class 中 GID 的最大值, 参数配置请参考 Kubernetes 官方文档
glusterfs_provisioner_volumetype	Volume 类型, 参数配置请参考 Kubernetes 官方文档
jwt_admin_key	heketi 服务器中 /etc/heketi/heketi.json 的 jwt.admin.key 字段

注: Glusterfs 存储类型中所需的 `glusterfs_provisioner_clusterid` 可在 glusterfs 服务端通过以下命令获得:

```
$ export HEKETI_CLI_SERVER=http://localhost:8080  
$ heketi-cli cluster list
```

集群节点扩容说明

安装 KubeSphere 后，在正式环境使用时可能会遇到服务器容量不足的情况，这时就需要添加新的服务器，然后将应用系统进行水平扩展来完成对系统的扩容。通过 KubeSphere 控制台，您可以根据实际业务需要对 Kubernetes 集群的 Node 节点进行扩容（暂不支持 Master 节点扩容），KubeSphere 对于在 Kubernetes 集群中加入新 Node 是非常简单的，仅需简单两步即可完成集群节点扩容。节点扩容基于 Kubelet 的自动注册机制，新的 Node 将会自动加入现有的 Kubernetes 集群中，以 root 用户增加 node3 的配置为例。

- 当申请完新的主机后，在主机配置文件 `conf/hosts.ini` 根据需要增加的主机信息在 [all] 和 [kube-node] 部分对应添加一行参数，若扩容多台主机则依次添加多行参数。需要注意的是，扩容新的节点时不能修改原节点的主机名如 master、node1 和 node2，如下添加新节点 node3：

```
[all]  
master ansible_connection=local ip=192.168.0.2  
node1 ansible_host=192.168.0.3 ip=192.168.0.3 ansible_ssh_pass=PASSWORD  
node2 ansible_host=192.168.0.4 ip=192.168.0.4 ansible_ssh_pass=PASSWORD  
node3 ansible_host=192.168.0.5 ip=192.168.0.5 ansible_ssh_pass=PASSWORD  
...  
[kube-node]  
master  
node1  
node2  
node3  
...
```

- 在 script 目录执行 `install.sh` 脚本，选择 `3). Cluster-scaling`。待扩容脚本执行成功后，即可看到包含新节点的集群节点信息，可通过 KubeSphere 控制台的菜单选择 **资源** 然后进入 **主机管理** 页面，或者通过 `Kubectl` 工具执行 `kubectl get node` 命令，查看扩容后的集群节点详细信息。同样，若需要停用或隔离集群中的节点，比如在硬件升级、硬件维护等情况下需要将某些 Node 进行隔

离，可以通过 KubeSphere 控制台菜单中选择 **资源** 进入 **主机管理** 页面执行停用或启用主机，可参考主机管理说明的 [停用或启用主机](#)。

组件版本信息

组件	版本
KubeSphere	1.0 Alpha (20180705.1800)
KubeSphere Console	Express Edition
Kubernetes	v1.10.5
OpenPitrix	v0.1.6

快速入门 – 部署 WordPress 博客网站

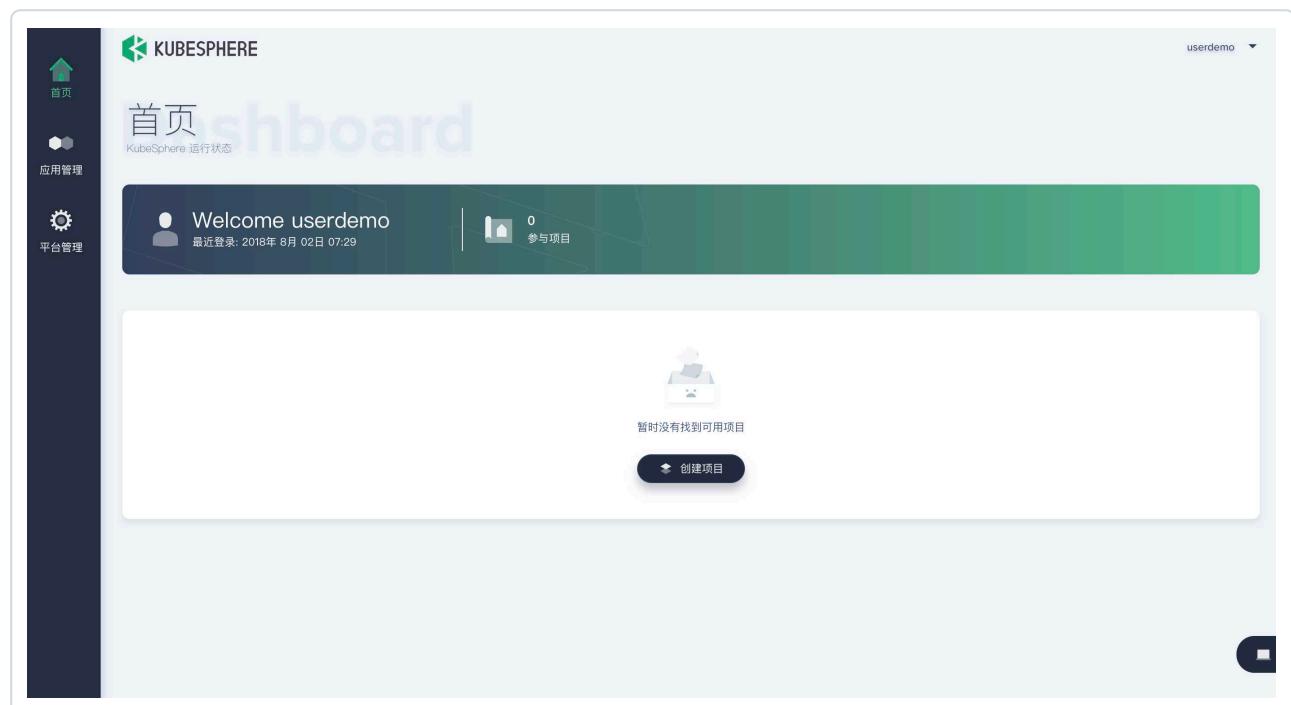
WordPress 是使用 PHP 语言开发的博客平台，用户可以在支持 PHP 和 MySQL 数据库的服务器上架设属于自己的网站，也可以把 WordPress 当作一个内容管理系统（CMS）来使用。在本指南中，将引导用户通过 KubeSphere 控制台部署一个后端为 MySQL 数据库的 WordPress 博客网站，帮助您快速入门 KubeSphere。

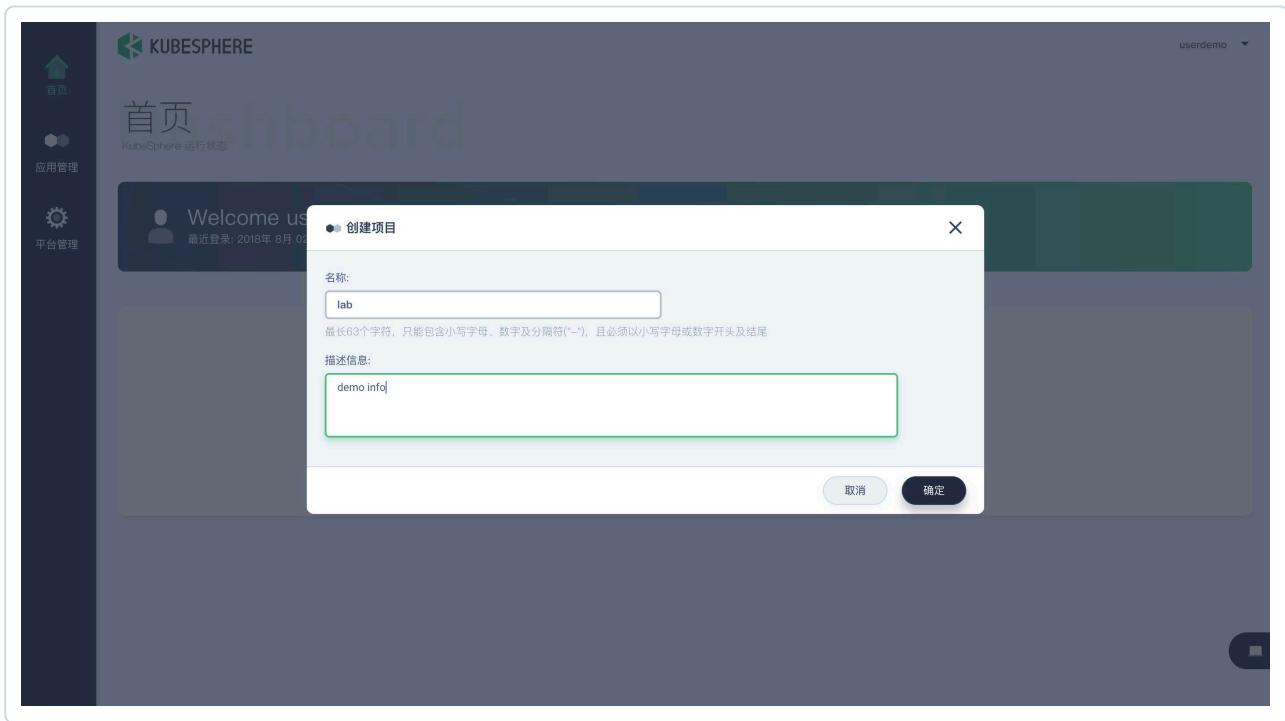
登录 KubeSphere

1、在开始实践之前，请以普通用户（Regular）的身份登录 KubeSphere，用户身份需要通过管理员创建。关于如何管理角色请参考 [角色管理说明](#)，关于成员管理请参考 [用户管理说明](#)。

创建项目

2、登录后，通过首页或项目管理创建项目。关于项目管理的详细介绍请参考 [项目管理说明](#)。





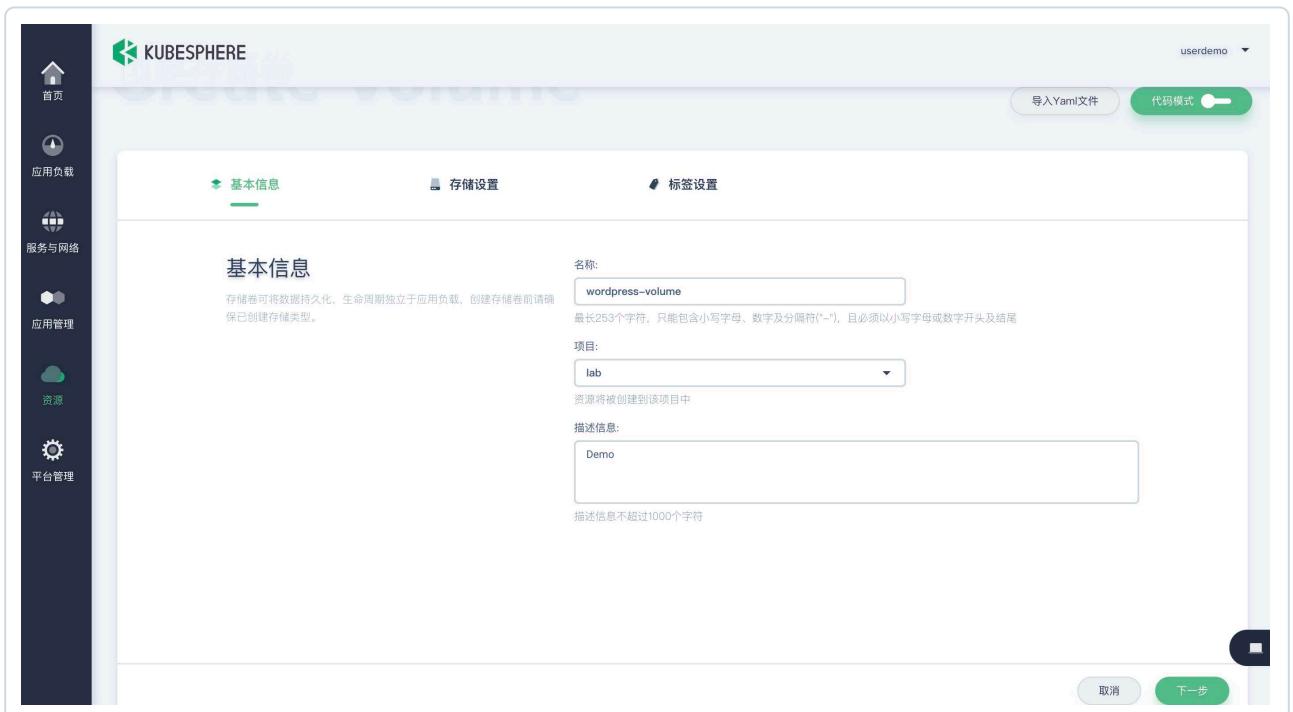
创建存储卷

3、在菜单栏的资源中选择存储卷，点击创建存储卷，按如下步骤分别为 WordPress 和 MySQL 数据库创建存储卷，可命名为 `wordpress-volume` 和 `mysql-volume`。关于存储卷管理和使用的详细介绍请参考 [存储卷使用说明](#)。



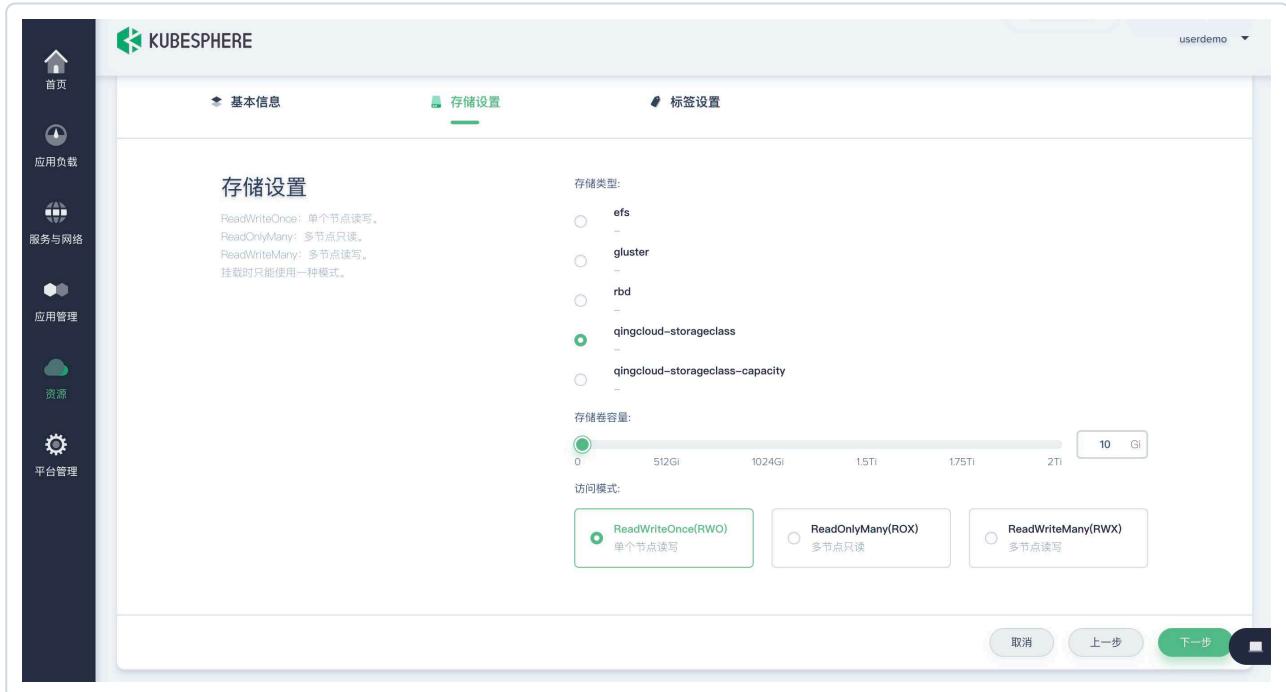
4、创建 WordPress 存储卷 `wordpress-volume` 需填写基本信息、配置存储设置和标签设置，请参考以下步骤：

- 4.1. 填写存储卷基本信息，完成后点下一步：

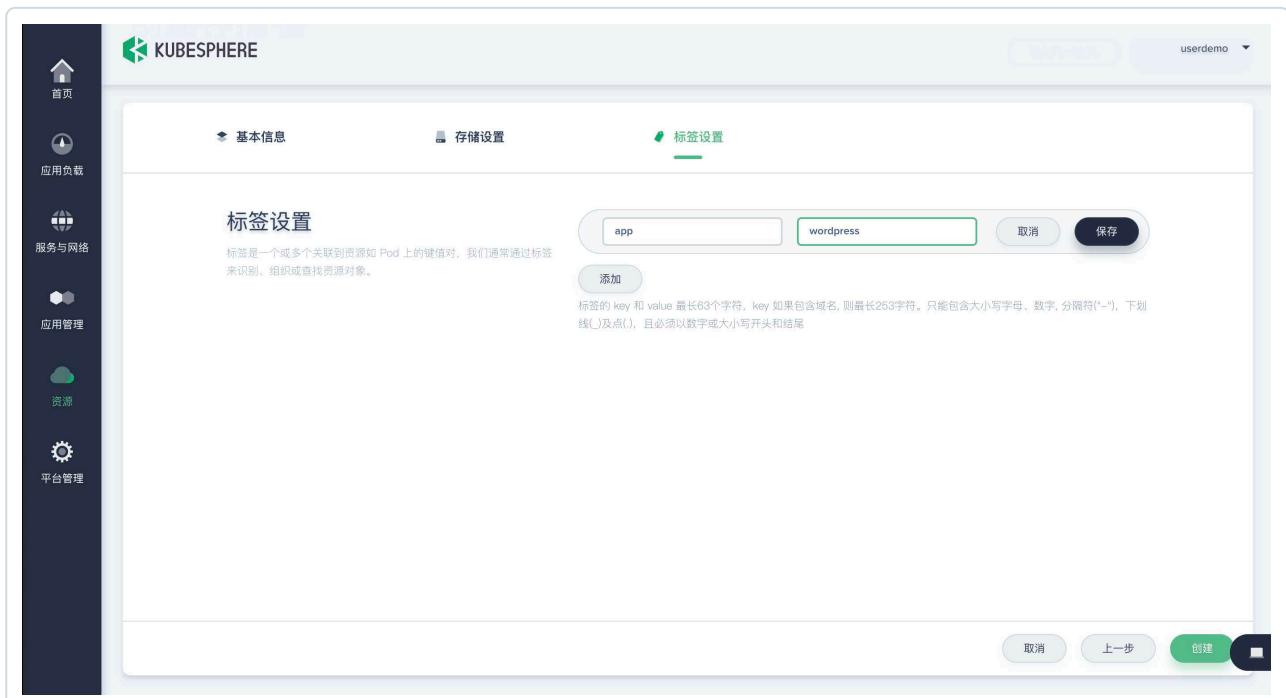


- 4.2. 选择存储类型 (请参考 [存储类型管理说明](#) 通过管理员创建存储类型)，设置存储大小和读写模

式，完成后点下一步：



- 4.3. 填写标签并保存，点击创建。在存储卷列表页，即可看到 WordPress 所需的存储卷创建成功（刚完成创建时存储卷状态为 Pending 是正常的，等待数秒后状态将自动更新为 Bound）：



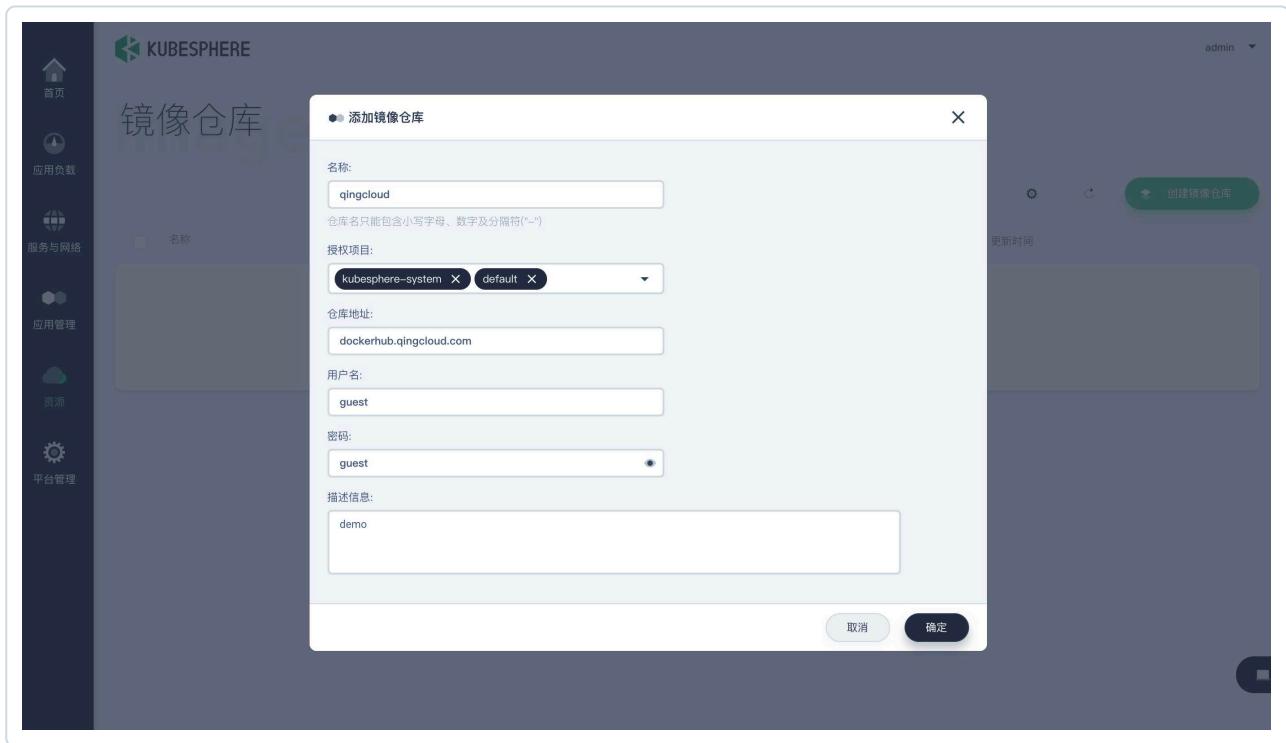
- 5、同上，创建 MySQL 所需存储卷 mysql-volume，参考上述步骤完成基本信息、存储设置和标签设置。至此，WordPress 和 MySQL 所需的存储卷都创建成功：

The screenshot shows the KubeSphere storage volume management interface. On the left is a sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area is titled '存储卷' (Storage Volumes) and shows '共 2 个存储卷' (2 storage volumes). A search bar at the top right contains the placeholder '输入查询条件进行过滤' (Filter by query conditions). Below the search bar is a toolbar with a refresh icon, a back arrow, and a green button labeled '创建存储卷' (Create Storage Volume). A dropdown menu above the table is set to 'lab'. The table lists two storage volumes:

	名称	状态	容量	访问模式	挂载状态	项目	创建时间	创建者
<input type="checkbox"/>	mysql-volume qingcloud-storagecl...	Bound	10Gi	RWO	未挂载	lab	2018年 8月 03日 23:28:47	userdemo
<input type="checkbox"/>	wordpress-volu... qingcloud-storagecl...	Bound	10Gi	RWO	未挂载	lab	2018年 8月 03日 23:25:27	userdemo

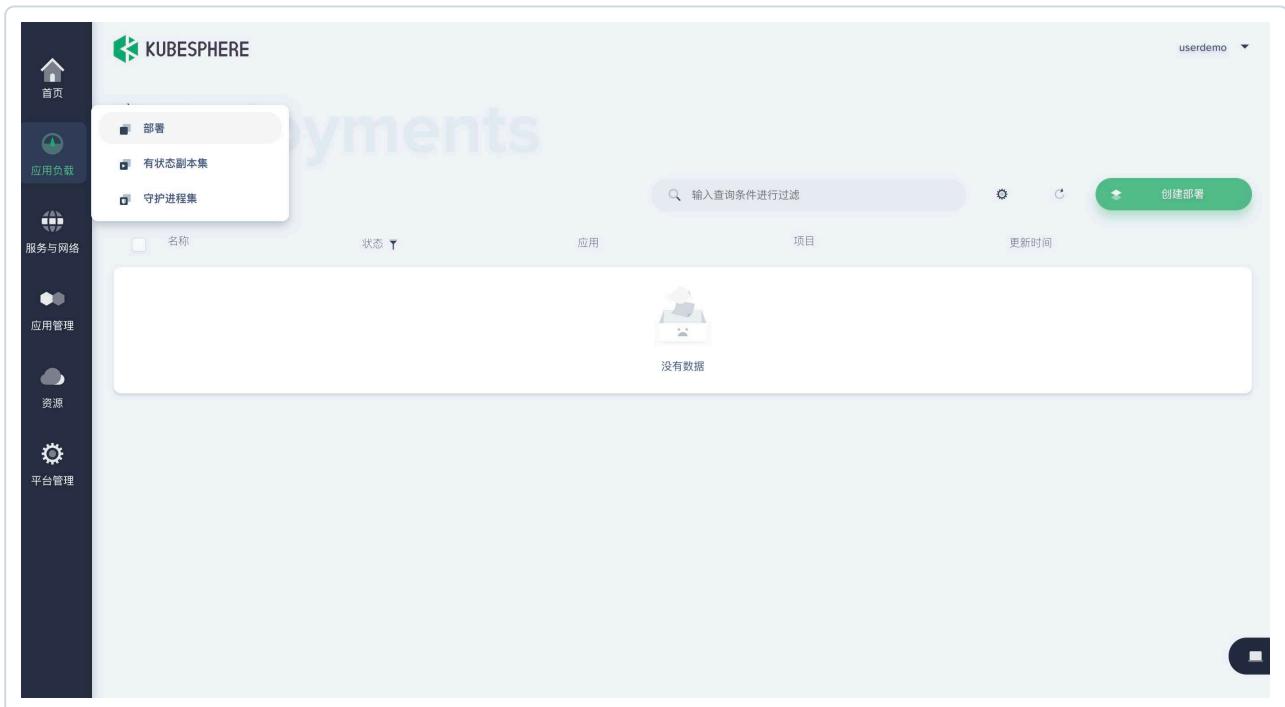
创建镜像仓库

6、创建镜像仓库需要通过管理员为其创建。在菜单栏的资源下选择镜像仓库，然后点击右上角创建镜像仓库按钮，弹出添加镜像仓库对话框，填写镜像仓库所需要的信息。在创建部署的容器组设置时需要填写镜像名称，将从当前添加的镜像仓库中拉取镜像。用户可以在 KubeSphere 创建 QingCloud 镜像仓库、Docker Hub 和 Harbor 镜像仓库，详细可参考 [镜像仓库管理](#)。此处以添加 QingCloud 镜像仓库 dockerhub.qingcloud.com 为例，参考如下截图完成 QingCloud 镜像仓库的添加：

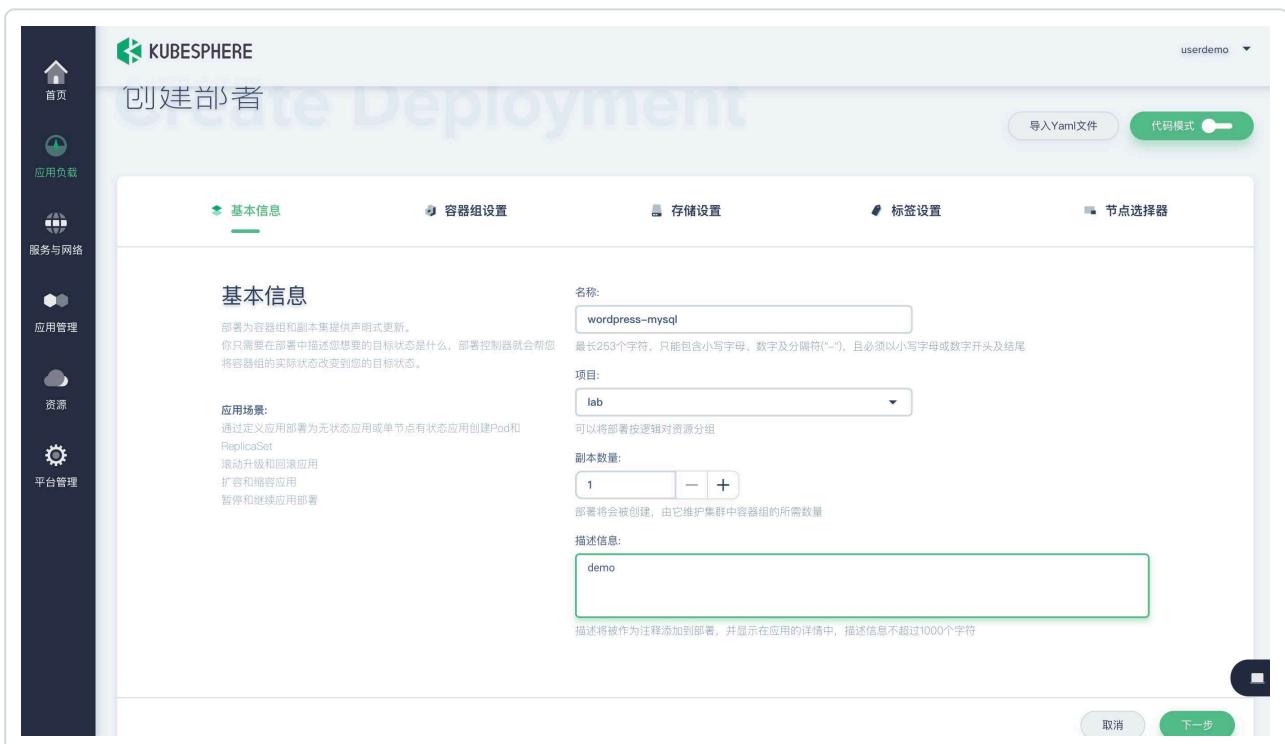


创建部署

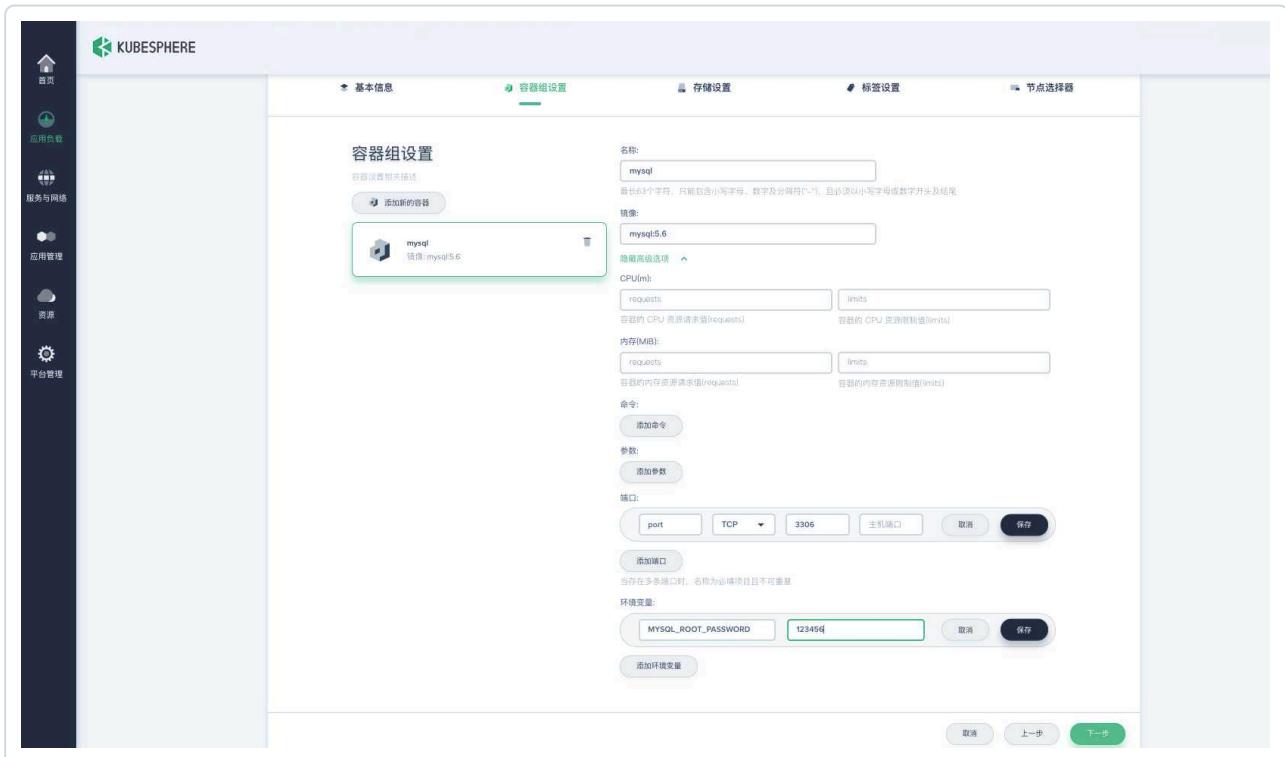
7、在菜单栏的应用负载中选择部署，点击创建部署，按照如下步骤分别为 WordPress 和 MySQL 数据库创建部署资源，可命名为 `wordpress` 和 `wordpress-mysql`。关于管理部署资源的详细介绍请参考 [部署管理说明](#)。



- 7.1. 填写创建 wordpress-mysql 部署的基本信息，完成后点下一步：



命令和启动参数。在容器组设置中配置 MySQL 的访问端口（3306）和 MySQL 的环境变量 `MYSQL_ROOT_PASSWORD` 即 root 用户的密码，**端口**用于指定容器需要暴露的端口，端口协议此处选择 TCP，主机端口是容器映射到主机上的端口，此处暂不设置主机端口，**环境变量**可以指定容器内部使用的环境变量。完成后点下一步：



注: 如果 docker 镜像不是来自默认的 Docker Hub, 请参考 [镜像仓库管理说明](#)。

- 7.3. 存储设置中, 点击添加存储卷并选择已有存储卷, 然后选择之前创建好的 mysql-volume 存储卷:

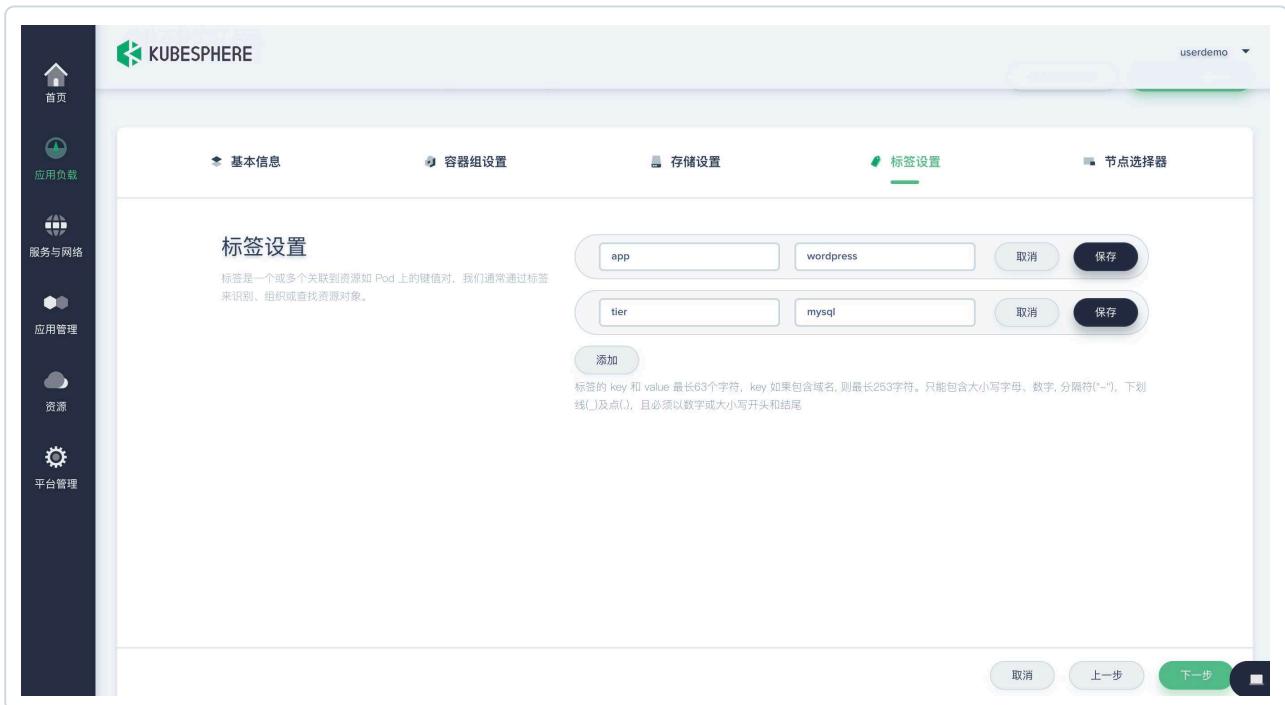
注: 如果安装 KubeSphere 使用的是 all-in-one 部署在单个主机, 使用的存储类型是 Local Volume, 请参考存储卷使用说明的 [Local Volume 使用方法](#)。如果集群配置的存储服务端是 Ceph RBD, 则需要通过 Kubectl 创建 Secret, 若遇到 Ceph RBD 存储卷挂载至工作负载时因缺少密钥无法挂载, 请参考存储卷使用说明的 [Ceph RBD 无法挂载解决方案](#)。

The screenshot shows the 'Storage Volume' creation interface in KubeSphere. On the left sidebar, there are icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main panel has tabs for Basic Information, Container Group Settings, Storage Settings (which is selected), Label Settings, and Node Selector. The Storage Settings tab displays two existing storage volumes: 'mysql-volume' (status: Available, capacity: 10Gi) and 'wordpress-volume' (status: Available, capacity: 10Gi). A green box highlights the 'mysql-volume' entry. Below the existing volumes, there is a section for adding a new volume, with a button labeled 'Add Storage Volume'.

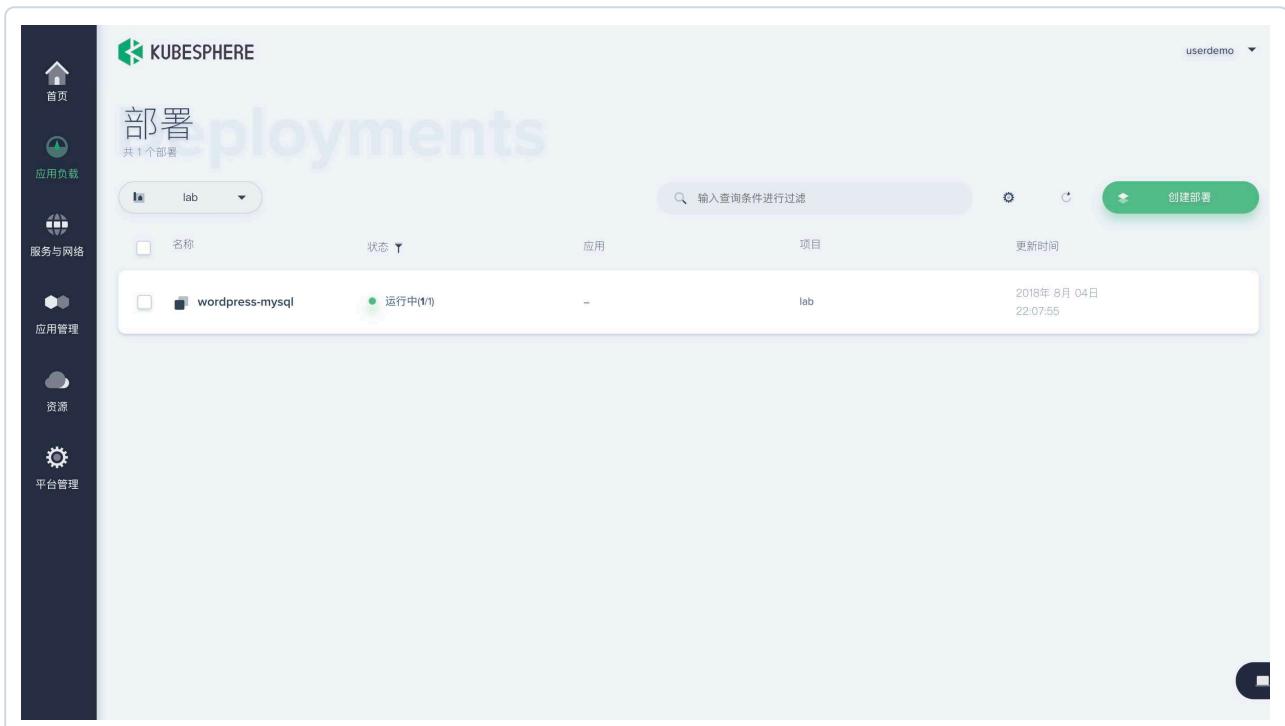
- 填写存储卷在容器中的挂载路径，填写后点击挂载，选择下一步：

This screenshot shows the continuation of the storage volume creation process. The 'Storage Settings' tab is still active. In the 'Mount Path' field, 'mysql' is entered in the container name field, 'ReadWrite' is selected in the access mode dropdown, and the path '/var/lib/mysql' is specified in the mount path field. A green box highlights the '挂载' (Mount) button. At the bottom right, there are buttons for 'Cancel', 'Previous Step', and 'Next Step'. The 'Next Step' button is highlighted with a green box.

- 7.4. 设置标签并保存，选择下一步：



- 7.5. 节点选择器可将容器组调度到期望运行的节点上，节点选择器可根据实际情况配置。此处不作配置，选择创建，则 wordpress-mysql 部署创建成功（刚完成创建时部署状态为“更新中”是正常的，系统为其拉取镜像和调度资源需要时间，等待数秒后状态将自动更新为运行中）：

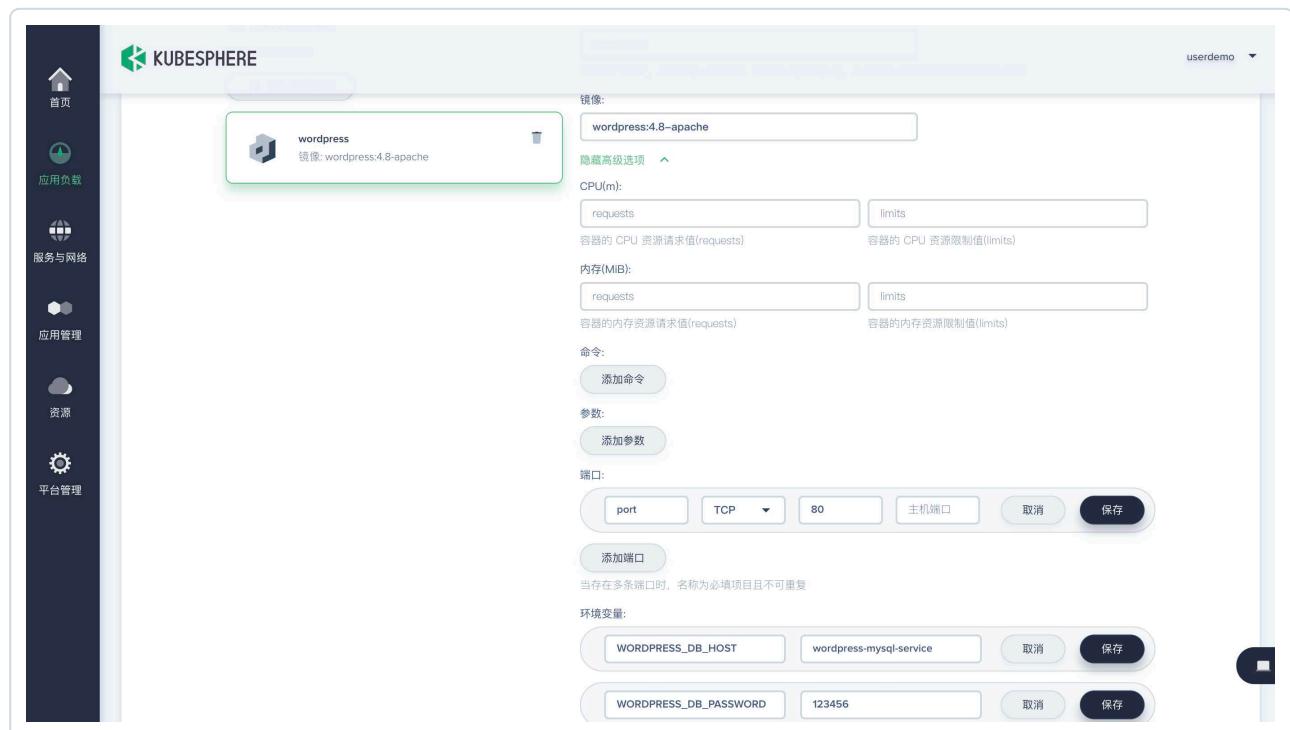


注: MySQL 数据库还可通过创建有状态副本集的方式来创建。

8、同上, 请参考以下步骤创建部署 wordpress :

- 8.1. 填写创建部署的基本信息, 本示例创建的 WordPress 部署名称 `wordpress`, 选择之前创建的项目 `lab`, 副本数为 `1`, 描述信息可自定义, 完成后点下一步。
- 8.2. 填写容器组设置, 名称可自定义, 镜像填写 `wordpress:4.8-apache`, 配置 WordPress 的容器需要暴露的端口: 80 端口(暂不设置主机端口)和关联 MySQL 数据库的环境变量(`WORDPRESS_DB_HOST` 和 `WORDPRESS_DB_PASSWORD`)并保存, CPU 和内存暂不作限定, 参考如下配置, 完成后点下一步。

注意: 环境变量中, `WORDPRESS_DB_HOST` 的值对应的是 MySQL 服务的名称, 在后续步骤创建 MySQL 服务时, 服务名应该与此处的环境变量值相同, 否则无法连接 MySQL 数据库。



- 8.3. 存储设置中, 点击添加存储卷并选择已有存储卷, 然后选择之前创建好的 `wordpress-volume` 存储卷, 填写存储卷在容器中的挂载路径, 本示例中选择 `读写`, 挂载点为 `/var/www/html`, 填写后点击挂载, 完成后选择下一步。
- 8.4. 设置标签并保存, 本示例标签的键值对设置如下, 完成后选择下一步。

```
app=wordpress  
tier=frontend
```

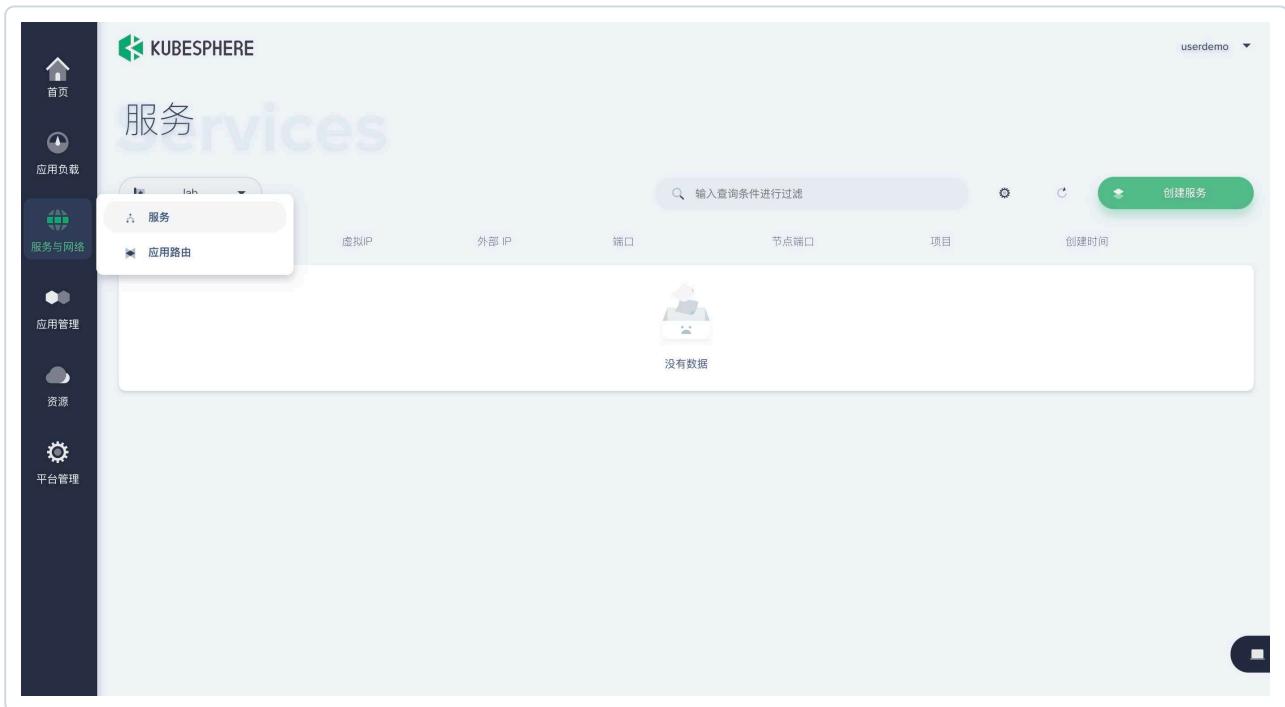
- 8.5. 节点选择器此处暂不作配置，点击创建，则 WordPress 部署创建成功。至此，WordPress 和 MySQL 部署都创建成功：

The screenshot shows the KubeSphere deployment management interface. On the left, there is a sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area is titled "部署" (Deployments) and shows a list of 2 deployments. The table has columns for Name, Status, Application, Project, and Last Updated. The first deployment is "wordpress" (Status: 运行中(1/1), Application: -, Project: lab, Last Updated: 2018年 8月 05日 12:11:24). The second deployment is "wordpress-mysql" (Status: 运行中(1/1), Application: -, Project: lab, Last Updated: 2018年 8月 04日 22:07:55). There is a search bar at the top right and a green "Create Deployment" button.

名称	状态	应用	项目	更新时间
wordpress	运行中(1/1)	-	lab	2018年 8月 05日 12:11:24
wordpress-mysql	运行中(1/1)	-	lab	2018年 8月 04日 22:07:55

创建服务

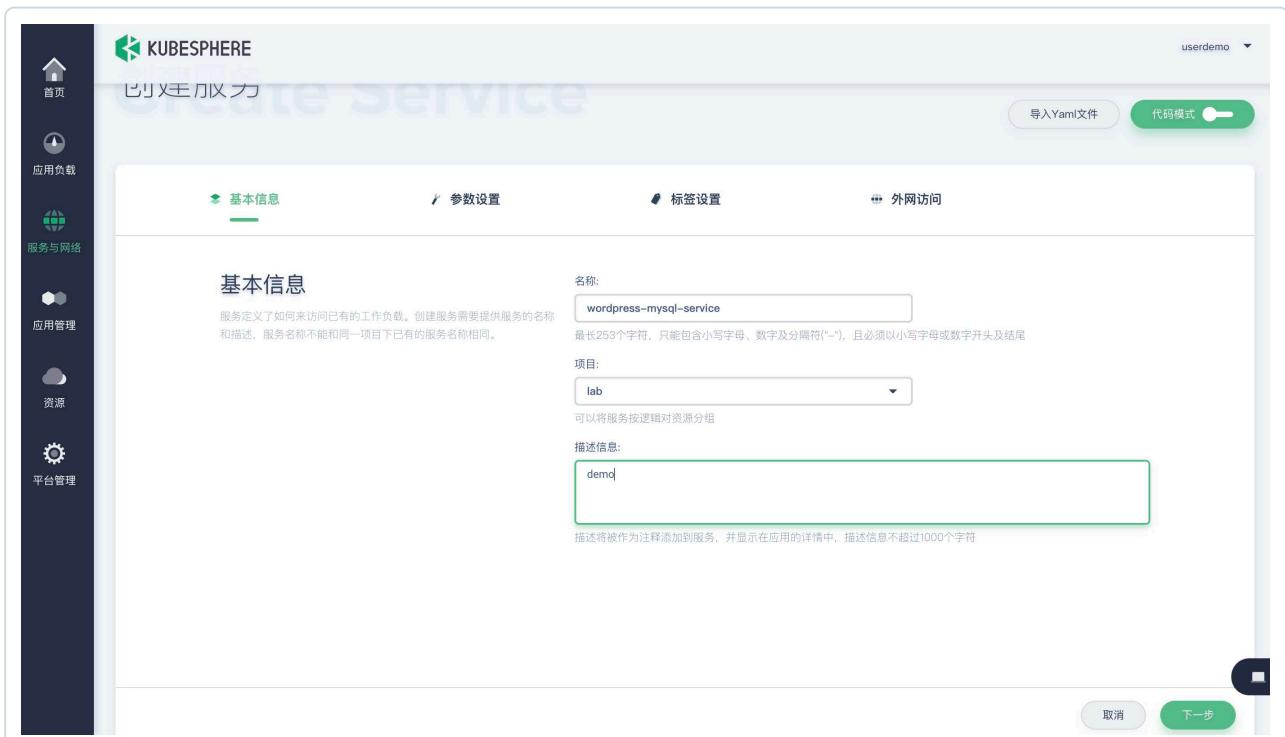
9、通过服务的方式，可以将部署的资源暴露出去以供访问，以下步骤将分别介绍如何暴露 MySQL 和 WordPress 供外部访问。在左侧菜单栏处选择服务与网络 → 服务，点击创建服务。关于管理服务的详细介绍请参考 [服务管理说明](#)。



10、请参考以下步骤为 MySQL 数据库创建服务：

- 10.1. 填写基本信息，注意服务名称要和 8.2 创建部署 WordPress 的环境变量

WORDPRESS_DB_HOST 的值保持一致，然后选择下一步：



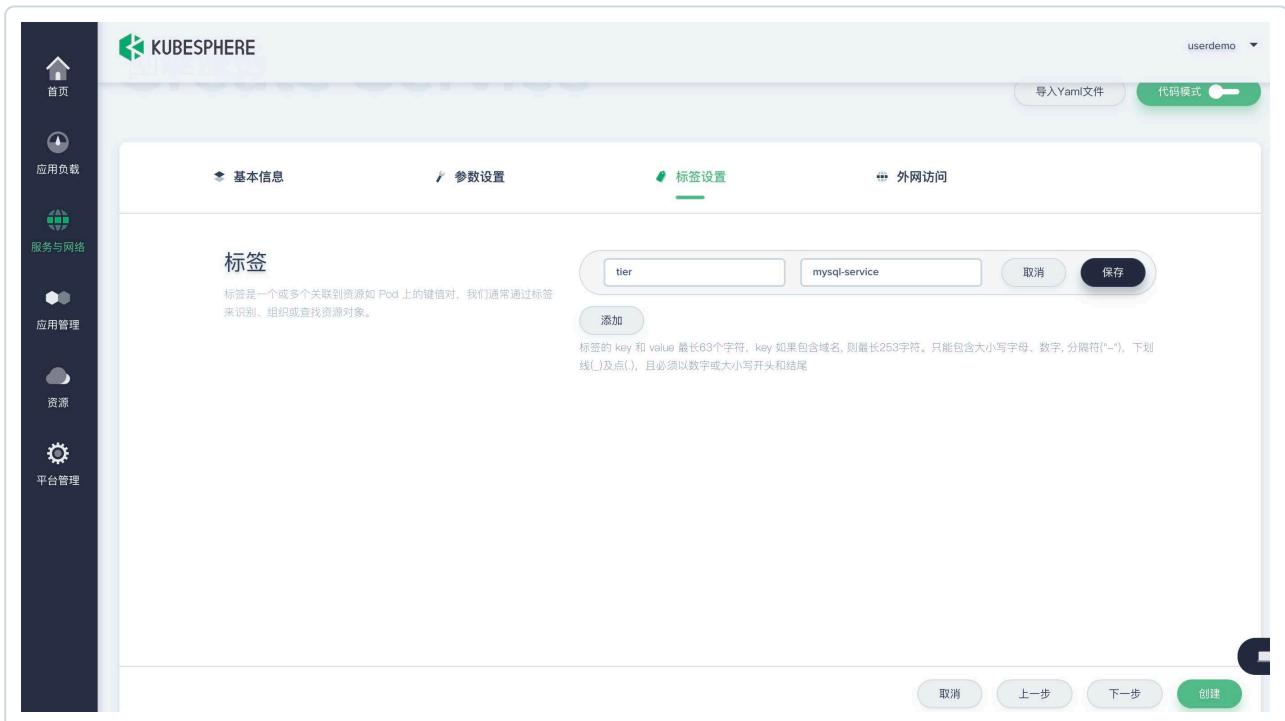
- 10.2. 参考以下截图完成参数设置。其中服务类型包括 Virtual IP、Headless (selector) 和

Headless (externalname) 三种，此处我们选择 Virtual IP，选择器一栏选择已创建的部署：wordpress-mysql，其中，第一个端口是需要暴露出去的服务端口，第二个端口（目标端口）是容器端口，此处的 MySQL 服务的端口和目标端口都填写 TCP 协议的 3306 端口，选择下一步：

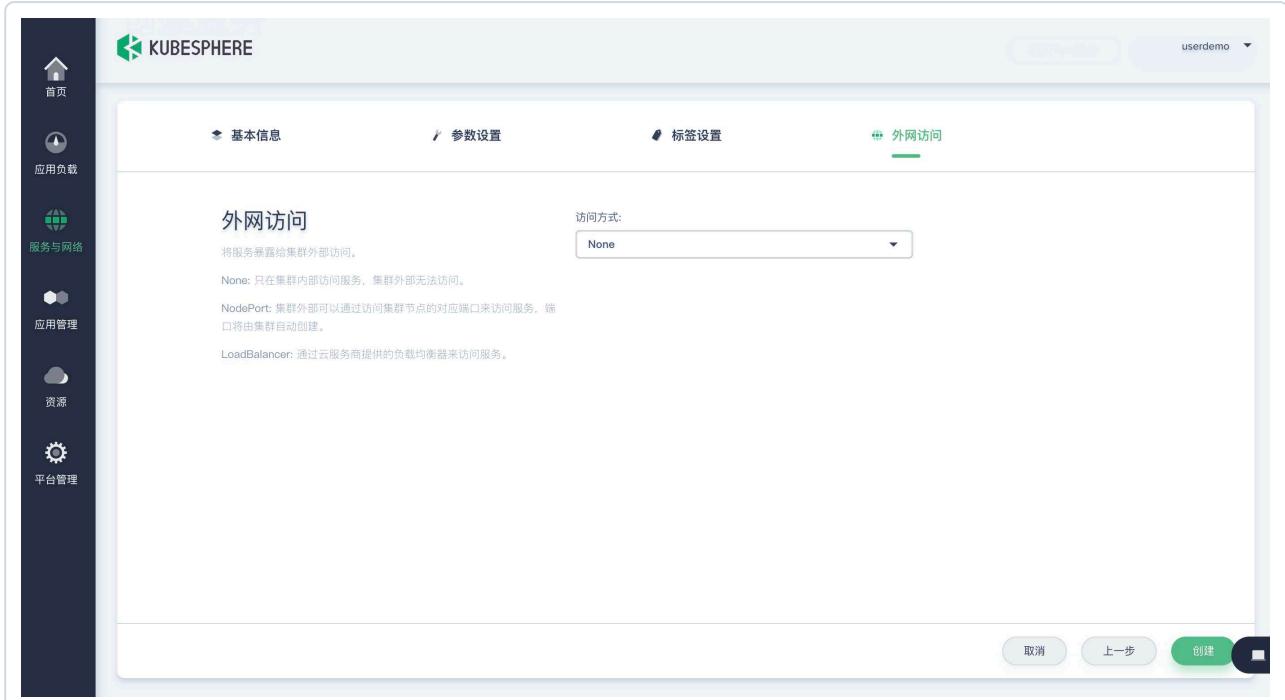
说明: 若要实现基于客户端 IP 的会话亲和性，可以在会话亲和性下拉框选择 "ClientIP" 或在代码模式将 service.spec.sessionAffinity 的值设置为 "ClientIP"（默认值为 "None"），该设置可将来自同一个 IP 地址的访问请求都转发到同一个后端 Pod。

The screenshot shows the KubeSphere interface for creating a new service. The left sidebar has icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area is titled '参数设置' (Parameter Settings). It includes tabs for 基本信息 (Basic Information), 参数设置 (Parameter Settings, which is selected), 标签设置 (Label Settings), and 外网访问 (External Access). The '参数设置' tab contains fields for Type (Virtual IP), Selector (部署 wordpress-mysql), Port (port, TCP, 3306), Target Port (3306), Session Affinity (None), and buttons for 确定 (Confirm) and 取消 (Cancel). At the bottom are buttons for 取消 (Cancel), 上一步 (Previous Step), and a green 下一步 (Next Step).

- 10.3. 添加标签并保存，选择下一步：



- 10.4. 在设置外网访问中，共有 None、NodePort、LoadBalancer 三种访问方式，可根据情景来设置访问方式。由于 MySQL 数据库不需要暴露服务给外网，因此选择 None，即可完成创建：

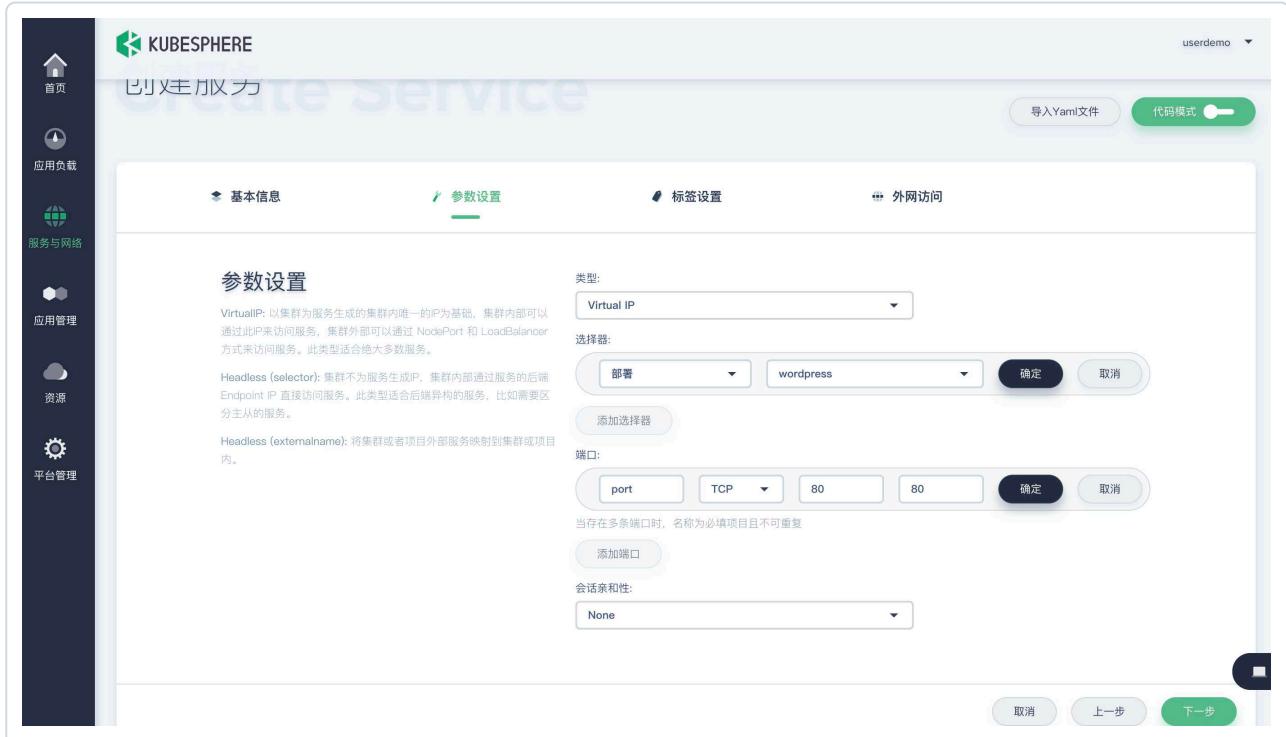


11、同上，请参考以下步骤为 WordPress 创建服务：

- 11.1. 填写基本信息，本示例服务名称为 `wordpress-service`，选择之前创建好的项目 `lab`，描述信

息可自定义，完成后选择下一步。

- 11.2. 参考以下参数，其中类型选择 Virtual IP，选择器选择之前创建好的 `wordpress` 部署，此处的 `wordpress` 服务的端口和目标端口都填写 `TCP` 协议的 `80` 端口，完成参数设置，选择下一步。



- 11.3. 添加标签并保存，本示例标签设置如下，添加后选择下一步。

```
tier=wordpress-service
```

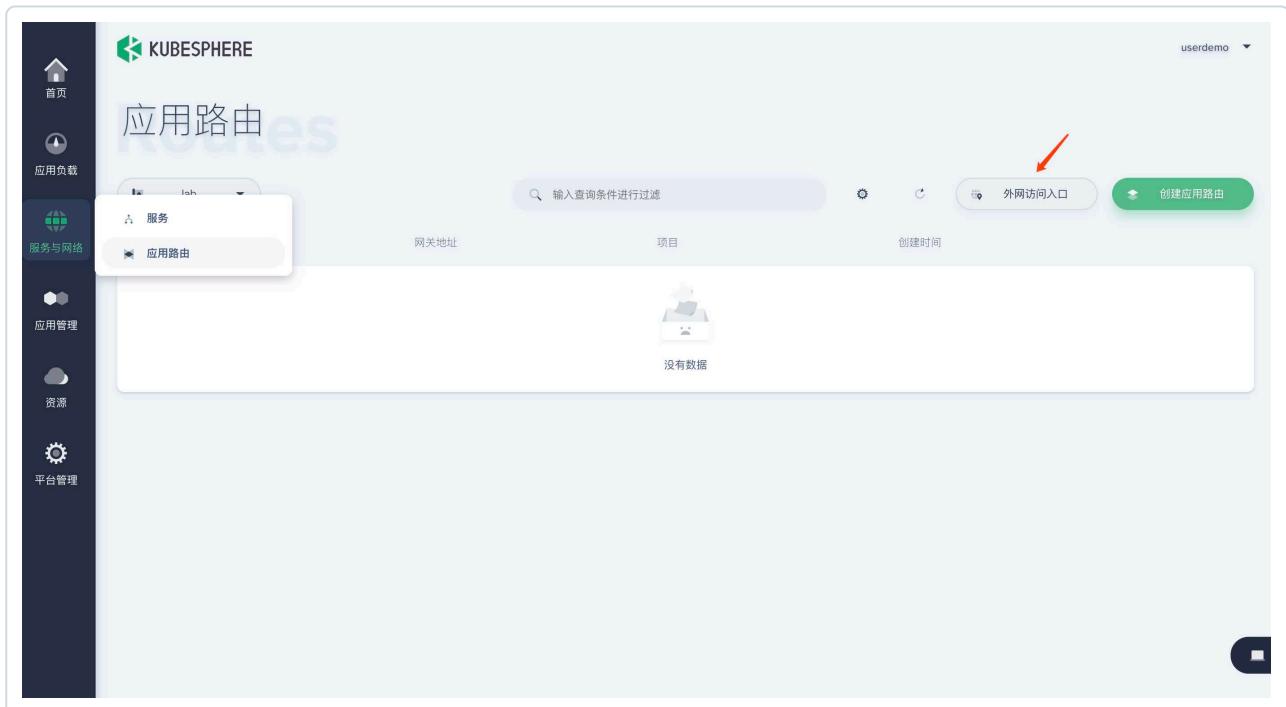
- 11.4. 设置外网访问时，我们将以应用路由的方式暴露服务给外网，所以此处也选择 `None`。至此，`WordPress` 与 `MySQL` 服务都已经创建成功。

名称	虚拟IP	外部IP	端口	节点端口	项目	创建时间
wordpress-service Virtual IP	10.96.127.15		80:80/TCP		lab	2018年 8月 15日 17:34:30
wordpress-mysql-... Virtual IP		10.96.170.212		3306:3306/TCP	lab	2018年 8月 06日 14:41:57

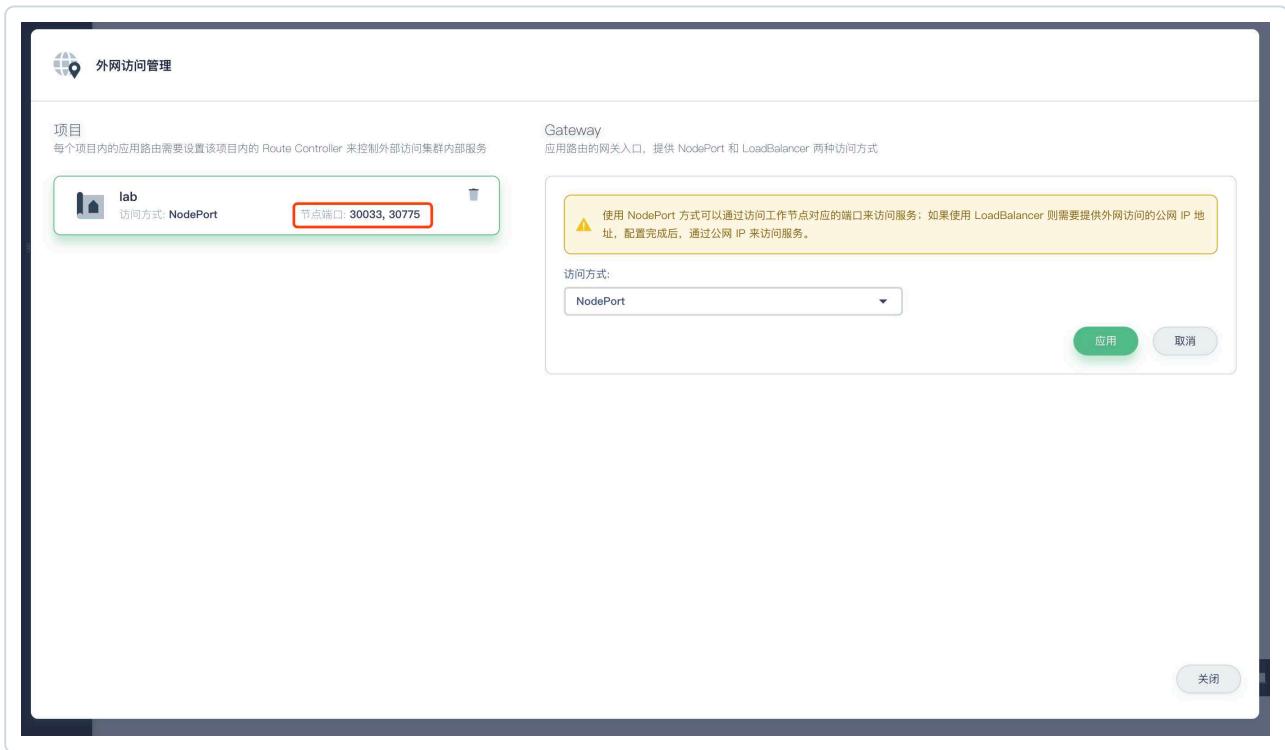
创建应用路由

12、通过创建应用路由的方式可以将 WordPress 暴露出去供外网访问，与将服务直接通过 NodePort 或 LoadBalancer 暴露出去不同之处是应用路由是通过配置 Hostname 和路由规则（即 ingress）来访问，请参考以下步骤配置应用路由。关于如何管理应用路由详情请参考 [应用路由管理说明](#)。

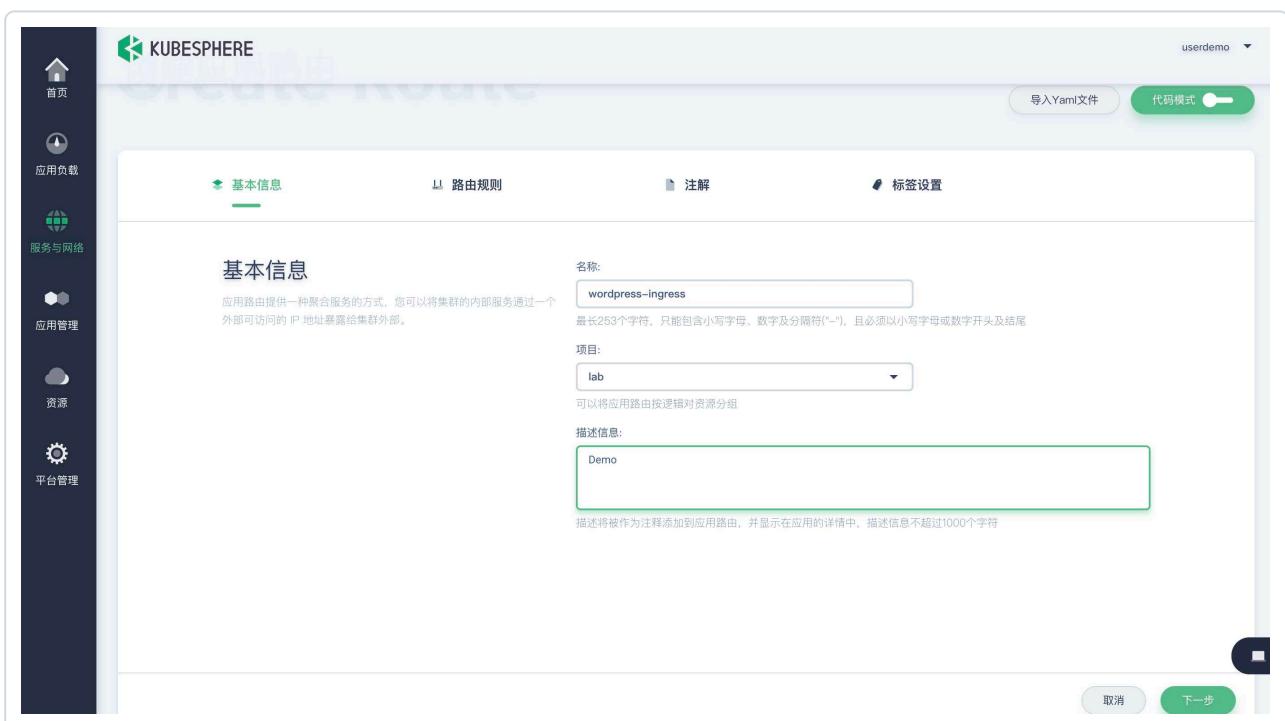
- 12.1. 首先配置外网访问入口，即应用路由的网关入口，每个项目都有一个独立的网关入口：



- 网关入口提供 NodePort 和 LoadBalancer 两种访问方式，如果用 LoadBalancer 的方式暴露服务，需要有云服务厂商的 LoadBalancer 插件支持，比如 [青云QingCloud KubeSphere 托管服务](#) 可以将公网 IP 地址的 [ID](#) 填入 Annotation 中，即可通过公网 IP 访问该服务。（如果外网访问方式设置的是 LoadBalancer，可以参考 [访问应用路由说明](#) 的 LoadBalancer 方式。）
- 本实践以 NodePort 访问方式为例配置网关入口，此方式网关可以通过工作节点对应的端口来访问，配置完成后点击 **应用**（左边节点端口生成的两个端口，分别是 HTTP 协议的 80 端口和 HTTPS 协议的 443 端口），然后点击关闭：



- 12.2. 创建应用路由，填入基本信息：



- 12.3. 配置路由规则，这里以 `kubesphere.wp.com` 为例，并且 path 选择之前的创建成功的服务 `wordpress-service`，选择下一步：

- Hostname: 应用规则的访问域名，最终使用此域名来访问对应的服务。(如果访问入口是以 NodePort 的方式启用，需要保证 Hostname 能够在客户端正确解析到集群工作节点上；如果是以 LoadBalancer 方式启用，需要保证 Hostname 正确解析到负载均衡器的 IP 上)。
- Paths: 应用规则的路径和对应的后端服务，端口需要填写成服务的端口。

- 12.4. 添加注解，Annotation 是用户任意定义的“附加”信息，以便于外部工具进行查找，参见 [Annotation 官方文档](#)。本示例注解如下所示，完成后选择下一步：

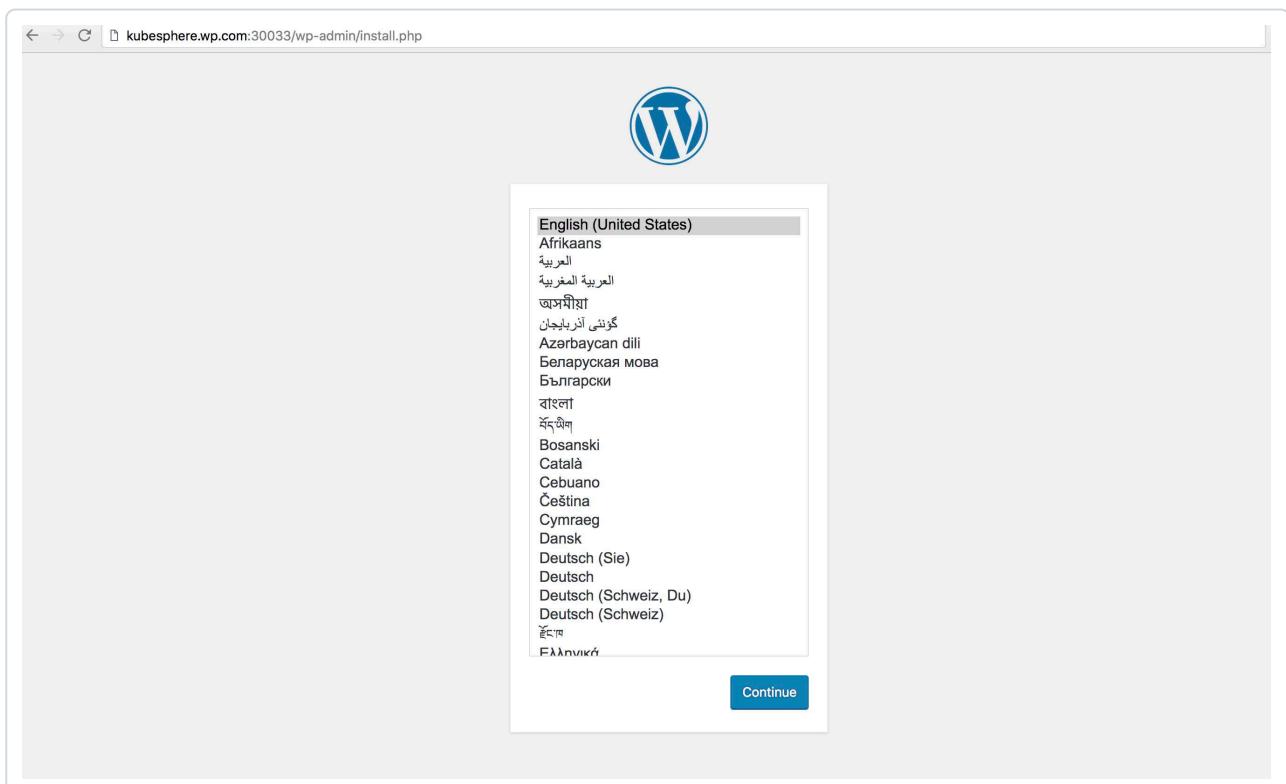
```
hostname=kubesphere.wp.com
```

- 12.5. 添加标签如下，选择创建：

```
tier=wordpress-ingress
```

- 12.6. 至此，WordPress 就以应用路由的方式通过网关入口暴露到外网，用户可以通过示例中配置的 `kubesphere.wp.com` 和端口号访问 WordPress 博客网站 (此示例中用的是第一个端口 30033，它对应的 HTTP 协议的 80 端口，由于目前应用路由中仅支持 HTTP 协议，将在下个版本中支持 HTTPS 协议)：

注: 创建应用路由之后应该把主机的公网 IP 和 配置的域名如: 139.198.17.33 kubesphere.wp.com 填入本地的 hosts 配置文件中, 即可通过浏览器访问。如果主机的公网 IP 有防火墙, 应在防火墙放行对应的端口, 否则外网无法访问。



副本数调节

在实际生产系统中, 我们经常会遇到某个服务需要扩容的场景, 也可能会遇到由于资源紧张或者工作负载降低而需要减少服务实例数的场景。在 `kubectl` 中我们可以利用命令 `kubectl scale rc` 来完成这些任务。还可以通过在 KubeSphere 控制台资源详情页中左下角处的容器组数量加减按钮, 来调整容器组数量, 进而完成对 Pod 的横向伸缩。

KUBESPHERE

← 部署 / wordpress

wordpress demo

资源信息 事件

容器组

标签

app=wordpress tier=frontend

详情

项目: lab

应用:

副本数量: 1 desired | 1 total | 1 updated | 1 available | 0 unavailable

状态: 运行中(1/1)

选择器: app = wordpress, tier = frontend

策略类型: 滚动更新

滚动更新: maxUnavailable=1, maxSurge=25%

创建时间: 2018年 8月 02日 08:55

容器组数量
1个容器组正在运行

+

删除资源

在实践完成后，建议删除不需要的部署和服务资源，在资源列表中勾选需要删除的资源，例如下图中点击删除部署按钮可删除选中的部署资源，详细可参考用户指南中的 [应用负载管理](#)、[服务与网络](#) 和 [资源管理说明](#) 释放资源。

The screenshot shows the KubeSphere deployment management interface. On the left, there is a vertical sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area is titled "部署 Deployments" and shows a list of 2 deployments. The table has columns for Name, Status, Application, Project, and Last Updated. Two rows are listed:

名称	状态	应用	项目	更新时间
wordpress-mysql	运行中(1/1)	-	lab	2018年 8月 02日 09:16:14
wordpress	运行中(1/1)	-	lab	2018年 8月 02日 08:55:31

Debug

- 在创建资源后如果遇到报错或资源一直处于 Pending 状态，可通过资源详情页中的事件页面和容器日志查看报错消息，进而排查问题的原因。例如下图中，由于 wordpress 部署资源中的容器组状态显示 CrashLoopBackOff，因此可以通过事件详情页中查看具体的错误消息，进一步针对性地解决问题。

KUBESPHERE

wordpress-57c65f8f5b-tc7k8

容器组

标签: app=wordpress, pod-template-hash=1372194916, tier=frontend

详情

项目: demo-case
状态: CrashLoopBackOff
主机名称: i-ohb602zs
主机 IP: 192.168.0.3
重启次数(总计): 3

资源信息 事件 ↑

原因	状态	消息
FailedAttachVolume me 2018年 8月 03日 08:46:03	异常中	Multi-Attach error for volume "pvc-35a48a6e-9599-11e8-b000-5254739bd60e". Volume is already exclusively attached to one node and can't be attached to another.
Scheduled 2018年 8月 03日 08:46:03	Normal	Successfully assigned wordpress-57c65f8f5b-tc7k8 to i-ohb602zs
SuccessfulMount Volume 2018年 8月 03日 08:46:05	Normal	MountVolume.SetUp succeeded for volume "default-token-v7qbd"
SuccessfulAttach Volume 2018年 8月 03日 08:46:05	Normal	AttachVolume.Attach succeeded for volume "pvc-35a48a6e-9599-11e8-b000-5254739bd60e".

- 若服务创建后无法访问或外部 IP 一直处于 Pending 状态，可通过服务详情页中查看应用负载和容器组的运行状态，如果运行都正常再查看事件页面检查是否有报错消息。例如下图中，由于公网 IP 与 KubeSphere 主机集群不在一个 region 导致无法解析到该公网 IP，因此服务无法暴露出去。

KUBESPHERE

服务 / wordpress-service

wordpress-service

标签: tier=wordpress-service

详情

项目: lab
类型: Virtual IP
虚拟IP: 10.96.232.79
外部 IP:
节点端口: 80:31035/TCP
会话亲和性: None
选择器: app = wordpress, tier = frontend
DNS: wordpress-service.lab
Endpoints: 10.244.4.37:80
创建时间: 2018年 8月 06日 14:56

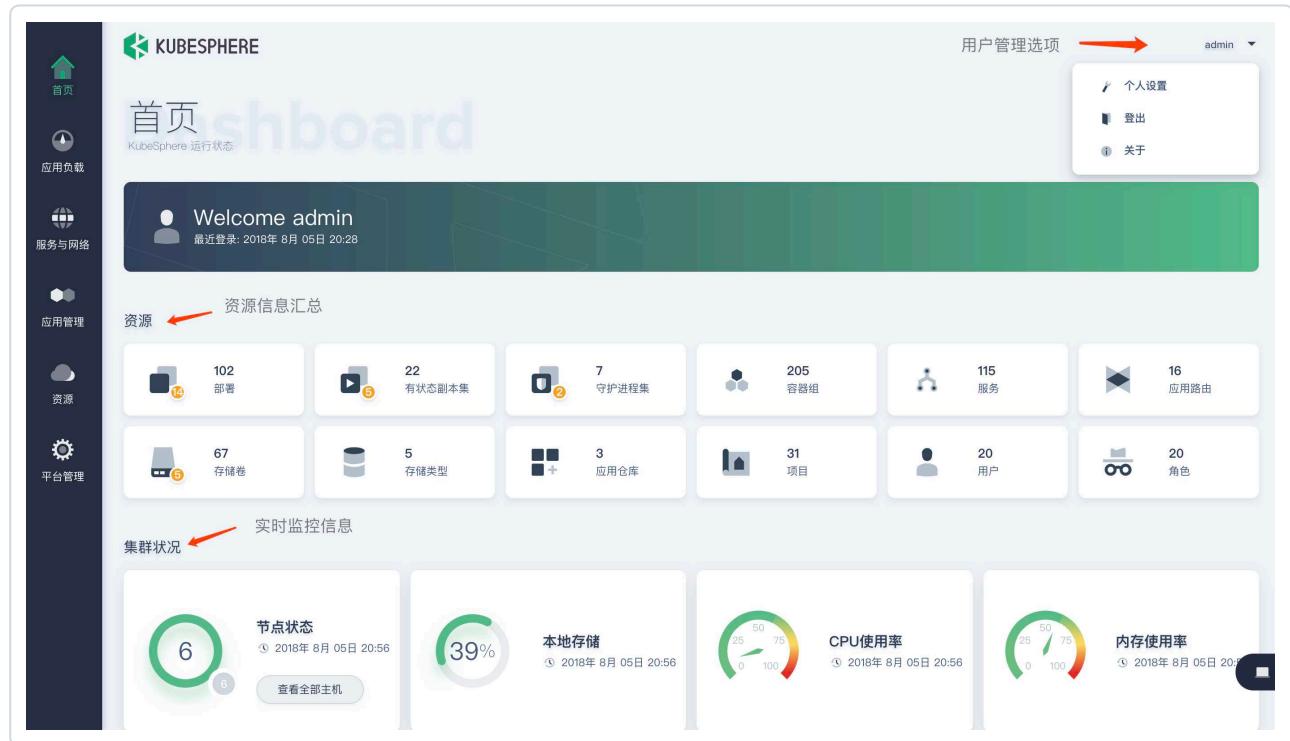
资源信息 事件 ↑

原因	状态	消息
EnsuringLoadBalancer 2018年 8月 06日 14:56:40	Normal	Ensuring load balancer
CreatingLoadBalancerFailed 2018年 8月 06日 14:56:44	异常中	Error creating load balancer (will retry): Failed to ensure load balancer for service lab/wordpress-service: QingCloud Error: Code (2100), Message (ResourceNotFound, resource [\'eip-0fe800q\']) not found)

首页

页面布局

首页汇总了当前集群下各种资源的使用情况，左侧为操作菜单栏，底部显示集群的实时监控状况，管理员可以看到所有项目的资源和集群状况汇总信息，而普通用户只能够看到该用户所属项目的资源信息。右上角为当前用户管理选项，用户可以在该选项中查看用户信息和版本信息、设置语言和登出系统。



如何使用 web kubectl

在 KubeSphere 控制台的右下角，可以看到一个命令行图标，点击图标，即可打开网页版的 kubectl 命令行控制台。

The screenshot shows the KubeSphere dashboard interface. On the left is a dark sidebar with navigation icons and labels: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 资源 (Resources), and 平台管理 (Platform Management). The main content area has a header with the KubeSphere logo and the text '首页' (Dashboard) and 'KubeSphere 运行状态' (KubeSphere Running Status). A welcome message 'Welcome admin' is displayed along with the last login time: '最近登录: 2018年 8月 05日 20:28'. Below this is a section titled '资源' (Resources) containing cards for Deployments (102), StatefulSets (22), Pods (7), Container Groups (205), Services (115), and Ingresses (16). Another section titled '集群状况' (Cluster Status) includes cards for Nodes (6), Local Storage (39%), CPU Usage Rate (20%), and Memory Usage Rate (20%). A button labeled '启动 kubectl' (Start kubectl) is visible next to the memory usage card.

在打开的网页版 kubectl 控制台上，用户可以按照之前使用 kubectl 的方式正常使用，同时仅限于操作此用户可以访问的资源。

部署

部署（Deployment）为 Pod 和 ReplicaSet 提供了一个声明式定义（declarative）方法来管理应用。典型的应用场景包括定义 Deployment 来创建 Pod 和 ReplicaSet、滚动升级和回滚应用、扩容和缩容以及暂停和继续 Deployment。

本节通过以下几个方面介绍如何管理部署：

- 创建部署
- 查看部署详情
- 编辑部署
- 删除部署

创建部署

首先登录 KubeSphere 控制台，访问左侧菜单栏，在 **应用负载** 菜单下，点击 **部署** 按钮，进入部署列表页面。选择不同的项目，就可以看到对应项目下的所有部署。如果是管理员登录，可以看到集群所有项目的部署情况，如果是普通用户，则只能查看授权项目下的所有部署。

The screenshot shows the KubeSphere deployment management interface. On the left, there's a dark sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area has a light blue header with the KubeSphere logo and the word 'Deployments'. Below the header, it says '共 102 个部署'. There are filters for '全部项目' (All Projects) and a search bar with placeholder '输入查询条件进行过滤' (Filter by input query). A green button on the right says '创建部署' (Create Deployment). The main table lists seven deployments:

名称	状态	应用	项目	更新时间
wordpress	更新中(0/1)	-	lab	2018年 8月 05日 12:11:24
wordpress-mysql	运行中(1/1)	-	lab	2018年 8月 04日 22:07:55
openpitrix-app-manager...	运行中(1/1)	-	openpitrix-system	2018年 8月 03日 12:01:48
multi-vol-single-con	运行中(1/1)	-	default	2018年 8月 03日 12:01:39
nginx-glusterfs	运行中(3/3)	-	project01	2018年 8月 03日 12:01:30
test-zq1	运行中(1/1)	-	zhangqun-project	2018年 8月 03日 12:01:18
openpitrix-job-manager...	运行中(1/1)	-	openpitrix-system	2018年 8月 03日 12:01:09

1. 点击右上角 **创建部署** 按钮，新建一个部署。



目前创建部署一共有三种方式，**页面创建**，**导入 yaml 文件** 创建，**代码模式** 创建。其中导入 yaml 文件方式会自动将 yaml 文件内容填充到页面上，用户根据需要可以在页面上调整后再行创建。代码模式可以方便习惯命令行操作的人员直接在页面上编辑 yaml 文件并创建部署。

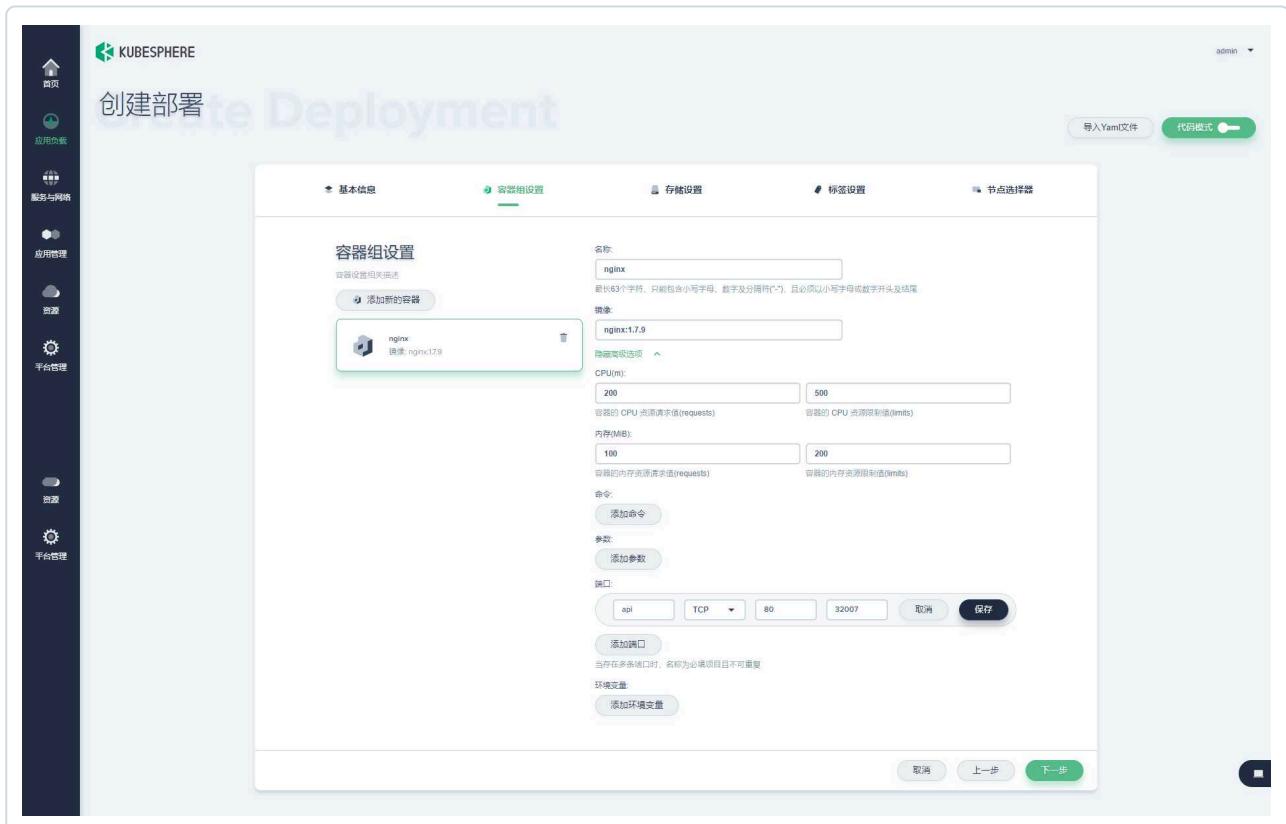


这里重点介绍**页面创建**的方式。需要输入部署的名称并选择创建部署的项目，确定部署的副本数量，用

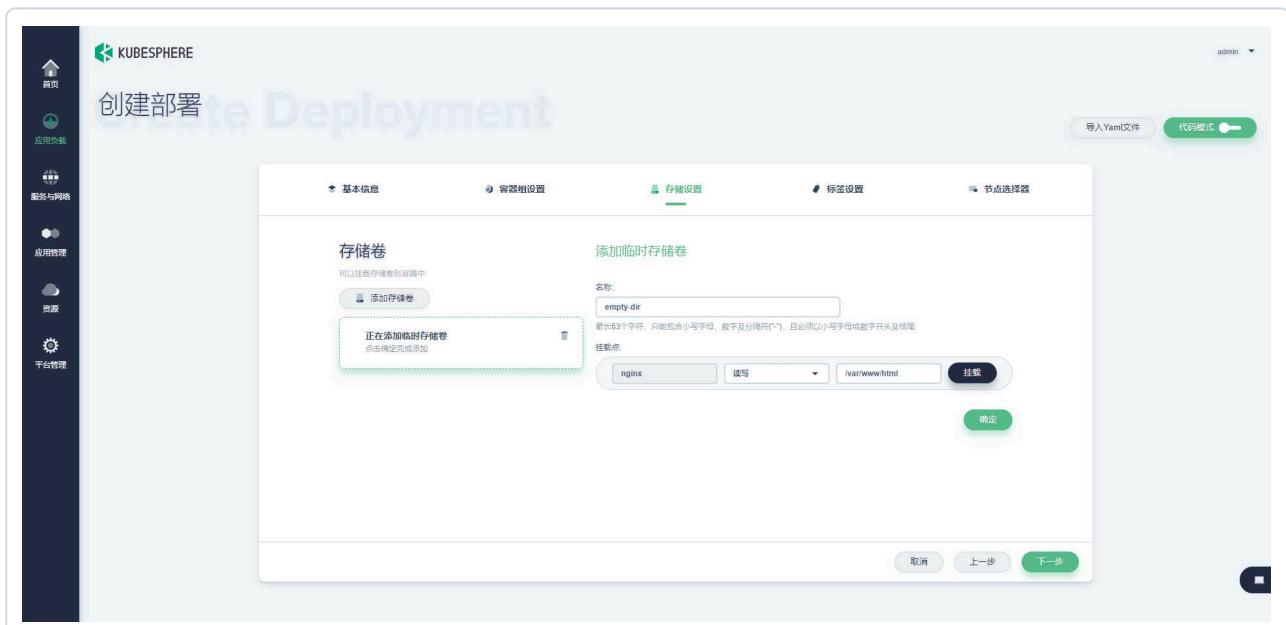
户还可以根据需求填写部署的描述信息。



- 在 **容器组设置** 页面, 用户可以根据需求对容器组进行配置。容器组可以包含一个或者多个容器。要求输入容器的名称和对应的镜像名。如果用户有更进一步的需求, 可以点击**高级选项**。高级选项中可以对 CPU 和内存的资源使用进行限定。其中 **requests** 是集群保证分配给容器的资源, **limits** 是容器可以使用的资源的上限。超过这个限定值, 容器会被 kill。用户可以通过**命令和参数** 选项自定义容器的启动命令和启动参数。**端口** 用于指定容器需要暴露的端口, 端口协议可以选择 TCP 和 UDP, 用户还可以指定端口与主机端口进行绑定。**环境变量** 可以指定容器内部使用的环境变量。上述配置信息填写完成以后, 点击下一步。

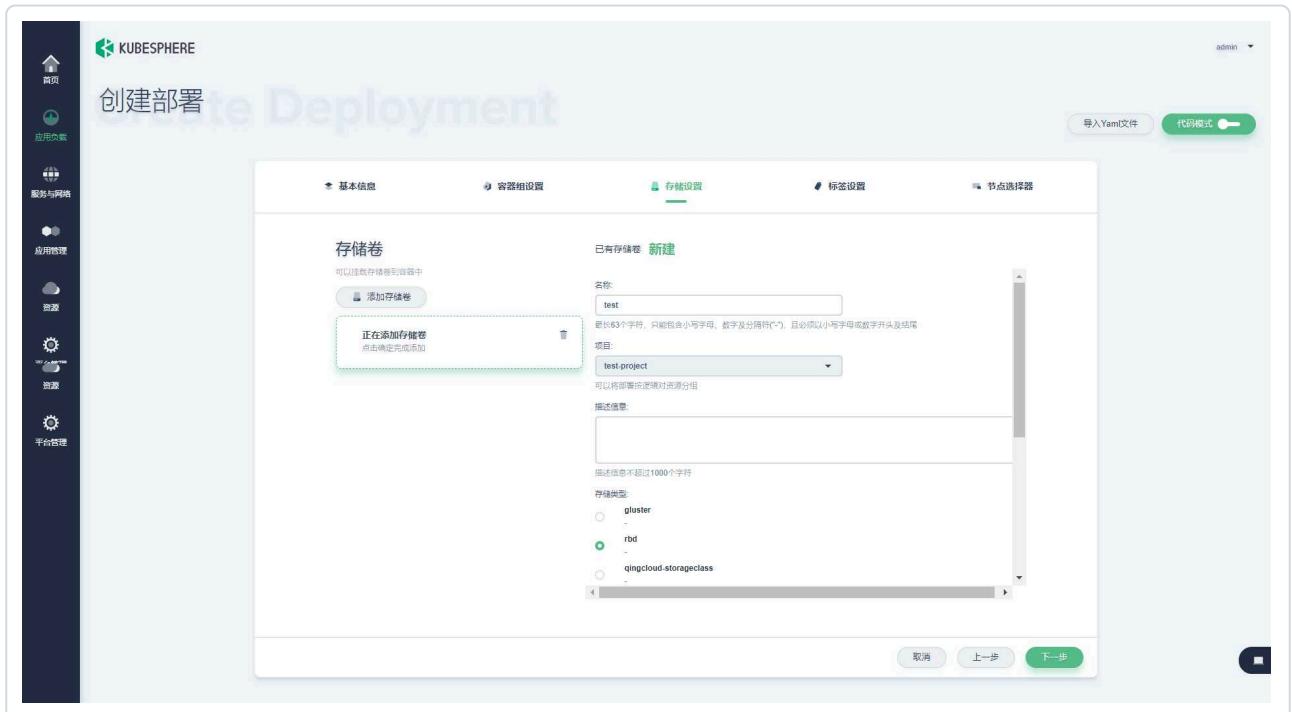


3. 在存储卷页面可以添加 **存储卷** 和 **临时存储卷**。存储卷可用于持久化用户数据；保留在临时存储卷中的数据，在容器组被删除以后，也将被同步删除。选择使用临时存储卷方案时要求用户输入存储卷的名称和相应的挂载点，然后点击 **挂载** 和 **确定**。



当选择使用存储卷时，可以选择 **已有存储卷** 或者根据需求 **新建存储卷**。当选择新建存储卷时，用户需

要填写存储卷的名称，选择存储类型，指定容量和访问模式。最后输入挂载点，并且点击挂载和确定，点击下一步。



4. 标签设置页用于指定部署对应的一组或者多组标签。



5. 节点选择器页面，用户可以通过指定一组或者多组键值对来指定期望运行容器组的主机。当不指定时，容器组将有可能调度到集群内满足调度条件的任意节点。最后点击创建，集群就会按照用户的配置创建对应的部署。



查看部署详情

- 在部署列表页，点击某个部署的名称，就可以进入到部署详情页，可以查看部署的详细信息以及部署对应的各种资源。

名称	状态	主机	宿宿 IP	监控
nginx-65cd787f98-p9ncz	运行中	i7l0rapp 192.168.0.40	10.244.1.176	CPU 0.00m 内存 3.80MiB

- 同时在本页可以通过左下角的容器组数量栏调整部署的副本数量，左上角的更多选项编辑配置文件，暂停/开启/删除部署。

The screenshot shows the KubeSphere interface for managing a deployment named 'nginx'. On the left sidebar, there are several navigation items: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area displays the deployment details for 'nginx'. At the top, there are tabs for 编辑 (Edit) and 详情 (Details). Below these are sections for 标签 (Labels), 详细 (Details), and 容器组 (Container Groups). The '容器组' section lists a single container named 'nginx-65cd787f98-p9ncz' with the following details:

名称	状态	主机	容器组 IP	监控
nginx-65cd787f98-p9ncz	运行中	i-7flreqs	192.168.0.40	CPU 0.00m 内存 3.80MiB

Below the container list is a storage volume section labeled '存储卷' (Storage Volume) with a capacity of 10Gi (rbd, RWO).

At the bottom of the deployment page, there is a summary section titled '容器组数量' (Number of Container Groups) showing 1 group in progress.

- 当点击容器组列表的容器时，页面会跳转到容器组的详情页。

This screenshot shows the detailed view of a specific container within a deployment. The URL in the browser is /nginx/nginx-65cd787f98-p9ncz. The interface is similar to the previous one, with a sidebar and a main content area. The main content area includes sections for 标签 (Labels), 详细 (Details), 资源信息 (Resource Information), and 事件 (Events). The '资源信息' section provides resource usage details:

CPU	0.00m	内存	3.80MiB
总计 500.0 m	使用量	总计 200.0 MiB	使用量

The '容器' (Container) section lists the single container 'nginx' with the following details:

名称	状态	镜像	端口	重启次数	监控
nginx	运行中	nginx:1.7.9	-	0	CPU 0.00m 内存 2.60MiB

Below the container list is a storage volume section labeled '存储卷' (Storage Volume) with a capacity of 10Gi (rbd).

- 点击容器组详情页的容器列表中的某一容器时，会跳转到容器的详情页，可以看到容器的配置信息，对应资源以及日志。在本页，用户还可以通过左上角的 终端 按钮进入容器内部。

The screenshot shows the KubeSphere application dashboard. On the left sidebar, there are several icons: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area displays deployment details for 'nginx'. At the top, it says '部署 / nginx / nginx-65cd787f98-p9ncz / nginx'. Below this, there's a summary card with CPU usage (0.00m) and memory usage (2.60MiB). The '资源信息' tab is selected, showing resource requests (cpu: 200m / memory: 100Mi) and limits (cpu: 500m / memory: 200Mi). It also shows environment variables, storage卷 (volume), and logs. A terminal icon is present at the bottom right.

编辑部署

- 在部署的列表页和详情页中都可以编辑部署对应的 yaml 文件，编辑是在代码模式下完成的，编辑完成以后，点击右下方的更新按钮，部署就会按照最新的配置进行更新。

The screenshot shows a code editor interface for editing a deployment YAML file. The title bar says '代码模式' and '正在编辑 test-project 的资源配置' (Editing test-project's resource configuration). The editor displays the following YAML code:

```
47       - name: volume-cfetum
48         mountPath: /var/www/html
49         terminationMessagePath: /dev/termination-log
50     terminationMessagePolicy: File
51     imagePullPolicy: IfNotPresent
52   restartPolicy: Always
53   terminationGracePeriodSeconds: 30
54   dnsPolicy: ClusterFirst
55   securityContext: {}
56   schedulerName: default-scheduler
57   strategy:
58     type: RollingUpdate
59     rollingUpdate:
60       maxUnavailable: 1
61       maxSurge: 25%
62     revisionHistoryLimit: 10
63     progressDeadlineSeconds: 600
64   status:
65     observedGeneration: 2
66     replicas: 1
67     updatedReplicas: 1
68     unavailableReplicas: 1
69   conditions:
70     - type: Available
71       status: 'True'
72       lastUpdateTime: '2018-07-16T08:01:51Z'
73       lastTransitionTime: '2018-07-16T08:01:51Z'
74       reason: MinimumReplicasAvailable
75       message: Deployment has minimum availability.
76     - type: Progressing
77       status: 'False'
78       lastUpdateTime: '2018-07-16T08:15:34Z'
79       lastTransitionTime: '2018-07-16T08:15:34Z'
80       reason: ProgressDeadlineExceeded
81       message: ReplicaSet "nginx-65cd787f98" has timed out progressing.
82
```

At the top right, there are buttons for 'Yaml 格式' (YAML format), '保存为Yaml文件' (Save as YAML file), and '保存为JSON文件' (Save as JSON file). At the bottom right, there are '取消' (Cancel) and '更新' (Update) buttons.

- 如果只是修改部署的描述信息，可以使用部署详情页左上角的 编辑 进行修改。

The screenshot shows the KubeSphere interface. On the left sidebar, there are icons for Home, Application Catalog, Services & Networks, Application Management, Resources, and Platform Management. The main area displays a deployment named 'nginx' under the 'test-project' project. The deployment has 1 desired, 1 total, and 1 updated pod, all of which are unavailable. The status is 'Running (1/1)'. The application is 'app = nginx'. The deployment type is '滚动更新' (Rolling Update) with 'maxUnavailable=1, maxSurge=25%'. It was created on July 16, 2018, at 16:01. A modal window titled '编辑信息' (Edit Information) is open, showing the name 'nginx', project 'test-project', and a description field. Buttons for '取消' (Cancel) and '确定' (Confirm) are at the bottom.

删除部署

- 在列表页最右侧的更多选项和部署详情页的左上方的更多选项中都提供了部署的删除功能。

The screenshot shows the KubeSphere deployment list page. The sidebar includes Home, Application Catalog, Services & Networks, Application Management, Resources, and Platform Management. The main area lists two deployments: 'nginx' and 'test-deploy'. 'nginx' is in the 'Running (1/1)' state, while 'test-deploy' is in the 'Updating (0/1)' state. Both are associated with the 'test-project' project and were created on July 16, 2018. A context menu is open for the 'nginx' deployment, showing options: '编辑YAML文件' (Edit YAML File), '容器组规模' (Container Group Scale), '暂停' (Pause), and '删除' (Delete). The 'Delete' option is highlighted.

The screenshot shows the KubeSphere web interface for managing applications. On the left, a sidebar lists various management categories: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), and 平台管理 (Platform Management). The main content area is titled 'nginx' under the '部署' (Deployment) section. It displays the '资源信息' (Resource Information) tab, which includes a table for the '容器组' (Pod) named 'nginx-65cd787f98-p9ncz'. The table shows the following columns: 名称 (Name), 状态 (Status), 主机 (Host), 容器组 IP (Pod IP), CPU (CPU), and 内存 (Memory). The pod status is '运行中' (Running), assigned to host 'i-7llapp-092168040', with 0.00m CPU and 3.80MiB memory. Below the table, there's a '存储卷' (Storage Volume) section showing a volume named 'test' with 10Gi capacity and labels 'app: nginx', 'pod: p9ncz', and 'rwcl'. On the left side of the main content, there's a detailed '详情' (Details) panel for the 'nginx' deployment, showing fields like 副本数量 (Replica Count), 状态 (Status), 选择器 (Selector), 卷插类型 (Volume Mount Type), 卷插更新 (Volume Mount Update), and 创建时间 (Creation Time). A large green circle at the bottom indicates '1' container group(s) are running.

有状态副本集

有状态副本集 (StatefulSet)，是为了解决有状态服务的问题（对应 Deployments 和 ReplicaSets 是为无状态服务而设计），可以保证资源创建和伸缩时候的顺序。应用场景包括：

- 稳定的持久化存储，即 Pod 重新调度后还是能访问到相同的持久化数据，基于 PVC 来实现。
- 稳定的网络标志，即 Pod 重新调度后其 PodName 和 HostName 不变，基于 Headless Service（即没有 Cluster IP 的 Service）来实现。
- 有序部署、有序扩展，即 Pod 是有序的，在部署或者扩展的时候要依据定义的顺序依次进行（即从 0 到 N-1，在下一个 Pod 运行之前所有之前的 Pod 必须都是 Running 和 Ready 状态），基于 init containers 来实现。
- 有序收缩、有序删除（即从 N-1 到 0）。

本节通过以下几个方面介绍如何管理有状态副本集：

- 创建有状态副本集
- 查看有状态副本集详情
- 编辑有状态副本集
- 删除有状态副本集

创建有状态副本集

首先登录 KubeSphere 控制台，访问左侧菜单栏，在 **应用负载** 菜单下，点击 **有状态副本集** 按钮进入列表页。选择不同的项目，就可以看到对应项目下面所有有状态副本集。如果是管理员登录，可以看到集群所有项目下的有状态副本集，如果是普通用户，则只能查看授权项目下的所有有状态副本集。

The screenshot shows the KubeSphere web interface. On the left is a dark sidebar with navigation icons for Home, Application Management, Services & Networks, Resource Management, and Platform Management. The main content area has a header '有状态副本集' (StatefulSets) with a sub-count '共 12 个有状态副本集'. Below this is a table listing 12 StatefulSets, each with a checkbox, name, status, application, project, and update time. A search bar at the top right says '输入查询条件进行过滤'. A green button at the top right says '创建有状态副本集'. The table rows are as follows:

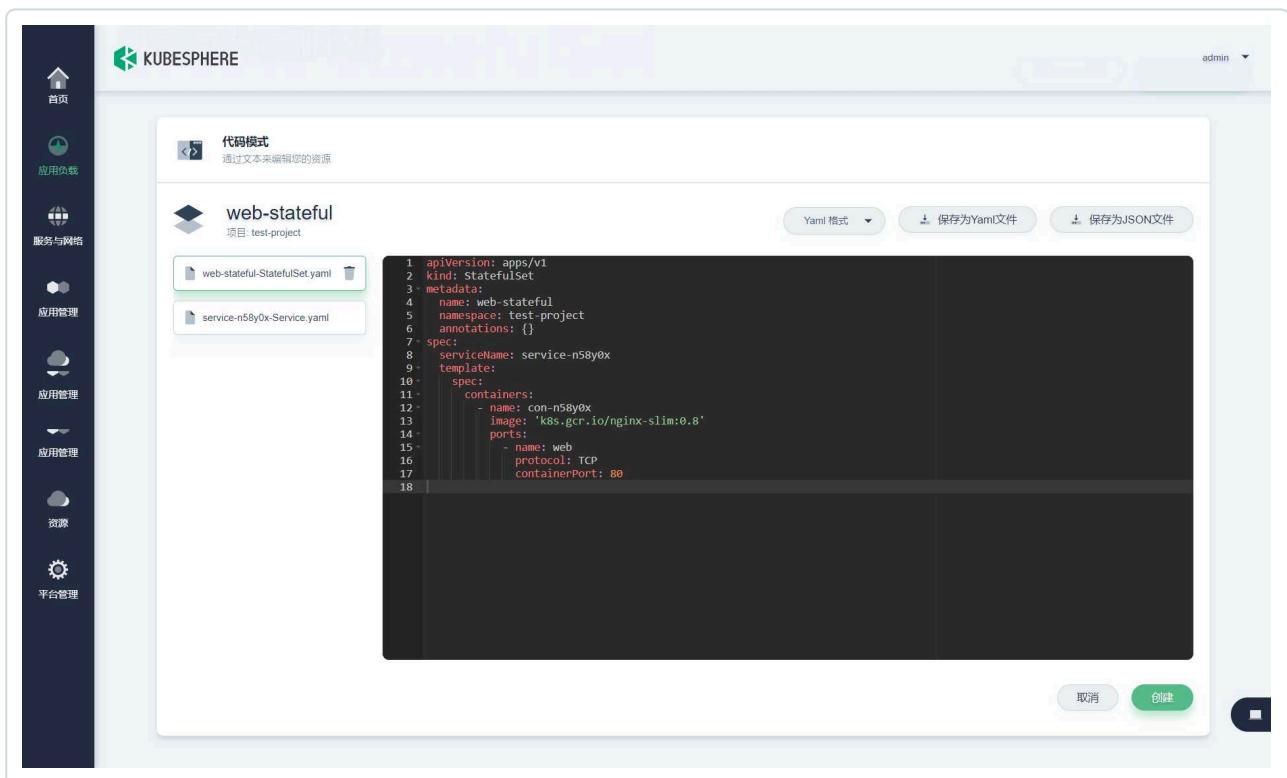
名称	状态	应用	项目	更新时间
mysql-mariadb-slave	运行中(1/1)	mysql/mariadb-4.2.7	firstex	2018年7月16日 20:29:34
mysql-mariadb-master	运行中(1/1)	mysql/mariadb-4.2.7	firstex	2018年7月16日 20:29:34
stateful	更新中(0/5)	-	test-project	2018年7月16日 20:29:34
consul	运行中(3/3)	consul/consul-3.2.0	xz	2018年7月16日 20:29:34
elevated-badger-rabbitmq-ha	更新中(0/1)	elevated-badger/rabbitmq-ha-1.5.2	demo-fu	2018年7月16日 20:29:34
xz-consul	运行中(3/3)	xz-consul/consul-3.2.0	hongming-project	2018年7月16日 20:29:34
consul-xz	运行中(3/3)	consul-xz/consul-3.2.0	xz2	2018年7月16日 20:29:34
ddddd-memcached	运行中(3/3)	ddddd-memcached-2.1.1	hongming-project	2018年7月16日 20:29:34
memcached-001031023	运行中(3/3)	memcached-001031023/memcached-2.1.1	hongming-project	2018年7月16日 20:29:34
nginx-rbd-test	运行中(2/2)	-	default	2018年7月16日 20:29:34

At the bottom right of the table area, there is a page number '1 / 2'.

1. 点击右上角 创建有状态副本集 按钮，新建一个有状态副本集。

The screenshot shows the 'Create StatefulSet' wizard. The sidebar on the left is identical to the previous screenshot. The main area has a title '创建有状态副本集' (Create StatefulSet). It features a tab navigation bar with '基本信息' (Basic Information) selected, followed by '容器组设置' (Container Group Settings), '存储卷模板' (Storage Volume Template), '服务设置' (Service Settings), '标签设置' (Label Settings), and '节点选择器' (Node Selector).
基本信息 (Basic Information):
- **名称 (Name):** A text input field with placeholder '请输入名称' (Enter name). A note below says '最长253个字符，只能包含小写字母、数字及分隔符“-”，且必须以小写字母或数字开头及结尾' (Length up to 253 characters, can only contain lowercase letters, numbers, and hyphens, must start and end with lowercase letters or numbers).
- **应用 (Application):** A dropdown menu set to 'test-project'. A note below says '可以将有状态副本集按逻辑对资源分组' (Can group StatefulSets logically by resources).
- **副本数量 (Replica Count):** An input field with value '1', with minus and plus buttons for adjustment. A note below says '有状态副本集将按逻辑分组创建，由已维护的集群中启停的所需数量' (StatefulSets will be created logically grouped, based on the number of pods managed by the cluster).
- **描述信息 (Description):** A text area for notes.
At the bottom right are '取消' (Cancel) and '下一步' (Next Step) buttons.

目前创建有状态副本集一共有三种方式，**页面创建**，**导入 yaml 文件创建**，**代码模式** 创建。其中导入 yaml 文件方式会自动将 yaml 文件内容填充到页面上，用户根据需要可以在页面上调整后再行创建；代码模式可以方便习惯命令行操作的人员直接在页面上编辑 yaml 文件并创建有状态副本集。

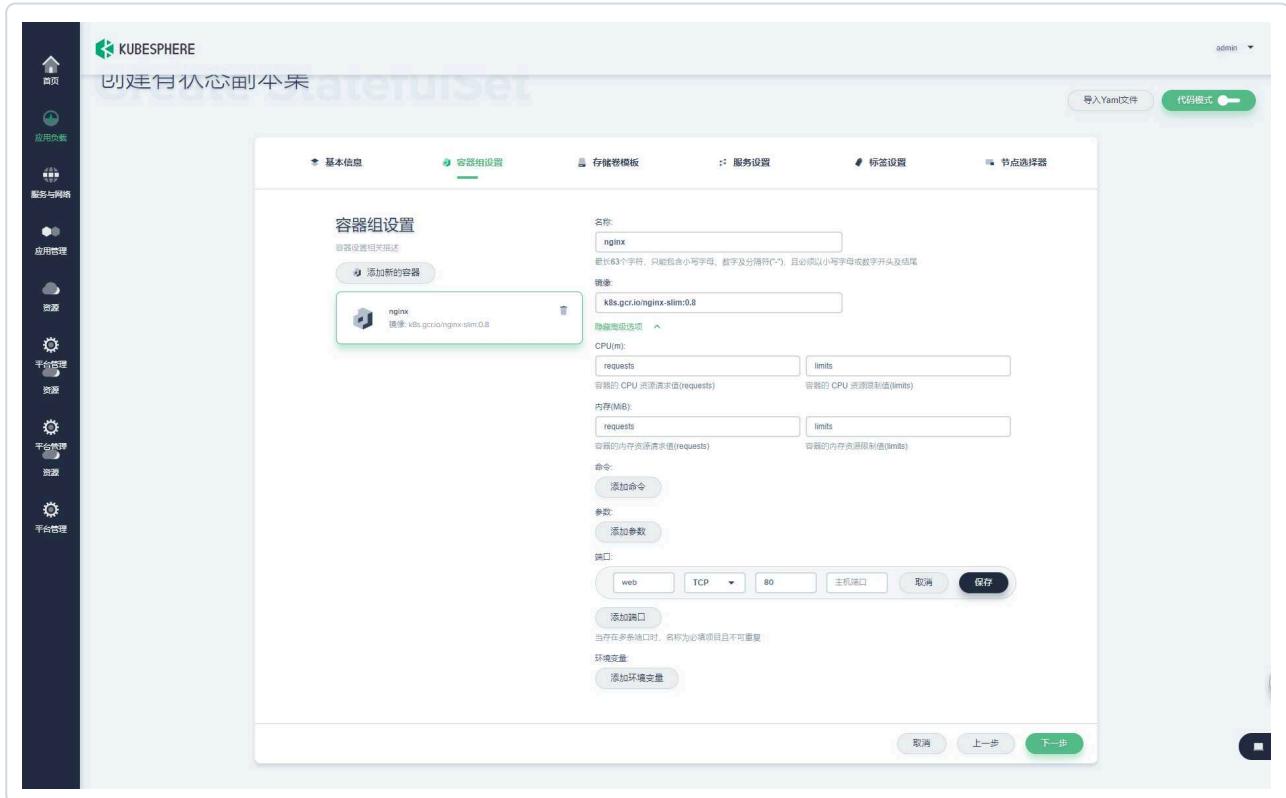


这里重点介绍页面创建的方式。创建时需要输入有状态副本集的名称并选择项目，确定副本数量，用户还可以根据需求填写有状态副本集的描述信息。

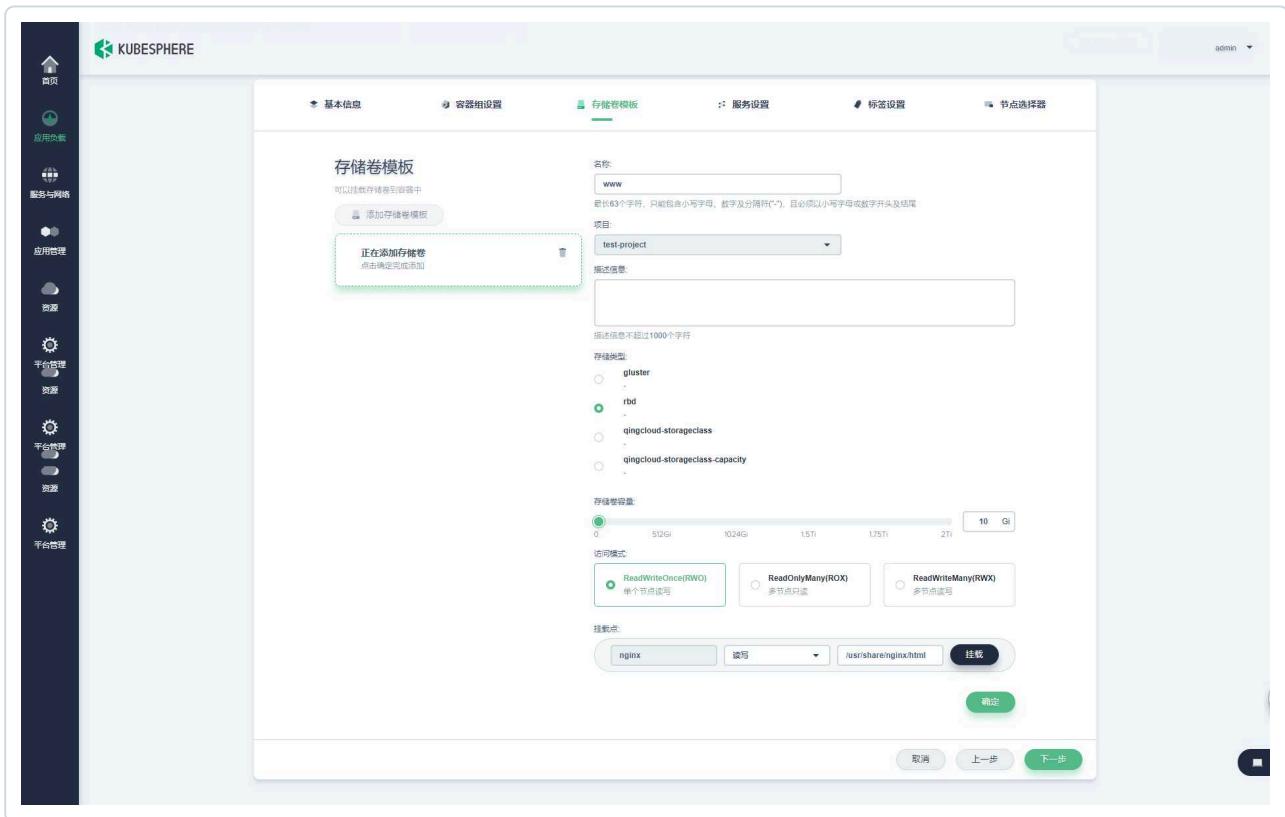


2. 在 **容器组设置页面** 用户可以根据需求对容器组进行配置。容器组可以包含一个或者多个容器，要求输入容器的名称和对应的镜像名。如果用户有更进一步的需求，可以点击 **高级选项**。高级选项中可以对 CPU 和内存的资源使用进行限定，其中 **requests** 是集群保证分配给容器的资源，**limits** 是容

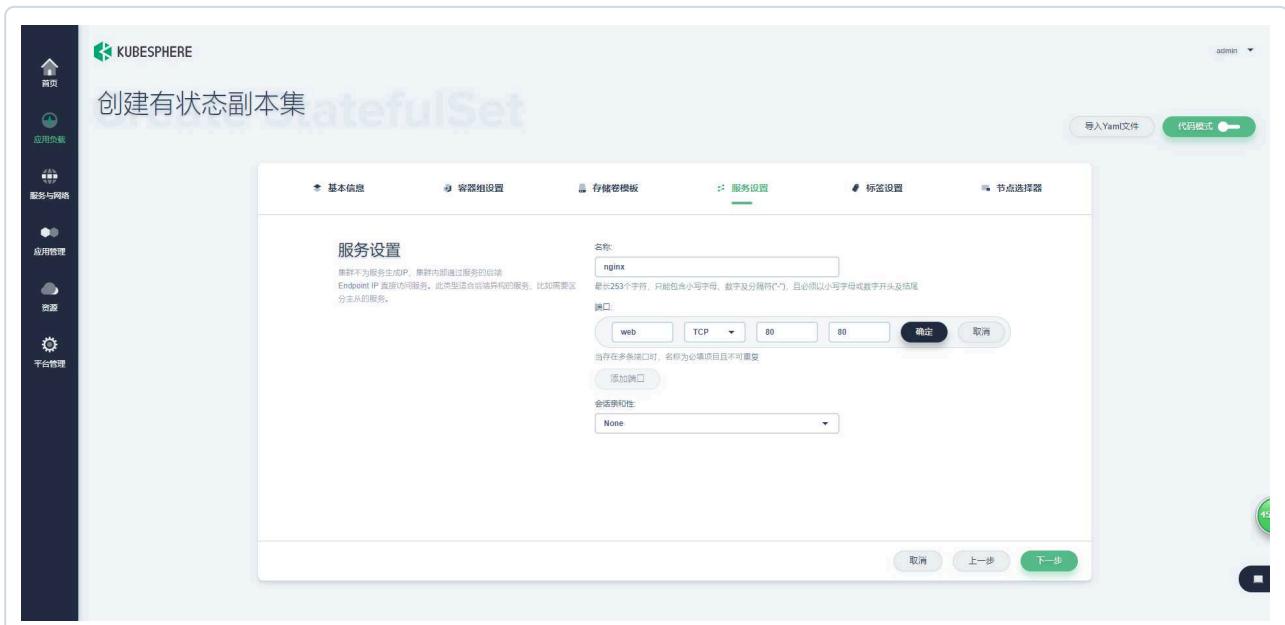
器可以使用的资源的上限。超过这个限定值，容器会被 kill。用户可以通过 **命令** 和 **参数** 选项自定义容器的启动命令和启动参数。**端口** 用于指定容器需要暴露的端口，端口协议可以选择 TCP 和 UDP，用户还可以指定端口与主机端口进行绑定。**环境变量** 可以指定容器内部使用的环境变量。上述配置信息填写完成以后，点击下一步。



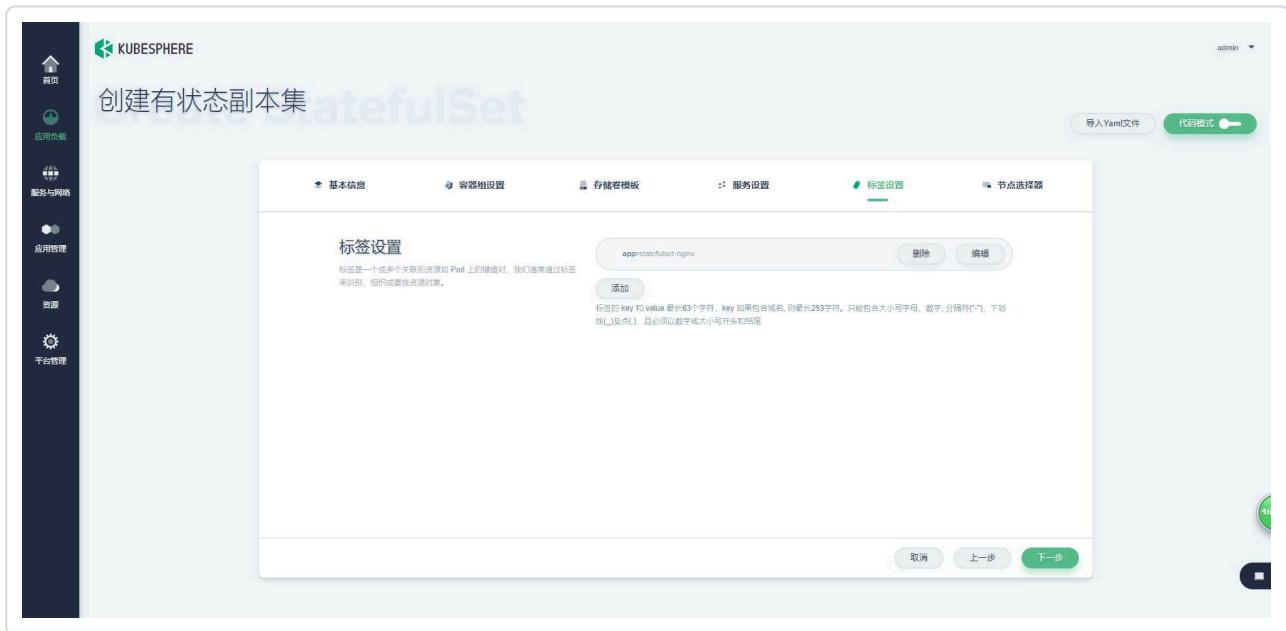
- 在存储卷页面可以添加 **存储卷模板**，根据用户的配置为有状态副本集的每个容器组生成存储卷。点击 **添加存储卷模板**，填写存储卷的名称，选择存储类型，指定容量和访问模式。最后确定挂载点，并且点击挂载和确定。然后点击下一步。



4. **服务设置** 页面用于给有状态副本集创建服务，需要填写服务名称，服务端口和目标端口，目标端口对应容器组的暴露端口。用户可根据需求设置 **会话亲和性**，选择完成以后点击下一步。



5. 标签设置页用于指定有状态副本集对应的一组或者多组标签。



6. 节点选择器页面，用户可以通过指定一组或者多组键值对来指定期望运行容器组的主机。当不指定时，将会在集群内的任意满足调度条件的节点上启动符合副本数量的容器组。点击右下角的创建后，集群就会按照用户的配置创建对应的有状态副本集。



查看有状态副本集详情

- 在有状态副本集列表页，点击某个有状态副本集的名称，就可以进入到有状态副本集详情页，可以查看有状态副本集的详细信息以及有状态副本集对应的各种资源。

KUBESPHERE

资源信息 事件

服务

名称: nginx 端口: 80:80/TCP

名称	状态	主机	容器组 IP	监控
web-0	运行中	i79lraap 192.168.0.40	10.244.1.180	CPU 0.00m 内存 4.80MB
web-1	运行中	i3q5u2pa 192.168.0.41	10.244.3.35	CPU 0.00m 内存 4.20MB
web-2	运行中	i3do5s6f 192.168.0.42	10.244.2.225	CPU 0.00m 内存 4.50MB

容器组数量: 3个容器组正在运行

- 同时在本页可以通过左下角的容器组数量栏调整有状态副本集的副本数量，左上角的更多选项编辑配置文件，暂停/开启/删除有状态副本集

KUBESPHERE

资源信息 事件

服务

名称: nginx 端口: 80:80/TCP

名称	状态	主机	容器组 IP	监控
web-0	运行中	i79lraap 192.168.0.40	10.244.1.180	CPU 0.00m 内存 4.80MB
web-1	运行中	i3q5u2pa 192.168.0.41	10.244.3.35	CPU 0.00m 内存 4.20MB
web-2	运行中	i3do5s6f 192.168.0.42	10.244.2.225	CPU 0.00m 内存 4.50MB

容器组数量: 3个容器组正在运行

- 当点击容器组列表的容器时，页面会跳转到容器组的详情页。

The screenshot shows the KubeSphere UI for a pod named 'web-0'. On the left, there's a sidebar with navigation links: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area has a header 'KUBESPHERE' and a breadcrumb '有状态副本集 / web / web-0'. Below this, there's a section for the pod 'web-0' with a '容器组' (Container Group) icon. A '标签' (Labels) section lists 'app:statefulset-nginx', 'controller-revision-hash:web-869d97b9c9', and 'statefulset.kubernetes.io/pod-name:web-0'. A '详情' (Details) section provides project information ('test-project'), state ('运行中' - Running), host name ('lxdd056f'), host IP ('192.168.0.42'), and restart count ('0'). To the right, there are tabs for '资源信息' (Resource Information) and '事件' (Events). Under '资源信息', there are sections for 'CPU' (0.00m usage, unlimited limit) and '内存' (4.40MB usage, 4.40MB limit). Below these are tables for '容器' (Containers) and '存储卷' (Storage Volumes). The container table shows one entry: 'nginx' (running, image 'k8s.gcr.io/nginx-slim:0.8', 0 restarts, CPU 0.00m, memory 2.90MB). The storage volume table shows one entry: 'www-web-0' (rbd, 10Gi capacity). A green button at the bottom right is partially visible.

- 点击容器组详情页的容器列表中的某一容器时，会跳转到容器的详情页，可以看到容器的配置信息，对应资源以及日志。在本页，用户还可以通过左上角的 **终端** 按钮进入容器内部。

This screenshot shows the KubeSphere UI for a specific container named 'nginx' within a pod. The sidebar and top navigation are identical to the previous screenshot. The main content area shows a container group 'nginx' with a '容器' (Container) icon. A '终端' (Terminal) button is visible. The '详情' (Details) section includes fields for '项目' (test-project), '状态' (运行中 - Running), '镜像' (image 'k8s.gcr.io/nginx-slim:0.8'), '端口' (ports), '命令' (commands), '环境变量' (environment variables), '资源请求' (resource requests), '资源限制' (resource limits), '镜像拉取策略' (image pull policy 'IfNotPresent'), '特权模式' (privileged mode), and '重启次数' (restart count '0'). To the right, there are tabs for '资源信息' (Resource Information) and '容器日志' (Container Logs). The '资源信息' tab displays 'CPU' (0.00m usage, unlimited limit) and '内存' (3.40MB usage, 3.40MB limit). Below these are tables for '容器' (Containers) and '存储卷' (Storage Volumes). The container table is empty. The storage volume table shows one entry: 'www-web-0' (rbd, 10Gi capacity). A green button at the bottom right is partially visible.

编辑有状态副本集

- 在有状态副本集的列表页和详情页的更多选项中都可以编辑对应的配置文件，编辑是在代码模式下完成的，编辑完成以后，点击右下方的更新按钮，就会按照最新的配置进行更新。

The screenshot shows the KubeSphere UI with the 'YAML' tab selected. The code editor displays the YAML configuration for a StatefulSet named 'web'. The configuration includes metadata, spec (with selector, replicas, volumeClaimTemplate, and service), and status (with podManangementPolicy, updateStrategy, and rollingUpdate). The status section shows 3 ready replicas and the latest revision is 'web-869d97b9b9'. At the bottom right are '取消' (Cancel) and '更新' (Update) buttons.

```
36      restartPolicy: Always
37      terminationGracePeriodSeconds: 30
38      dnsPolicy: ClusterFirst
39      securityContext: {}
40      schedulerName: default-scheduler
41    volumeClaimTemplate:
42      - metadata:
43          name: web
44          namespace: test-project
45          creationTimestamp: null
46        spec:
47          accessModes:
48            - ReadWriteOnce
49          resources:
50            requests:
51              storage: 10Gi
52            storageClassName: rbd
53            volumeMode: Filesystem
54          status:
55            phase: Pending
56        serviceName: nginx
57      podManagementPolicy: OrderedReady
58      updateStrategy:
59        type: RollingUpdate
60        rollingUpdate:
61          partition: 0
62        revisionHistoryLimit: 10
63      status:
64        observedGeneration: 1
65      replicas: 3
66      readyReplicas: 3
67      currentReplicas: 3
68      currentRevision: web-869d97b9b9
69      updateRevision: web-869d97b9b9
70      collisionCount: 0
71
```

- 在有状态副本集的详情页左上方的编辑可以对描述信息进行修改。

The screenshot shows the KubeSphere UI with the 'Resource Information' tab selected. On the left, the 'web' StatefulSet is listed under the 'test-project' project. A modal dialog titled '编辑信息' (Edit Information) is open, showing the 'Description' field with the value 'web'. Below the modal, the 'web' container group is shown with three replicas (web-0, web-1, web-2) and their resource usage metrics (CPU 0.00m, 内存 4.80MiB).

删除有状态副本集

- 在列表页最右侧的更多选项和详情页的左上方的更多选项中都提供了有状态副本集的删除功能。

KUBESPHERE

有状态副本集

共 13 个有状态副本集

名称	状态	应用	项目	更新时间
web	运行中(3/3)	-	test-project	2018年7月 16日 21:01:41
mysql-mariadb-slave	运行中(1/1)	mysql/mariadb-4.2.7	firstex	2018年7月 16日 21:01:41
mysql-mariadb-master	运行中(1/1)	mysql/mariadb-4.2.7	firstex	2018年7月 16日 21:01:41
stateful	更新中(0/3)	-	test-project	2018年7月 16日 21:01:41
consul	运行中(3/3)	consul/consul-3.2.0	xz	2018年7月 16日 21:01:41
elevated-badger-rabbitmq-ha	更新中(0/1)	elevated-badger/rabbitmq-ha-1.5.2	demo-fu	2018年7月 16日 21:01:41
consul	运行中(3/3)	consul/consul-3.2.0	xz	2018年7月 16日 21:01:41
elevated-badger-rabbitmq-ha	更新中(0/1)	elevated-badger/rabbitmq-ha-1.5.2	demo-fu	2018年7月 16日 21:01:41
xz-consul	运行中(3/3)	xz-consul/consul-3.2.0	hongming-project	2018年7月 16日 21:01:41
consul	运行中(3/3)	consul/consul-3.2.0	xz	2018年7月 16日 21:01:41
elevated-badger-rabbitmq-ha	更新中(0/1)	elevated-badger/rabbitmq-ha-1.5.2	demo-fu	2018年7月 16日 21:01:41
xz-consul	运行中(3/3)	xz-consul/consul-3.2.0	hongming-project	2018年7月 16日 21:01:41
consul-xz	运行中(3/3)	consul-xz/consul-3.2.0	xz2	2018年7月 16日 21:01:41
dddddmemcached	运行中(3/3)	dddddmemcached-2.1.1	hongming-project	2018年7月 16日 21:01:41
memcached-001031023	运行中(3/3)	memcached-001031023/memcached-2.1.1	hongming-project	2018年7月 16日 21:01:41

创建有状态副本集

KUBESPHERE

有状态副本集 / web

资源信息 事件

服务

名称	端口
nginx	80:80/TCP

容器组

名称	状态	主机	容器组 IP	监控
web-0	运行中	i-7lirapq 192.168.0.40	10.244.1.190	CPU 0.00m 内存 4.80MiB
web-1	运行中	i-3g9u2po 192.168.0.41	10.244.3.35	CPU 0.00m 内存 4.20MiB
web-2	运行中	i-wd905v6f 192.168.0.42	10.244.2.225	CPU 0.00m 内存 4.50MiB

容器组数量 3个容器组正在运行

守护进程集

守护进程集 (DaemonSet)，保证在每个 Node 上都运行一个容器副本，常用来部署一些集群的日志、监控或者其他系统管理应用。典型的应用场景包括：

- 日志收集，比如 Fluentd, Logstash 等。
- 系统监控，比如 Prometheus Node Exporter, collectd, New Relic agent, Ganglia gmond 等。
- 系统程序，比如 kube-proxy, kube-dns, Gluster, Ceph 等。

本节通过以下几个方面介绍如何管理守护进程集：

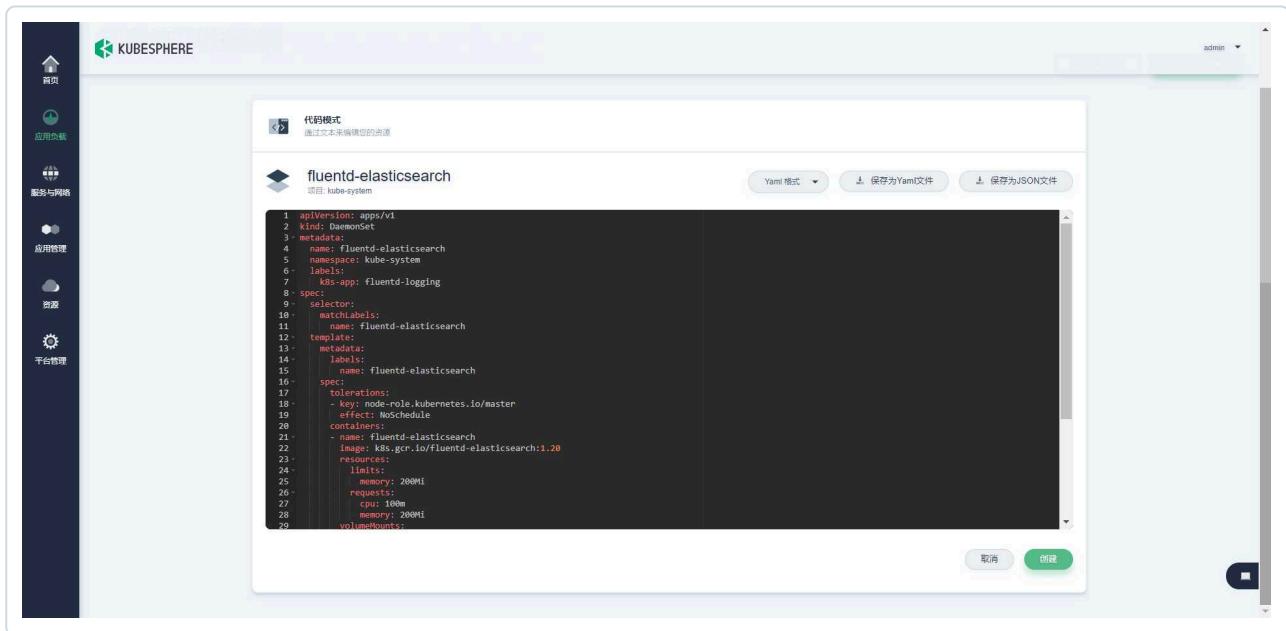
- 创建守护进程集
- 查看守护进程集
- 编辑守护进程集
- 删除守护进程集

创建守护进程集

首先登录 KubeSphere 控制台，访问左侧菜单栏，在 **应用负载** 菜单下，点击 **守护进程集** 按钮进入列表页。选择不同的项目，就可以看到对应项目下面的所有守护进程集。如果是管理员登录，可以看到集群所有项目的守护进程集，如果是普通用户，则只能查看授权项目下的所有守护进程集。

1. 点击右上角 **创建守护进程集** 按钮，新建一个守护进程集。

目前创建守护进程集一共有三种方式，**页面创建**，**导入 yaml 文件** 创建，**代码模式** 创建。其中**导入 yaml 文件** 方式会自动将 yaml 文件内容填充到页面上，用户根据需要可以在页面上调整后再行创建。**代码模式**可以方便习惯命令行操作的人员直接在页面上编辑yaml文件并创建守护进程集。

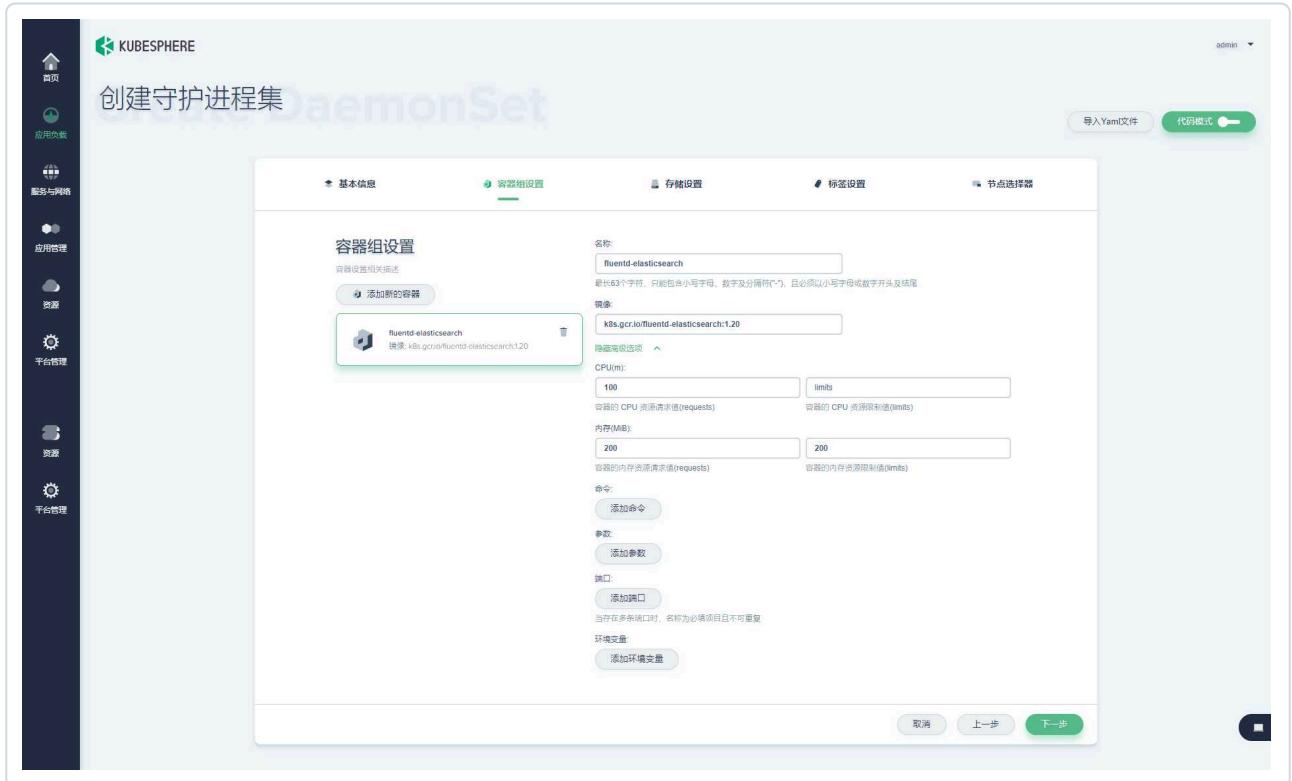


这里重点介绍 **页面创建** 的方式。创建时需要输入守护进程集的名称并选择项目，用户还可以根据需求填写守护进程集的描述信息。



- 在 **容器组设置页面** 用户可以根据需求对容器组进行配置。容器组可以包含一个或者多个容器，要求输入容器的名称和对应的镜像名。如果用户有更进一步的需求，可以点击 **高级选项**。高级选项中可以对 CPU 和内存的资源使用进行限定。其中 **requests** 是集群保证分配给容器的资源，**limits** 是容器可以使用的资源的上限。超过这个限定值，容器会被 kill。用户可以通过 **命令** 和 **参数** 选项自定义容器的启动命令和启动参数。**端口** 用于指定容器需要暴露的端口，端口协议可以选择 TCP 和 UDP，用户还可以指定端口与主机端口进行绑定。**环境变量** 可以指定容器内部使用的环境变量。上

述配置信息填写完成以后，点击下一步。



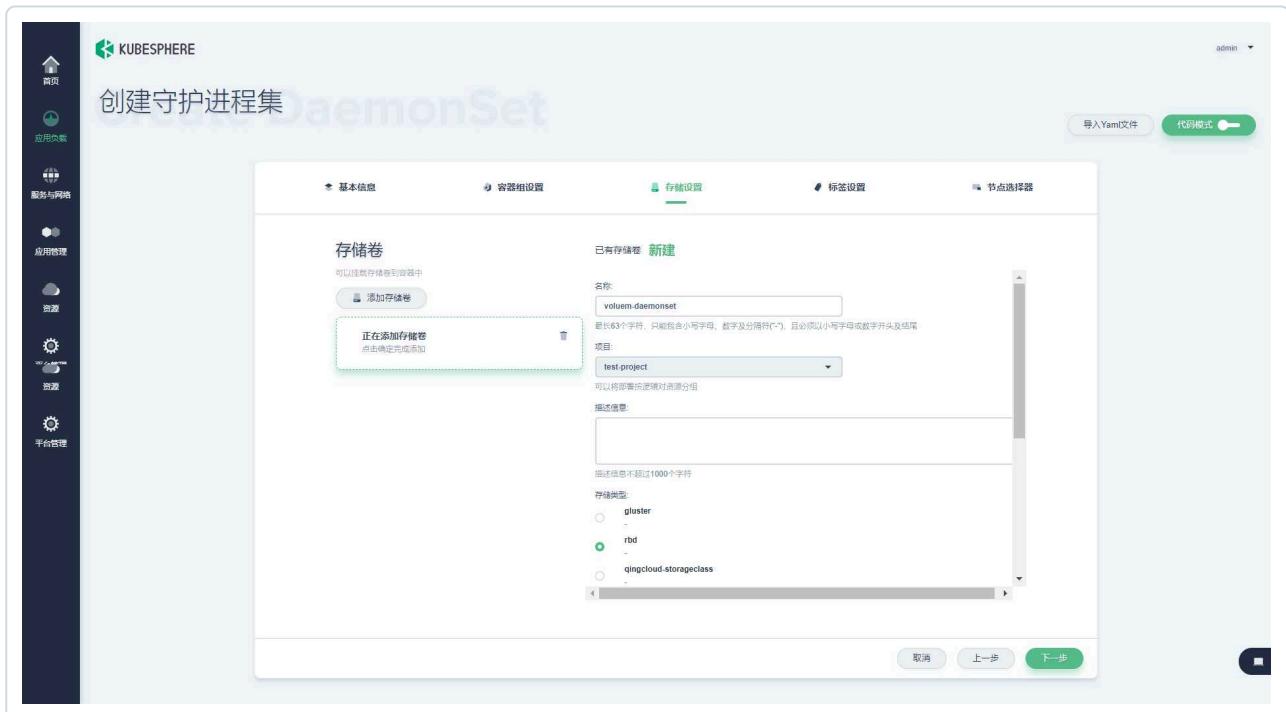
- 在存储卷页面可以添加 **存储卷**, **临时存储卷**, **HostPath**。存储卷可用于持久化用户数据；保留在临时存储卷中的数据，在容器组被删除以后，也将被同步删除。HostPath 直接将主机目录挂载到容器。



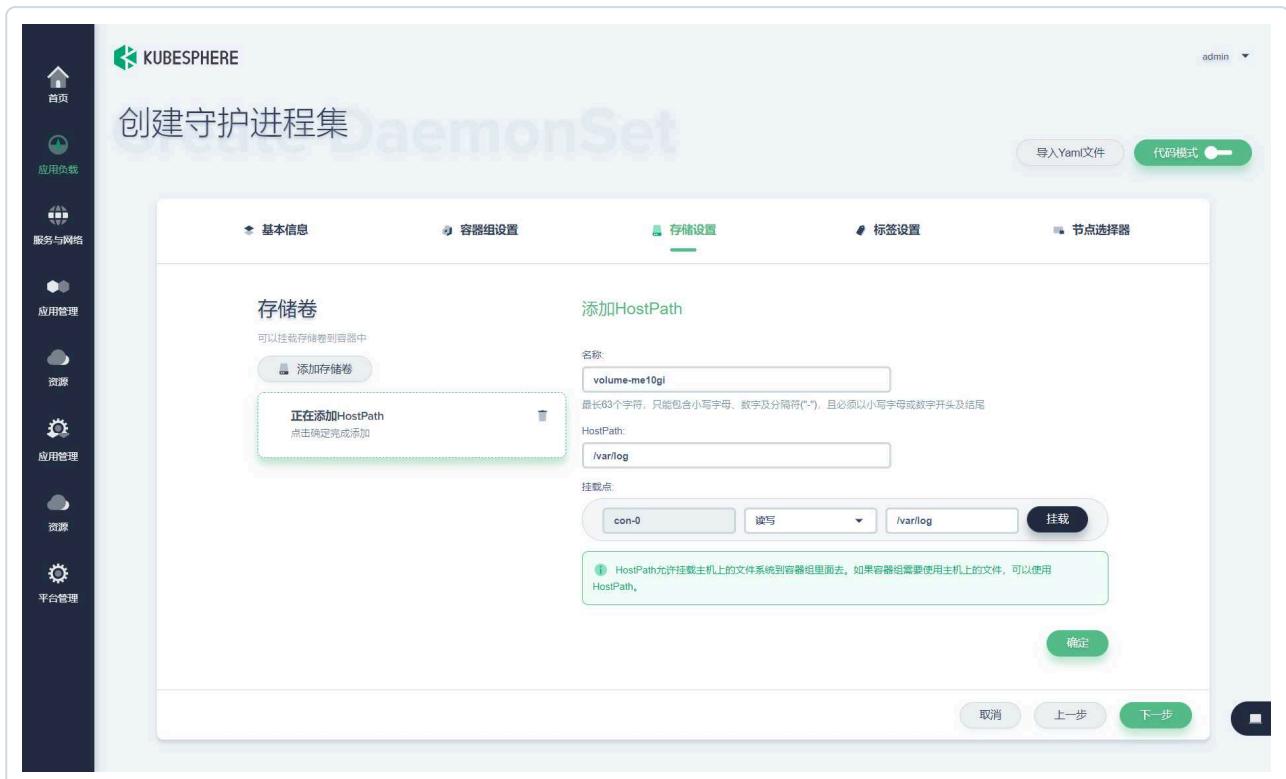
选择使用临时存储卷方案时要求用户输入存储卷的名称和相应的挂载点，然后点击 **挂载** 和 **确定**。



当选择使用存储卷时，可以选择 **已有存储卷** 或者根据需求 **新建存储卷**。当选择新建存储卷时，用户需要填写存储卷的名称，选择存储类型，指定容量和访问模式。最后确定挂载点，并且点击挂载和确定。然后点击下一步。



当选择使用 HostPath 时，用户需要填写存储卷的名称，指定 HostPath 和容器内部的挂载点，点击挂载和确定。然后点击下一步。



4. 标签设置页用于指定守护进程集对应的一组或者多组标签。



5. 节点选择器页面，用户可以通过指定一组或者多组键值对来指定期望运行容器组的主机。当不指定时，将会在集群内的所有节点上启动容器组。点击右下角的创建后，集群就会按照用户的配置创建对应的守护进程集。



查看守护进程集详情

- 在守护进程集列表页，点击某个守护进程集的名称，就可以进入到守护进程集详情页，可以查看守护进程集的详细信息以及守护进程集对应的各种资源。

KUBESPHERE

守护进程集 / fluentd-elasticsearch

fluentd-elasticsearch 资源信息 事件

容器组

名称	状态	主机	容器组 IP	CPU	内存
fluentd-elasticsearch-8k7hr	运行中	i-3gr5u2pa 192.168.0.41	10.244.3.32	0.00m	0.00MiB
fluentd-elasticsearch-mg98n	运行中	i-nd05x6f 192.168.0.42	10.244.2.222	0.00m	0.00MiB
fluentd-elasticsearch-w5q69	运行中	i-78r2dq6l 192.168.0.39	10.244.0.11	0.00m	0.00MiB
fluentd-elasticsearch-x2vbm	运行中	i-7l9r3qp 192.168.0.40	10.244.1.178	0.00m	0.00MiB

存储卷

/var/lib/docker/containers HostPath

/var/log HostPath

- 当点击容器组列表的某一容器组时，页面会跳转到容器组的详情页。

KUBESPHERE

守护进程集 / fluentd-elasticsearch / fluentd-elasticsearch-8k7hr

fluentd-elasticsearch-8k7hr 资源信息 事件

容器

名称	状态	镜像	端口	重启次数	监控
fluentd-elasticsearch	运行中	k8s.gcr.io/fluentd-elasticsearch...	-	0	CPU 7.00m 内存 52.90MiB

存储卷

/var/lib/docker/containers HostPath

/var/log HostPath

- 当点击容器组页面中容器列表中的某一容器时，页面会跳转到容器详情页，可以看到容器的配置信息，对应资源以及日志。在本页，用户还可以通过左上角的 **终端** 按钮进入容器内部。

编辑守护进程集

- 在守护进程集的列表页和详情页的更多选项中都可以编辑对应的配置文件，编辑是在代码模式下完成的，编辑完成以后，点击右下方的更新按钮，就会按照最新的配置进行更新。

- 如果只是修改描述信息，可以使用守护进程集详情页左上角的 编辑 进行修改。

KUBESPHERE

守护进程集 / fluentd-elasticsearch

fluentd-elasticsearch

资源信息

编辑

标签

l8s-app-fluentd-logging

详情

项目: test-project

应用:

节点选择器:

可用节点数: 4

期望节点数: 4

当前节点数: 4

更新策略: 滚动更新

名称: fluentd-elasticsearch

最长253个字符，只能包含小写字母、数字及分隔符("-")，且必须以小写字母或数字开头及结尾

项目: test-project

可以将部署按逻辑对资源分组

描述信息:

描述信息不超过1000个字符

取消 确定

CPU 6.00m 内存 64.90MiB

CPU 3.00m 内存 65.70MiB

CPU 3.00m 内存 50.10MiB

CPU 7.00m 内存 51.00MiB

删除守护进程集

- 在列表页最右侧的更多选项和详情页左上方的更多选项中都提供了删除功能。

KUBESPHERE

守护进程集

共 7 个守护进程集

全部项目

名称 状态 节点选择器 应用 项目 更新时间

fluentd-elasticsearch	运行中(4/4)	-	-	kube-system	2018年7月16日 17:58:12
nnnnnnnnnn-fluentd-cloudwat...	运行中(3/3)	-	nnnnnnnnnn-fluentd-cloudwatch-0.4.3	demo-fu	2018年7月16日 17:58:12
nginx-hostpath	运行中(3/3)	-	-	default	2018年7月16日 17:58:12
nginx-rbd-test	更新中(1/3)	-	-	default	2018年7月16日 17:58:12
kube-proxy	运行中(4/4)	-	-	kube-system	2018年7月16日 17:58:12
fluent-bit	运行中(4/4)	-	-	kube-system	2018年7月16日 17:58:12
kube-flannel-ds	运行中(4/4)	beta.kubernetes.io/arch=amd64	-	kube-system	2018年7月16日 17:58:12

/ 编辑YAML文件
删除

The screenshot shows the KubeSphere UI for managing applications. On the left, a sidebar lists various management categories: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area displays the 'fluentd-elasticsearch' application under the '守护进程集 / fluentd-elasticsearch' section. The top navigation bar includes '资源信息' (Resource Information) and '事件' (Events). The '容器组' (Container Group) section lists four pods: fluentd-elasticsearch-8k7hr, fluentd-elasticsearch-mg98n, fluentd-elasticsearch-w5q69, and fluentd-elasticsearch-x2vbm. Each pod entry includes the pod name, status (Running), IP address, and resource usage (CPU and Memory). Below the container group, there are two storage volume sections: '/var/lib/docker/containers' and '/var/log'. A modal window titled 'fluentd-elasticsearch' is open, showing options to '编辑YAML文件' (Edit YAML file) or '删除' (Delete).

服务管理

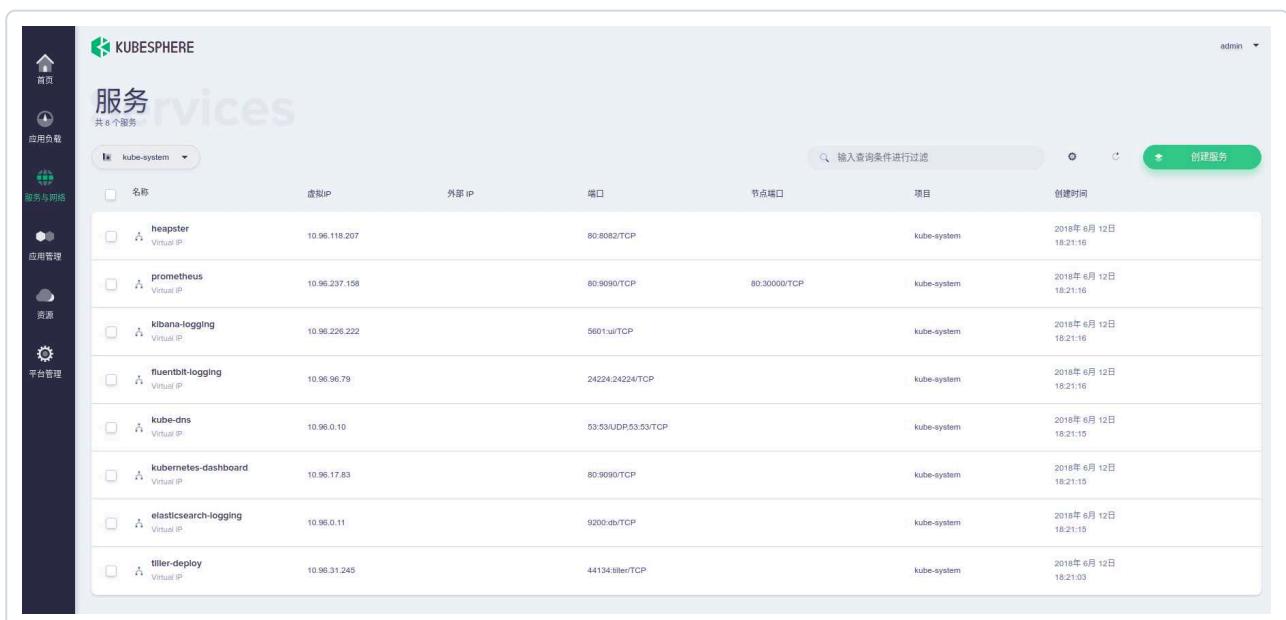
一个 Kubernetes 的服务 (Service) 是一种抽象，它定义了一类 Pod 的逻辑集合和一个用于访问它们的策略 – 有的时候被称之为微服务，而在这个集合中的 Pod 的 IP 地址以及数量等都会发生动态变化，这个服务的客户端并不需要知道这些变化，也不需要自己来记录这个集合的 Pod 信息，这一切都是由抽象层 Service 来完成。

本节通过以下几个方面介绍如何管理服务：

- 创建服务
- 查看服务详情

创建服务

首先登录 KubeSphere 控制台，在左侧导航栏选择 **服务与网络** 菜单的 **服务** 进入到服务列表页面。



The screenshot shows the KubeSphere control panel with the sidebar navigation bar. The 'Services' section is selected, displaying a list of 8 services. Each service entry includes a checkbox, the service name, its virtual IP, external IP, port, node port, project, and creation time. A green 'Create Service' button is located at the top right of the list.

名称	虚拟IP	外部IP	端口	节点端口	项目	创建时间
heapster Virtual IP	10.96.118.207		80:8082/TCP		kube-system	2018年 6月 12日 18:21:16
prometheus Virtual IP	10.96.237.158		80:9090/TCP	80:30000/TCP	kube-system	2018年 6月 12日 18:21:16
kibana-logging Virtual IP	10.96.226.222		5601:ui/TCP		kube-system	2018年 6月 12日 18:21:16
fluentbit-logging Virtual IP	10.96.96.79		24224:24224/TCP		kube-system	2018年 6月 12日 18:21:16
kube-dns Virtual IP	10.96.0.10		53:53/UDP,53:53/TCP		kube-system	2018年 6月 12日 18:21:16
kubernetes-dashboard Virtual IP	10.96.17.83		80:9090/TCP		kube-system	2018年 6月 12日 18:21:16
elasticsearch-logging Virtual IP	10.96.0.11		9200:db/TCP		kube-system	2018年 6月 12日 18:21:16
tiller-deploy Virtual IP	10.96.31.245		44134:tiller/TCP		kube-system	2018年 6月 12日 18:21:03

1. 填写服务的名称，注意不能和现有项目下的其它服务名称相同。

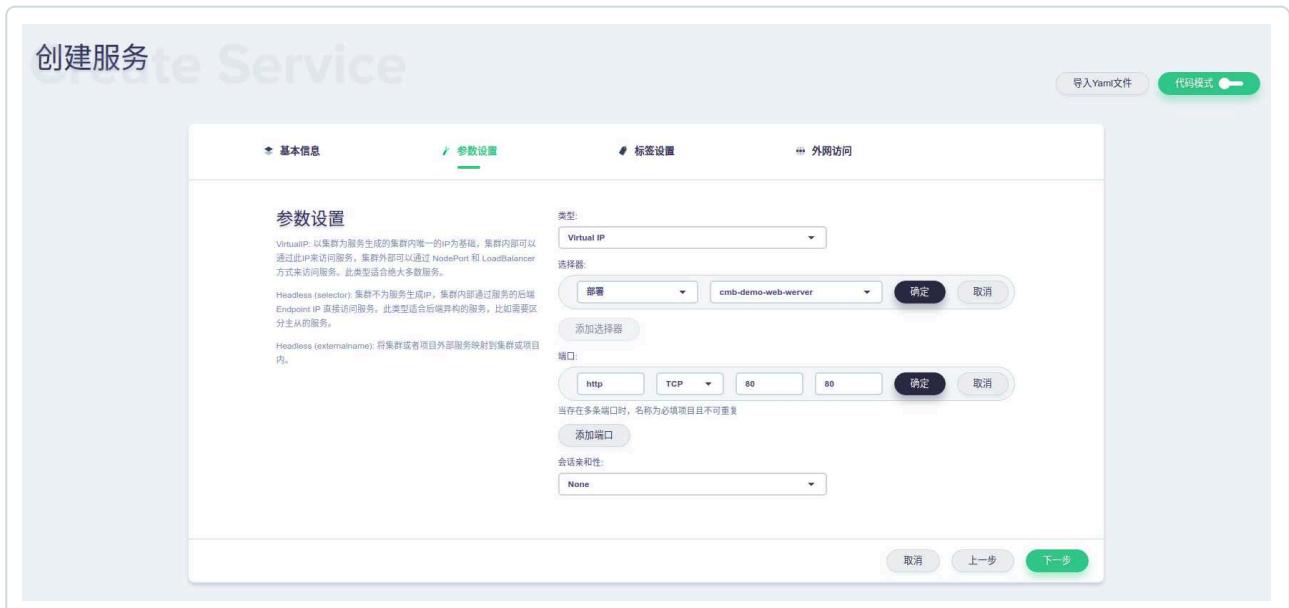
2. 选择需要创建服务的类型，每种服务类型适合不同的场景：

- VirtualIP: 以集群为服务生成的集群内唯一的 IP 为基础，集群内部可以通过此 IP 来访问服务，集群外部可以通过 NodePort 和 LoadBalancer 方式来访问服务。此类型适合绝大多数

服务。

- Headless (selector): 集群不为服务生成 IP，集群内部通过服务的后端 Pod IP 直接访问服务。此类型适合后端异构的服务，比如需要区分主从的服务。
- Headless (externalname): 将集群或者项目外部服务映射到集群或项目内。

3. 您可以使用选择器来选择不同的后端，使用键值对 (Label Selector) 可以选择多个部署。填写完成服务的端口号和对应的后端端口号



4. 为即将创建的服务添加标签，让 KubeSphere 更好的管理资源。
5. 为服务选择外网访问方式。LoadBalancer 的方式需要对应的负载均衡器插件来启用，如果未安装插件则无法使用，也可以在服务创建完成后修改外网访问方式。

- None: 只在集群内部访问服务，集群外部无法访问。
- NodePort: 集群外部可以通过访问集群节点的对应端口来访问服务，端口将由集群自动创建。
- LoadBalancer: 通过云服务商提供的负载均衡器来访问服务。



6. 创建完成后即可在服务列表中查看到已创建的服务。

查看服务详情

在服务列表里点击服务即可进入服务详情页面。可以在详情页面查看到服务的具体信息，也可以对服务进行修改编辑操作。譬如修改服务的外网访问方式，删除服务等。

The screenshot shows the 'Resource Information' tab of the 'ks-console' service details page. The top navigation bar includes back, forward, and search icons, along with tabs for '资源信息' (Resource Information) and '事件' (Events). The '资源信息' tab is selected and highlighted in green. The page displays various service configurations:

- Ports:** A table showing a single port mapping: Name: -, Protocol: TCP, Port: 80, Target Port: 8000.
- 标签 (Labels):** app=kubesphere, component=ks-console, tier=frontend.
- 详情 (Details):** Project: kubesphere-system, Type: Virtual IP, Virtual IP: 10.96.135.138, External IP: -, Node Port: -, Session Affinity: None, Selector: app = kubesphere, component = ks-console, tier = frontend, DNS: ks-console.kubesphere-system, Endpoints: 10.244.193.8000, Creation Time: 2018年6月21日 20:41:59.
- 容器组 (Container Groups):** A table showing one container group: Name: ks-console-6d8f56f5c6-cwhnj, Status: 运行中 (Running), Host: i-7ilraqs, Container IP: 10.244.1.193. Below the table are CPU and Memory usage graphs.

说明：在集群中访问服务时，同一项目下可以直接使用服务的 DNS 名称来访问。如上图，在项目 `kubepshere-system` 下可以直接使用 `ks-console:80` 来访问服务；在其它项目下，需要加上项目后缀，使用 `ks-console.kubesphere-system:80` 来访问服务。

修改外网访问方式

点击左侧 “...” 按钮来编辑服务外网访问方式。

注：修改外网访问方式可能导致其它依赖此服务的应用不可用，请谨慎操作。

服务 / ks-console

资源信息

标签

编辑Yaml文件

编辑外网访问

删除

项目: kubesphere-system

类型: Virtual IP

虚拟IP: 10.96.135.138

外部IP:

节点端口: -

会话亲和性: None

选择器: app = kubesphere, component = ks-console, tier = frontend

DNS: ks-console.kubesphere-system

Endpoints: 10.244.1.193:8000

创建时间: 2018年 6月 21日 20:45

资源信息

事件

Ports

名称	协议	端口	目标端口
-	TCP	80	8000

Pods

名称	状态	应用	创建时间
ks-console	运行中[1]	-	2018年 6月 21日 20:41:59

容器组

名称	状态	主机	容器组IP	监控
ks-console-6d8f56f5c6-cwhnj	运行中	i-7lirraqs	10.244.1.193	CPU 3.00m 内存 112.70MB

应用路由管理

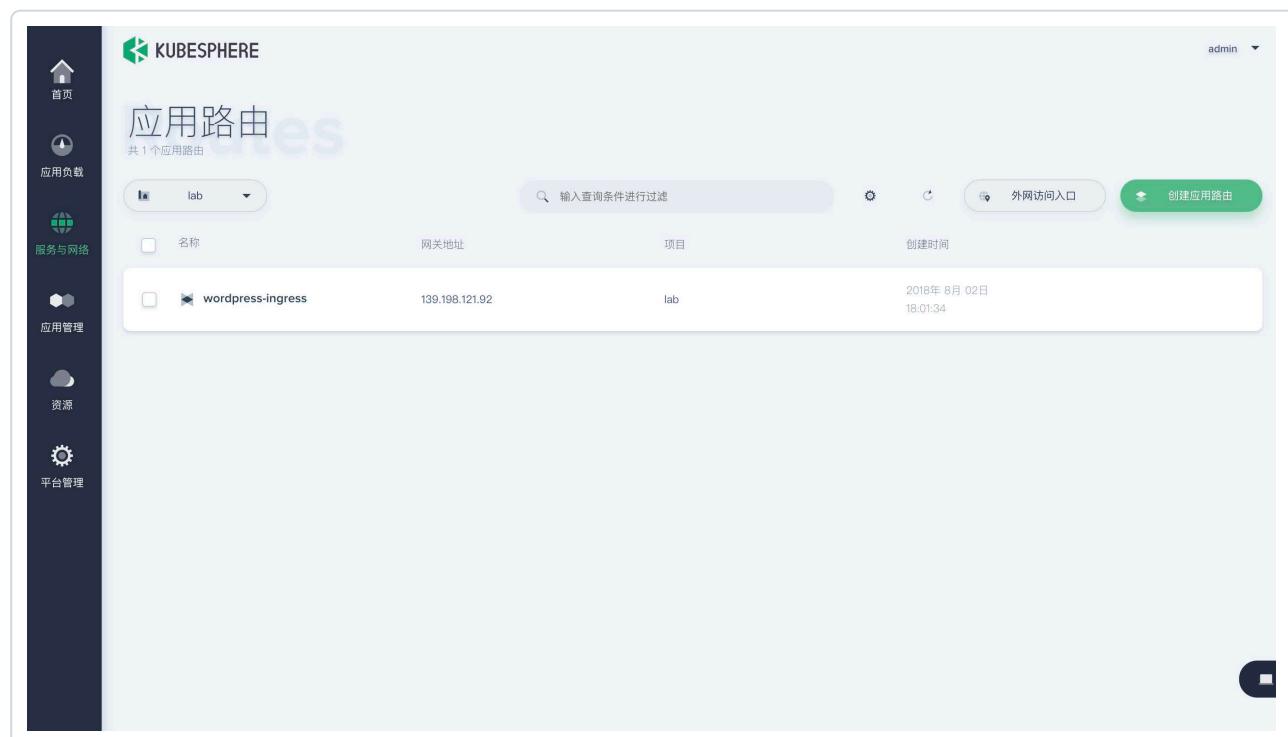
应用路由是来聚合集群内服务的方式，对应的是 Kubernetes 的 Ingress 资源，后端使用了 Nginx Controller 来处理具体规则。Ingress 可以给 service 提供集群外部访问的 URL、负载均衡、SSL termination、HTTP 路由等。

本节通过以下几个方面介绍如何管理应用路由：

- 启用外网访问入口
- 创建应用路由
- 访问应用路由

启用外网访问入口

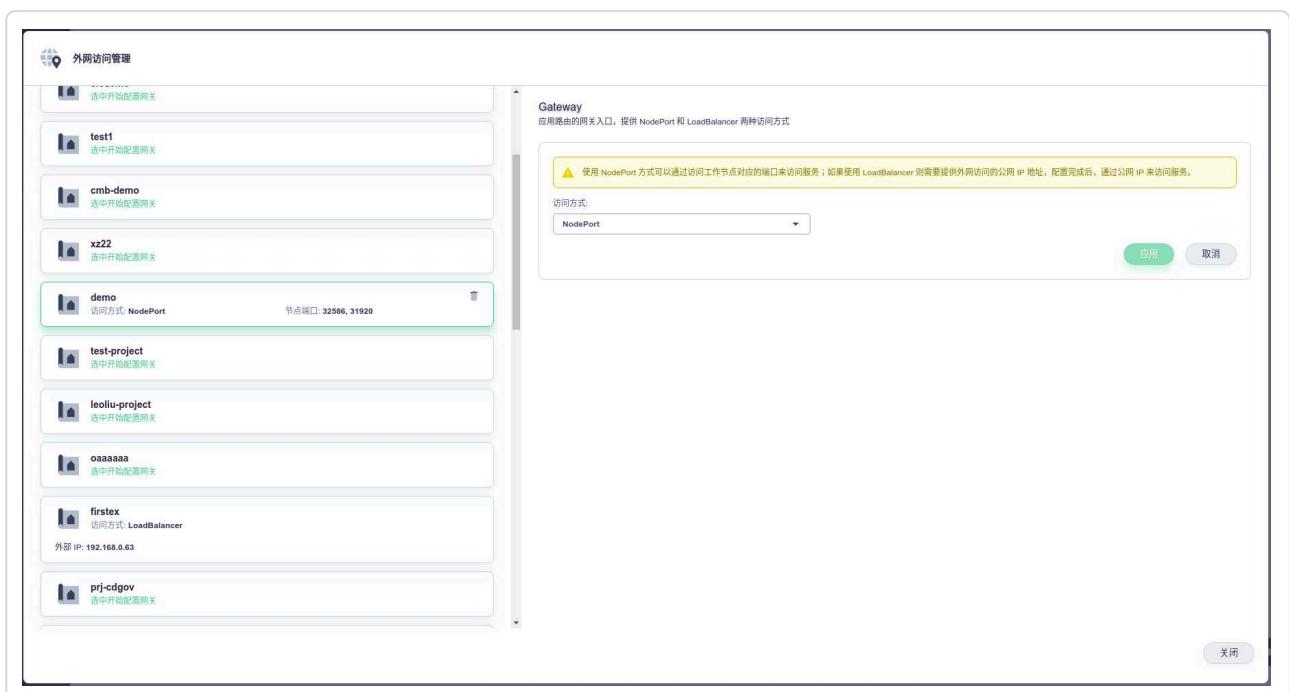
首先登录 KubeSphere 控制台，在左侧导航栏选择 **服务与网络** 菜单的 **应用路由** 进入到应用路由列表页面。



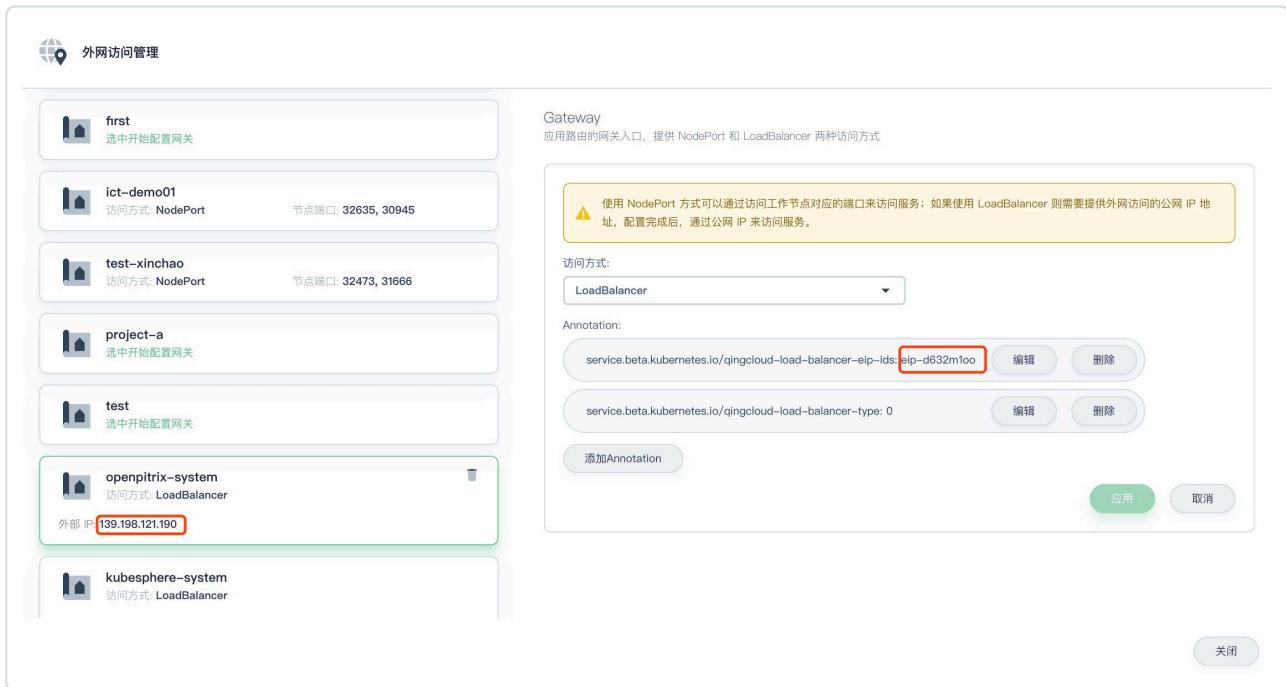
名称	网关地址	项目	创建时间
wordpress-ingress	139.198.121.92	lab	2018年 8月 02日 18:01:34

在创建应用路由之前，需要先启用外网访问入口，即网关。这一步是创建对应的应用路由控制器，来负责将请求转发到对应的后端服务。如不预先启用外网访问入口，创建的应用路由规则将无法访问。

1. 点击应用路由规则列表上方的 **外网访问入口** 按钮。
2. 在弹出的窗口左侧选择需要启用的项目，在右侧选择网关的类型。每个项目都有一个独立的网关。
 - NodePort: 此方式网关可以通过工作节点对应的端口来访问
 - LoadBalancer: 此方式网关可以通过统一的一个外网 IP 来访问 (需要对应的负载均衡器插件)
3. 选择对应的网关启用方式后，点击 **应用** 来创建网关，如下图选择的是 NodePort 的方式，左边节点端口生成的两个端口，分别是 HTTP 协议的 80 端口和 HTTPS 协议的 443 端口。完成后选择关闭。



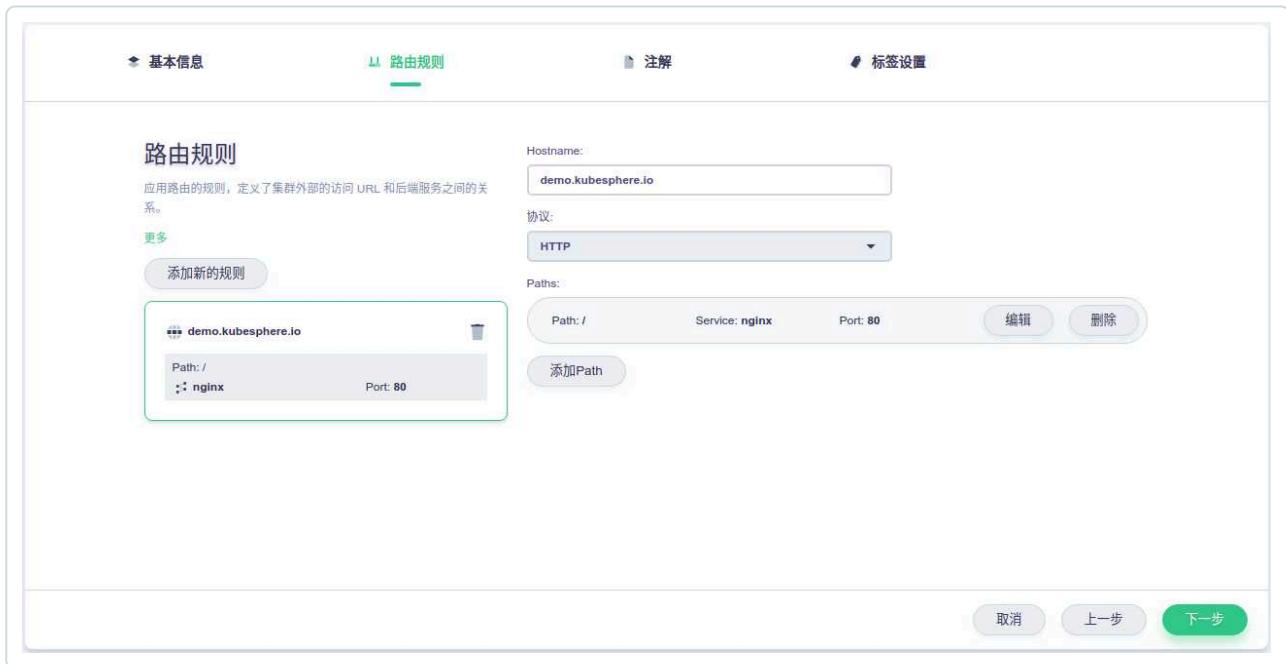
4. 若选择的是 LoadBalancer，则需要将公网 IP 的 ID 填入 Annotation，如下图所示，点击 **应用** 来创建网关。



创建应用路由

应用路由需要先启用外网访问入口，请按照上面步骤启用外网访问入口

1. 点击应用路由规则右上角的 **创建应用路由** 按钮，填写应用路由的名称。
2. 填写应用路由的规则，如下图所示。
 - Hostname: 应用规则的访问域名，最终使用此域名来访问对应的服务。如果访问入口是以 NodePort 的方式启用，需要保证 Host 能够在客户端正确解析到集群工作节点上；如果是以 LoadBalancer 方式启用，需要保证 Host 正确解析到负载均衡器的 IP 上。
 - Path: 应用规则的路径和对应的后端服务，端口需要填写成服务的端口。



3. 点击下一步，为应用规则添加注解。如无特殊设置，这一步可以跳过。

4. 添加标签后，创建完成。

访问应用路由

应用路由创建完成后，确保设置的域名可以解析到外网访问入口的 IP 地址，即可使用域名访问。如在私有环境中，可以使用修改本地 hosts 文件的方式来使用应用路由。例如，

设置的域名	路径	外网访问入口方式	端口/IP	集群工作节点IP
demo.kubesphere.io	/	NodePort	32586,31920	192.168.0.4,192.168.0.3,192.168.0.2
demo2.kubesphere.io	/	LoadBalancer	139.198.1.1	192.168.0.4,192.168.0.3,192.168.0.2

如上表格，创建了两条应用路由规则，分别使用 NodePort 和 LoadBalancer 方式的访问入口。

NodePort

对于外网访问方式设置的 NodePort，如果是在私有环境下，我们可以直接在主机的 hosts 文件中添加记录来使域名解析到对应的 IP。例如，对于 `demo.kubesphere.io`，我们添加如下记录：

192.168.0.4 demo.kubesphere.io

需要保证客户端与集群工作节点 192.168.0.4 网络可通，可以使用其它工作节点的 IP，只要客户端和工作节点网络是通的，设置完之后，在浏览器中使用域名和网关的端口号

<http://demo.kubesphere.io:32586> 即可访问。(此示例中用的是第一个端口 32586，它对应的 HTTP 协议的 80 端口，目前仅支持 HTTP 协议，将在下个版本中支持 HTTPS 协议。)

LoadBalancer

如果外网访问方式设置的是 LoadBalancer，对于 demo2.kubesphere.io，除了参考以上方式在 hosts 文件中添加记录之外，还可以使用 nip.io 作为应用路由的域名解析。nip.io 是一个免费的域名解析服务，可以将符合下列格式的域名解析对应的ip，可用来作为应用路由的解析服务，省去配置本地 hosts 文件的步骤。

格式

```
10.0.0.1.nip.io maps to 10.0.0.1
app.10.0.0.1.nip.io maps to 10.0.0.1
customer1.app.10.0.0.1.nip.io maps to 10.0.0.1
customer2.app.10.0.0.1.nip.io maps to 10.0.0.1
otherapp.10.0.0.1.nip.io maps to 10.0.0.1
```

例如，应用路由的网关公网IP地址为 139.198.121.154，在创建应用路由时，Hostname 一栏填写为 demo2.kubesphere.139.198.121.154.nip.io，其它保持原来的设置。

The screenshot shows the KubeSphere UI for creating a route rule. On the left sidebar, there are several navigation items: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main area has tabs for 基本信息 (Basic Information), 路由规则 (Route Rule) (which is selected), 注解 (Annotations), and 标签设置 (Label Settings). The Route Rule tab contains fields for Hostname (demo2.kubesphere.139.198.121.154.nip.io), 协议 (Protocol) (HTTP), and Paths (Path: /, Target: ks-console, Port: 80). A '确定' (Confirm) button is visible. At the bottom, there are '取消' (Cancel), '上一步' (Previous Step), and '下一步' (Next Step) buttons.

创建完成后，直接使用 <http://demo2.kubesphere.139.198.121.154.nip.io>，即可访问对应的服务。

The screenshot shows a browser window with the URL <http://demo2.kubesphere.139.198.121.154.nip.io>. The page displays a login form for KubeSphere. It includes fields for 账户邮箱 (Account Email) containing 'User@example.com' and 密码 (Password). A green '登录' (Login) button is at the bottom right. The background features a dark blue design with white geometric shapes.

应用模板

应用模板相当于 Helm 应用仓库中的 chart 模板，通过可视化的方式在 KubeSphere 中展示并提供部署及管理功能，用户可以基于应用模板快速地一键部署应用。

部署应用

参考 [添加应用仓库说明](#) 来添加应用仓库，KubeSphere 会自动加载此仓库下的所有应用。在左侧菜单栏 **应用管理** 菜单下，点击 **应用模板** 按钮进入列表页。注意：默认会将来自所有仓库的应用模板都列出来，可以通过左上角的下拉框进行来源过滤。

The screenshot shows the KubeSphere application template interface. On the left is a sidebar with navigation links: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management) (which is selected and highlighted in blue), and 资源 (Resources). The main content area has a header with the title '应用模板' (Application Templates) and a note '共 464 个应用模板'. Below the header is a search bar with placeholder '输入查询条件进行过滤' (Filter by input query conditions) and a red box highlighting the filter dropdown menu which is set to '全部资源' (All Resources). There are two tabs: '仓库来源' (Repository Source) and '最新发布' (Latest Releases), with '仓库来源' currently selected. The main area displays a grid of application templates. Each card includes the name, repository source, description, latest version, and a '应用详情' (View Details) button. The cards shown are: rookout (仓库: QingStor), hazelcast (仓库: Google), hazelcast (仓库: QingStor), rookout (仓库: Google), kuberos (仓库: Google), kuberos (仓库: QingStor), efs-provisioner (仓库: Google), and efs-provisioner (仓库: Google).

1. 找到所需的应用，点击 **应用详情** 按钮。
2. 弹出窗口左侧为当前应用的详细信息和说明文档，右侧为需要填写的和部署相关的参数。

- 名称：为所部署应用命名，如不填，则自动生成一个随机名称
- 版本：选择所需应用版本
- 项目：选择应用部署的目标项目
- 描述信息：关于被部署应用的一些详细信息
- 配置参数：以可视化的方式，将 Helm Chart 应用包中的 values.yaml 展示出来，里面包含了应用开发者允许用户自定义的参数，用户可以根据实际情况进行填写

tomcat
Deploy a basic tomcat application server with sidecar as web archive container
关键字: -
维护者: yahavb

Tomcat
Tomcat is an open source implementation of the Java Servlet, JavaServer Pages, Java Expression Language and Java WebSocket technologies.

Introduction
This chart creates a tomcat application server Deployment, plus http Services for the server. The chart offers an optimization for application updates running in a servlet container-type engines like tomcat and JBoss. The chart uses the WAR, EAR, and other deployable components outside of the Servlet engine as sidecar container so application upgrades requires the sidecar container image only to be updated and not the Servlet engine as if both would run at the same image.

Prerequisites
• Kubernetes 1.8+

Provider-specific Prerequisites

Installing the Chart
To install the chart with the release name `my-release`:
`$ helm install --name my-release stable/tomcat`
This command deploys a tomcat dedicated server with sane defaults.

Setting
Setting 定义应用配置参数

replicaCount: 1	image.webarchive.repository: ananwaresystems/webarchive
image.webarchive.tag: 1.0	image.tomcat.repository: tomcat
image.tomcat.tag: 7.0	image.pullPolicy: IfNotPresent
deploy.directory: /usr/local/tomcat/webapps	service.name: http
service.type: LoadBalancer	service.externalPort: 80
service.internalPort: 8080	ingress.enabled: false

取消 部署

3. 点击右上角 代码模式 按钮，可以查看当前应用中的所有文件的详细内容。

tomcat
Deploy a basic tomcat application server with sidecar as web archive container
关键字: -
维护者: yahavb

Tomcat
Tomcat is an open source implementation of the Java Servlet, JavaServer Pages, Java Expression Language and Java WebSocket technologies.

Introduction
This chart creates a tomcat application server Deployment, plus http Services for the server. The chart offers an optimization for application updates running in a servlet container-type engines like tomcat and JBoss. The chart uses the WAR, EAR, and other deployable components outside of the Servlet engine as sidecar container so application upgrades requires the sidecar container image only to be updated and not the Servlet engine as if both would run at the same image.

Prerequisites
• Kubernetes 1.8+

Provider-specific Prerequisites

Installing the Chart
To install the chart with the release name `my-release`:

配置文件 (选填)
可以查看应用模板配置信息

values.yaml
Yaml 格式

```
3 - webarchive:
4   repository: ananwaresystems/webarchive
5   tag: '1.0'
6 - tomcat:
7   repository: tomcat
8   tag: '7.0'
9   pullPolicy: IfNotPresent
10 - deploy:
11   directory: /usr/local/tomcat/webapps
12 - service:
13   name: http
14   type: LoadBalancer
15   externalPort: 80
16   internalPort: 8080
17 - ingress:
18   enabled: 'false'
19   path: /
20   hosts: []
21   tls: []
22   tolerations: []
```

取消 部署

4. 当所需参数输入完成后，点击 部署 按钮进行部署，用户可以在相关工作负载菜单下查看部署进展情况。

应用仓库管理

KubeSphere 基于 [OpenPitrix](#) 构建了应用仓库服务，OpenPitrix 是由 [QingCloud](#) 主导开源的跨云应用管理平台，可以支持基于 Helm 的 Kubernetes 应用。

本节通过以下几个方面介绍如何应用仓库：

- 添加应用仓库
- 查看或编辑应用仓库
- 删减应用仓库

添加应用仓库

首先登录 KubeSphere 管理控制台，访问左侧菜单栏，在 **应用管理** 菜单下，点击 **应用仓库** 按钮进入列表页。

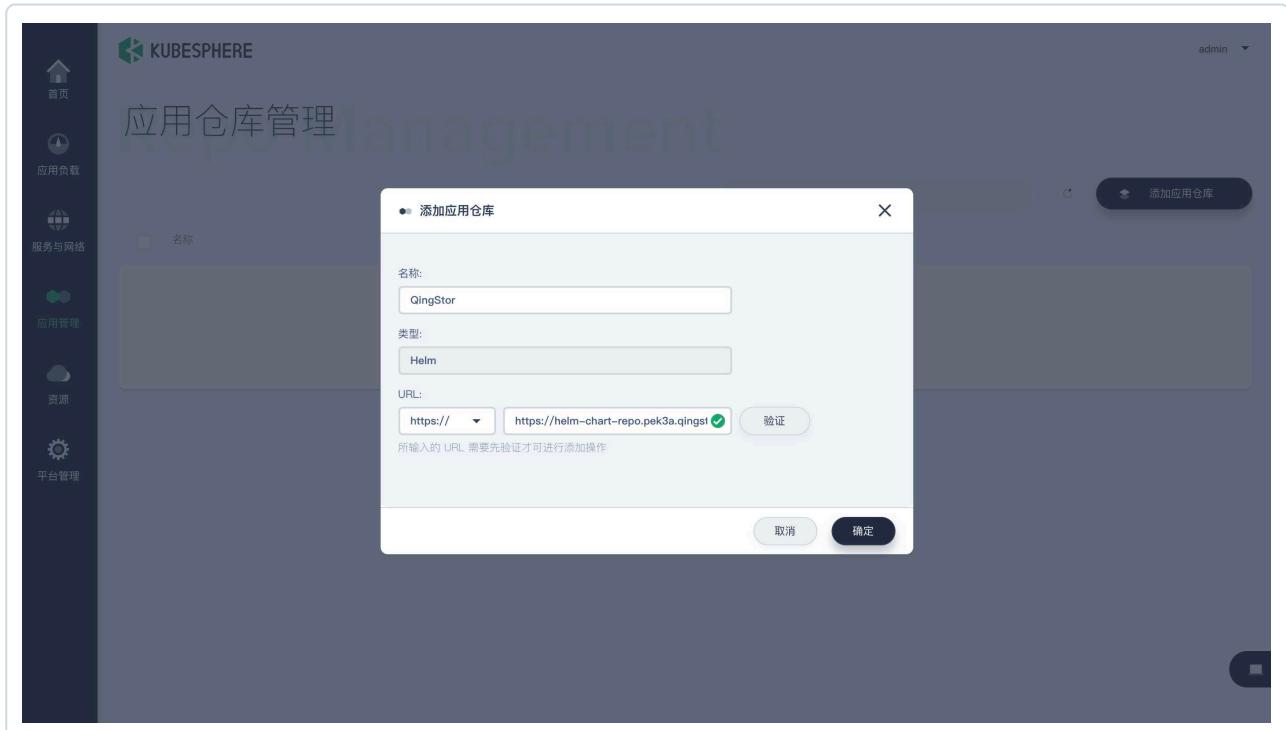


The screenshot shows the KubeSphere management interface with the sidebar menu open. Under the '应用管理' (Application Management) section, the '应用仓库' (Application Repository) button is selected, leading to the '应用仓库管理' (Application Repository Management) page. This page displays a list of currently configured application repositories:

名称	状态	类型	Url
incubator	已启用	Helm	https://kubernetes-charts-incubator.storage.googleapis.com/
Google	已启用	Helm	https://kubernetes-charts.storage.googleapis.com
QingStor	已启用	Helm	https://helm-chart-repo.pek3a.qingstor.com/kubernetes-charts/

1. 点击右上角 **添加应用仓库** 按钮。
2. 在弹出窗口填入应用仓库的基本信息并点击 **验证** 按钮。

- 验证通过后，点击 **确认** 按钮完成应用仓库的添加。当添加应用仓库后，KubeSphere 会自动加载此仓库下的所有应用模板。



Google 有两个应用仓库可以试用，QingStor 对其中稳定的仓库做了一个 mirror (后续我们会开发商业版的应用仓库供企业使用)，用户可根据需要添加所需应用仓库：

- Google Stable Helm Repo: <https://kubernetes-charts.storage.googleapis.com/>
- Google Incubator Helm Repo: <https://kubernetes-charts-incubator.storage.googleapis.com/>
- QingStor Helm Repo: <https://helm-chart-repo.pek3a.qingstor.com/kubernetes-charts/>

在企业内私有云场景下，用户可以基于 [Helm](#) 规范去构建自己的应用仓库，并且可以开发和上传满足企业业务需求的应用到自己的应用仓库中，然后基于 KubeSphere 完成应用的分发部署。

查看或编辑应用仓库

在应用仓库列表页，点击某个应用仓库，选择 **编辑** 按钮，打开详情页，可以查看或者编辑此仓库信息。

删除应用仓库

在应用仓库列表页，点击某个应用仓库，选择 **删除** 按钮，可以删除此仓库信息。



The screenshot shows the KubeSphere Application Repository Management interface. On the left is a sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area has a header "应用仓库管理" (Application Repository Management) with a sub-header "Management". It displays "共 3 个应用仓库" (3 application repositories). A red button at the top left says "删除应用仓库" (Delete Application Repository). Below is a table with columns: 名称 (Name), 状态 (Status), 类型 (Type), and Url. The table contains three rows:

名称	状态	类型	Url
incubator	已启用	Helm	https://kubernetes-charts-incubator.storage.googleapis.com/
Google	已启用	Helm	https://kubernetes-charts.storage.googleapis.com
QingStor	已启用	Helm	https://helm-chart-repo.pek3a.qingstor.com/kubernetes-charts/

At the top right, there are buttons for "删除应用仓库" (Delete Application Repository) and "取消选择" (Cancel Selection). At the bottom right is a circular icon with a square inside.

镜像仓库

Docker 镜像是一个只读的模板，可用于部署容器服务，每个镜像有特定的唯一标识（镜像的 Registry 地址+镜像名称+镜像 Tag）。例如：一个镜像可以包含一个完整的 Ubuntu 操作系统环境，里面仅安装了 Apache 或用户需要的其它应用程序。而镜像仓库，是集中存放镜像文件的场所，镜像仓库用于存放 Docker 镜像。

本节通过以下几个方面介绍如何管理镜像仓库：

- 创建镜像仓库
 - 创建 QingCloud 镜像仓库
 - 创建 Docker Hub 镜像仓库
 - 创建 Harbor 镜像仓库
- 查看镜像仓库
- 修改镜像仓库
- 如何使用镜像仓库
- 删 除 镜 像 仓 库

创建镜像仓库

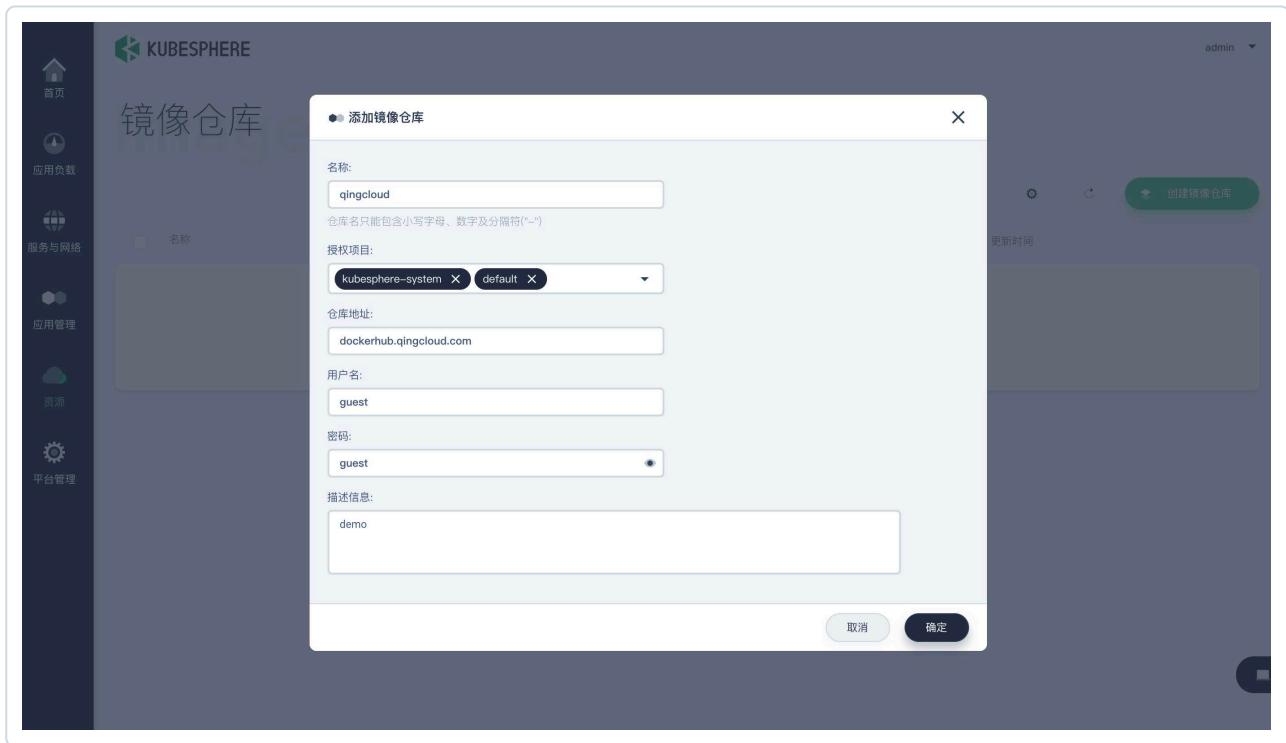
登录 KubeSphere 管理控制台，访问左侧菜单栏，在 **资源** 菜单下，点击 **镜像仓库** 按钮，进入镜像仓库列表页面。作为集群管理员，可以查看当前集群下所有的镜像仓库。

名称	状态	仓库地址	授权项目	仓库用户	更新时间
demo	已启用	dockerhub.qingcloud.com	app-test	alex.tan	2018年7月16日 15:32:31
index.docker.io	已启用	index.docker.io	calvin-test05	beer2100	2018年7月16日 13:50:04

创建 QingCloud 镜像仓库

1. 点击右上角 **创建镜像仓库** 按钮，弹出添加镜像仓库对话框，填写镜像仓库所需要的信息。
2. 信息确认无误后，点击 **确定** 按钮，该过程中有个验证，验证用户所填写的镜像地址、用户名和密码是否正确，如果认证错误请检查镜像仓库所填写的地址信息是否有误。
3. 验证成功即完成镜像仓库的添加。

以下用 QingCloud 镜像仓库地址 `dockerhub.qingcloud.com` 作为示例，参考如下截图完成此镜像仓库的添加：



创建 Docker Hub 镜像仓库

如果需要添加 [Dokcer Hub](#) 中的镜像仓库，请先确保已在 Docker Hub 注册过账号再进行添加，在添加镜像仓库的窗口填写仓库名称并选择授权项目，仓库地址填写 [docker.io](#)，输入用户名和密码即可。

创建 Harbor 镜像仓库

Harbor 简介

[Harbor](#) 是一个用于存储和分发 Docker 镜像的企业级 Registry 服务器，通过添加一些企业必需的功能特性，例如安全、标识和管理等，扩展了开源 Docker Distribution。作为一个企业级私有 Registry 服务器，Harbor 提供了更好的性能和安全。以下详细介绍如何在 KubeSphere 中创建 Harbor 镜像仓库，添加之前请确保已创建了 Harbor 镜像仓库服务端。

根据 Harbor 镜像仓库的地址类型，需要分 http 和 https 两种认证方法：

http

- 首先，需要修改集群中所有节点的 docker 配置。以 <http://139.198.16.232> 为例，在 </etc/>

`systemd/system/docker.service.d/docker-options.conf` 文件添加字段 `--insecure-registry=139.198.16.232` :

示例：

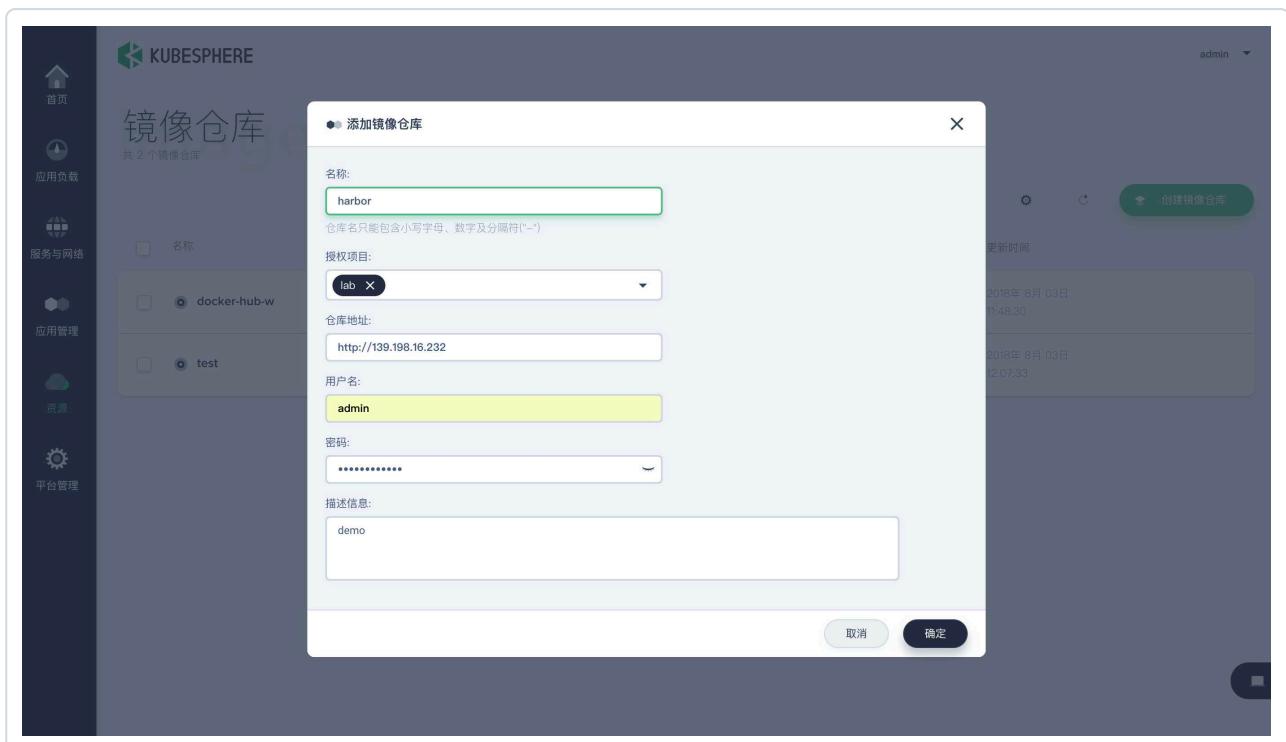
```
[Service]
Environment="DOCKER_OPTS='--registry-mirror=https://registry.docker-cn.com --insecure-registry=10.232.13.13
--iptables=false \
--insecure-registry=139.198.16.232'"
```

2. 添加完成以后，需要重载修改过的配置文件并重启 docker:

```
$ sudo systemctl daemon-reload
```

```
$ sudo systemctl restart docker
```

3. 然后通过 KubeSphere 控制台，填写镜像仓库所需要的信息，创建 Harbor 镜像仓库。



https

- 对于 https 协议的镜像仓库，首先需要获取镜像仓库的证书，记为 `ca.crt`，以 <https://harbor.openpitrix.io> 这个镜像仓库的地址为例，对集群中的所有节点都需要执行以下操作：

```
$ sudo cp ca.crt /etc/docker/certs.d/harbor.openpitrix.io/ca.crt
```

- 如果还是报权限错误，针对不同的操作系统，需要执行以下操作：

UBUNTU

```
$ sudo cp ca.crt /usr/local/share/ca-certificates/harbor.openpitrix.io.ca.crt
```

```
$ sudo update-ca-certificates
```

RED HAT ENTERPRISE LINUX

```
$ sudo cp ca.crt /etc/pki/ca-trust/source/anchors/harbor.openpitrix.io.ca.crt
```

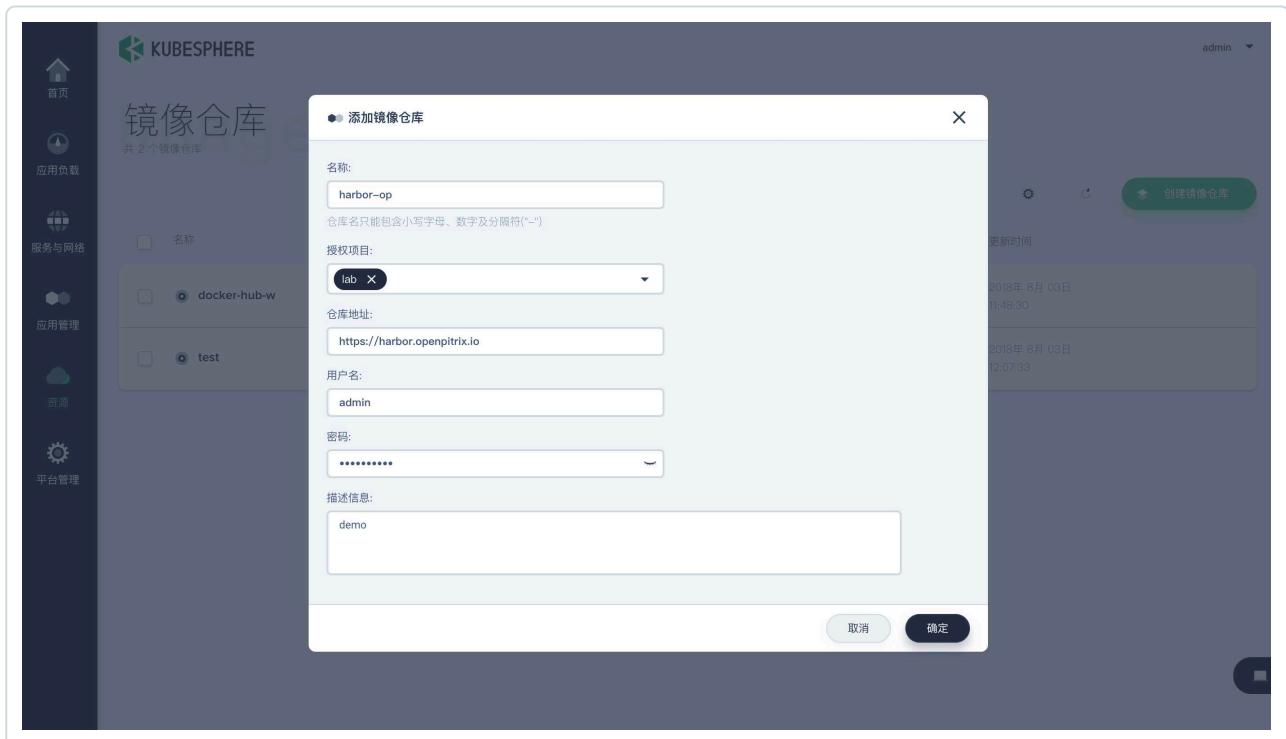
```
$ sudo update-ca-trust
```

- 添加完成以后，需要重载修改过的配置文件并重启 docker（详情可参照 [docker官网](#)）：

```
$ sudo systemctl daemon-reload
```

```
$ sudo systemctl restart docker
```

- 然后通过 KubeSphere 控制台，填写镜像仓库所需要的信息，创建 Harbor 镜像仓库。



查看镜像仓库

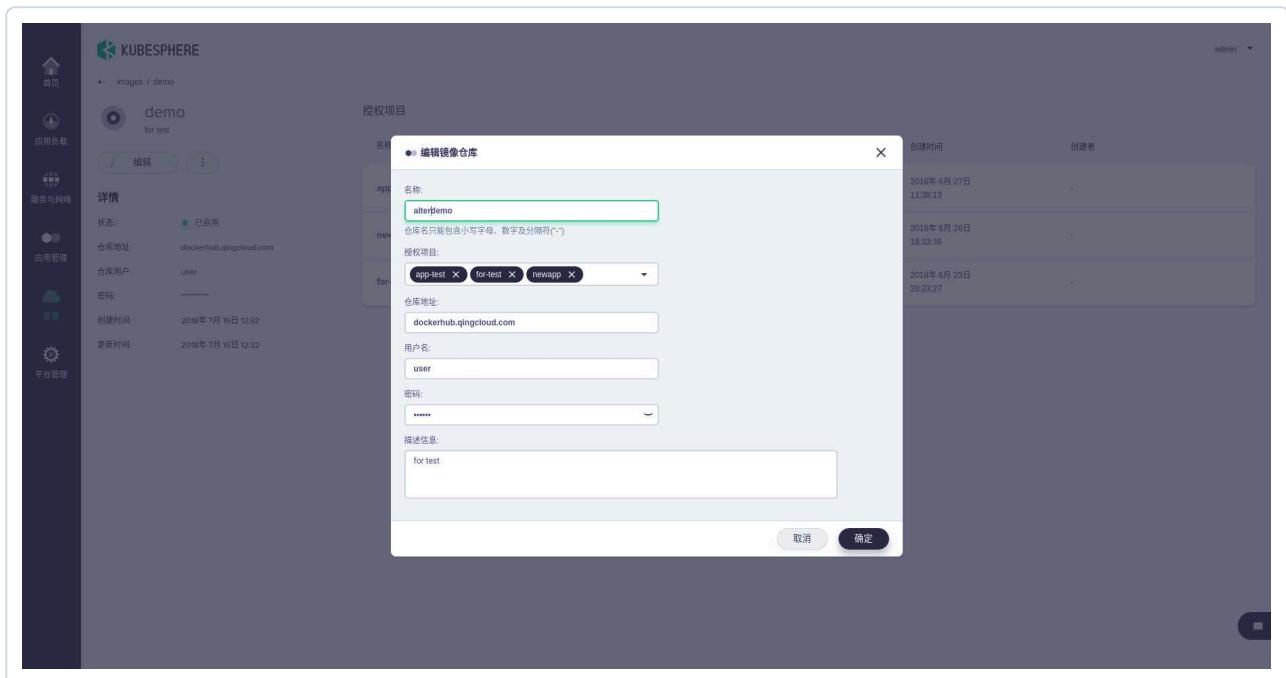
1. 点击列表镜像仓库 名称，进入该镜像仓库详情页面。
2. 可以查看当前镜像仓库的授权项目列表以及镜像仓库的详情信息。

The screenshot shows the KubeSphere interface displaying the details of the "demo" image repository. The repository information includes its name (demo), address (dockerhub.qingcloud.com), user (user), and creation time (2018年7月16日 12:32). To the right, a table lists authorized projects:

名称	状态	描述信息	创建时间	创建者
app-test	已启用	-	2018年6月27日 11:38:13	-
newapp	已启用	-	2018年6月26日 18:33:38	-
for-test	已启用	-	2018年6月23日 20:23:27	-

修改镜像仓库

- 进入镜像仓库详情页面，点击左上角 **编辑** 按钮，可以修改当前镜像仓库信息。



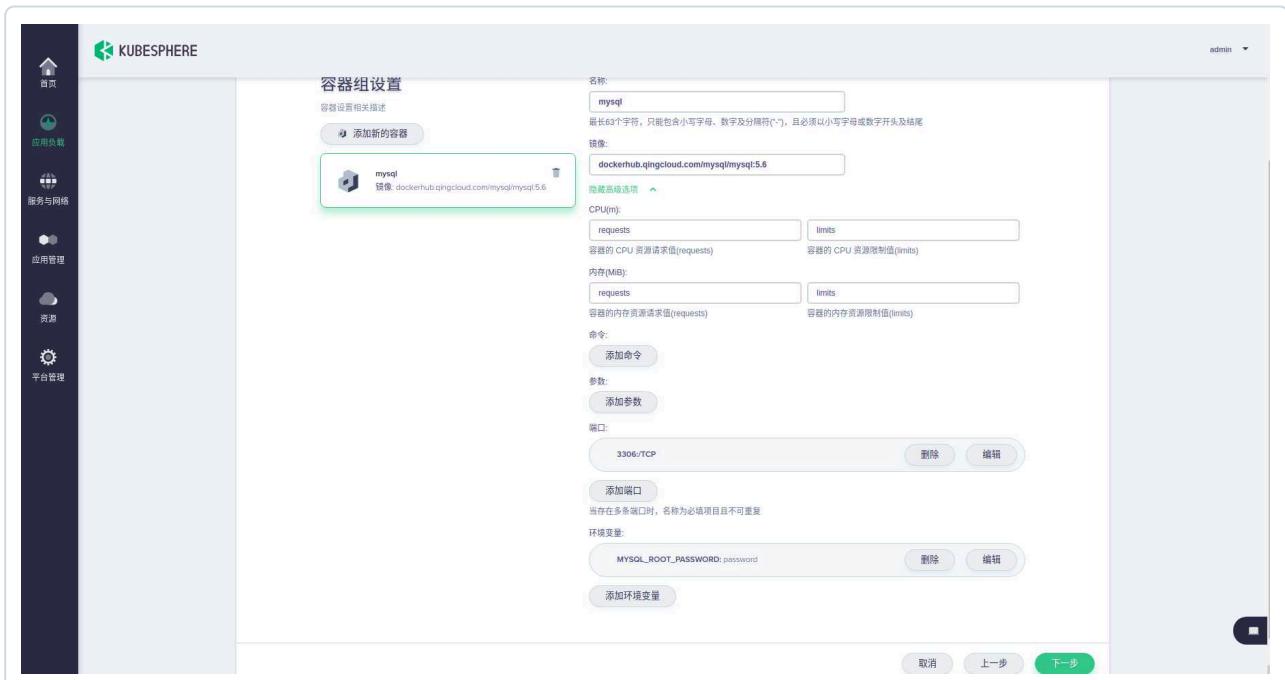
使用镜像仓库

以创建 Deployment 为例展示如何使用镜像仓库来拉取镜像。比如 QingCloud 镜像仓库中有 `mysql:5.6` 的 docker 镜像，镜像地址为 `dockerhub.qingcloud.com/mysql/mysql:5.6`。

- 在镜像仓库已经授权的项目中，创建 deployment。

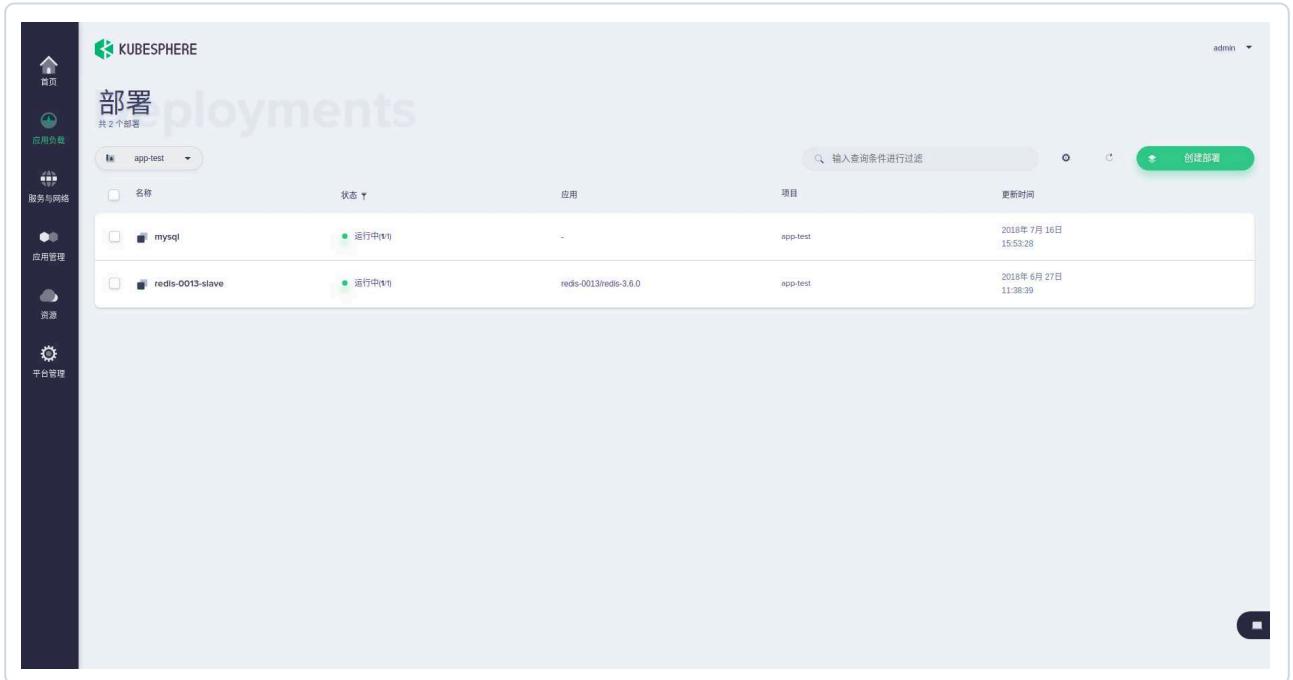


2. 在容器组设置阶段，点击镜像输入框，会弹出授权该项目下的镜像仓库列表，选择 QingCloud 镜像仓库，然后输入具体的镜像仓库地址（命名空间+镜像+ TAG），本例中输入的是 `/mysql/mysql:5.6`。在容器组设置中配置 MySQL 的访问端口和 MySQL 的 `MYSQL_ROOT_PASSWORD` 即 root 用户的密码并保存。高级选项中可以对 CPU 和内存的资源使用进行限定，其中 `requests` 是集群保证分配给容器的资源，`limits` 是容器可以使用的资源的上限，此处暂不作限定。



3. 后续操作请查看 **部署管理**，最终会看到刚才部署的 MySQL 已经运行起来了，说明部署能够正确地

从创建的镜像仓库拉取镜像。



The screenshot shows the KubeSphere deployment management interface. On the left, there is a sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area is titled "Deployments" and shows a list of two deployments under the "app-test" project. The table has columns for Name, Status, Application, Project, and Last Updated. The "mysql" deployment is in "Running" status, and the "redis-0013-slave" deployment is also in "Running" status. Both were last updated on July 18, 2018, at 15:53:28.

名称	状态	应用	项目	更新时间
mysql	运行中(1/1)	-	app-test	2018年 7月 18日 15:53:28
redis-0013-slave	运行中(1/1)	redis-0013:redis-3.6.0	app-test	2018年 6月 27日 11:38:39

删除镜像仓库

- 在镜像仓库列表页 勾选 镜像仓库或者进入详情页面，点击左侧镜像仓库操作菜单，点击 **删除** 按钮删除镜像仓库。

KUBESPHERE

Images / alterdemo

alterdemo
for test

编辑

详情

状态: 已启用

仓库地址: dockerhub.qingcloud.com

仓库用户: user

密码: ****

创建时间: 2018年7月16日 12:32

更新时间: 2018年7月16日 12:34

授权项目

名称	状态	描述信息	创建时间	创建者
app-test	已启用		2018年 6月 27日 11:38:13	
newapp	已启用		2018年 6月 26日 18:33:38	
for-test	已启用		2018年 6月 23日 20:23:27	

KUBESPHERE

Image Registries

共 1 个镜像仓库

Delete Image Registry

取消选择

名称	状态	仓库地址	授权项目	仓库用户	更新时间
demo	已启用	dockerhub.qingcloud.com	app-test	user	2018年 7月 16日 12:36:20

存储卷

存储卷供用户创建工作负载使用，是将工作负载数据持久化的一种资源对象。在 KubeSphere 中创建存储卷之前必须存在相应的存储类型。KubeSphere 推荐使用动态分配方式创建存储卷，即用户在 KubeSphere 控制台发起创建和删除存储卷请求后即可在存储服务端创建和删除存储卷，并可供 KubeSphere 工作负载使用。在 all-in-one 部署方式中，可以使用 Local 存储卷将数据持久化，无需存储服务端支持，但此类型存储卷不支持动态分配方式。如果希望体验 KubeSphere 推荐的动态分配 (Dynamic Provisioning) 方式创建存储卷，我们推荐使用 [青云云平台块存储](#)，平台已集成 [QingCloud-CSI](#) 块存储插件，支持使用青云块存储作为平台的存储服务，免去手动配置存储服务端的繁琐。如果需要手动配置 GlusterFS 或 Ceph RBD，需要准备相应的存储服务端，参考附录中的 [部署 Ceph RBD 存储服务端](#) 或 [部署 GlusterFS 存储服务端](#)。

本节通过以下几个方面介绍如何管理存储卷：

- 创建存储卷
- 查看存储卷
- 使用存储卷
- 删除存储卷
- Local Volume 使用方法
- Ceph RBD 密钥无法挂载解决方案

创建存储卷

首先登录 KubeSphere 控制台，访问左侧菜单栏，在 [资源](#) 菜单下，点击 [存储卷](#) 按钮进入列表页。作为集群管理员，可以查看当前集群下所有的存储卷以及挂载情况。普通用户只能看到自己有权限访问的存储卷。

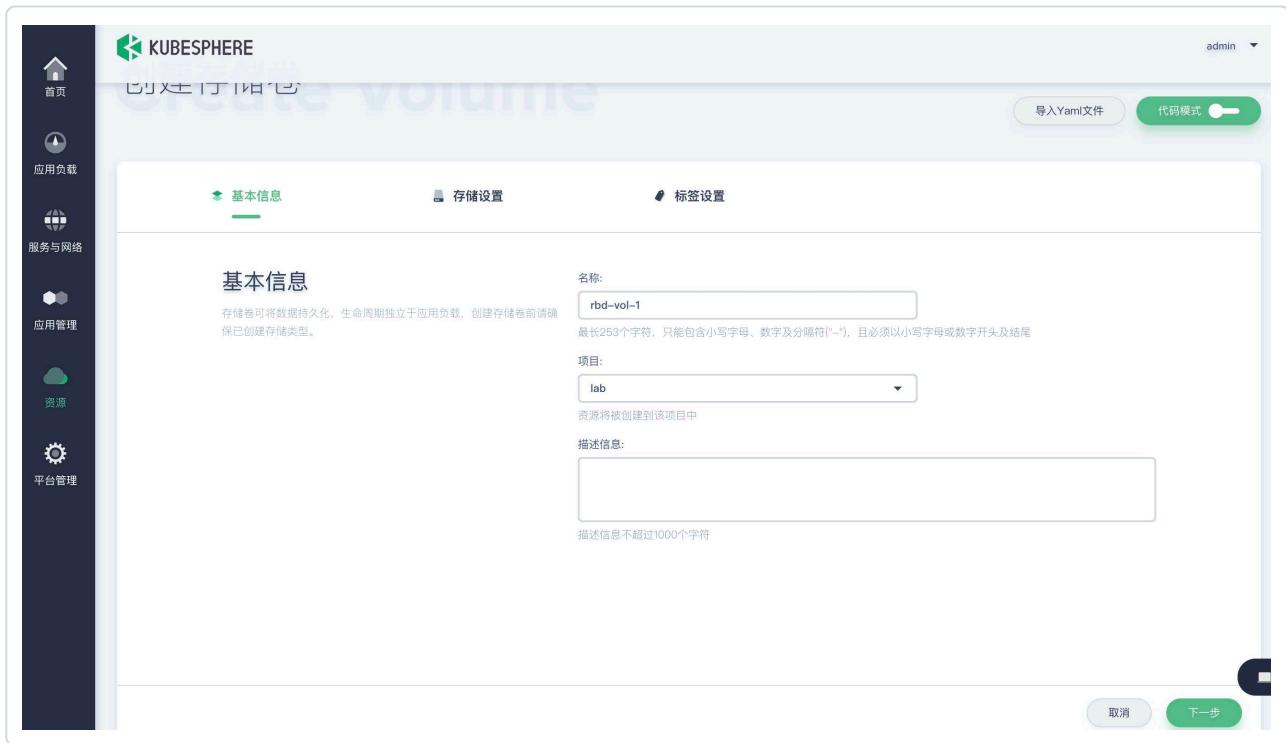
The screenshot shows the KubeSphere storage volumes management interface. On the left is a dark sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area has a header 'KUBESPHERE' and '存储卷' (Storage Volumes). It displays a table with one row of data:

名称	状态	容量	访问模式	挂载状态	项目	创建时间	创建者
rbd-vol-1	Bound	15Gi	RWO	未挂载	default	2018年7月17日 11:45:54	admin

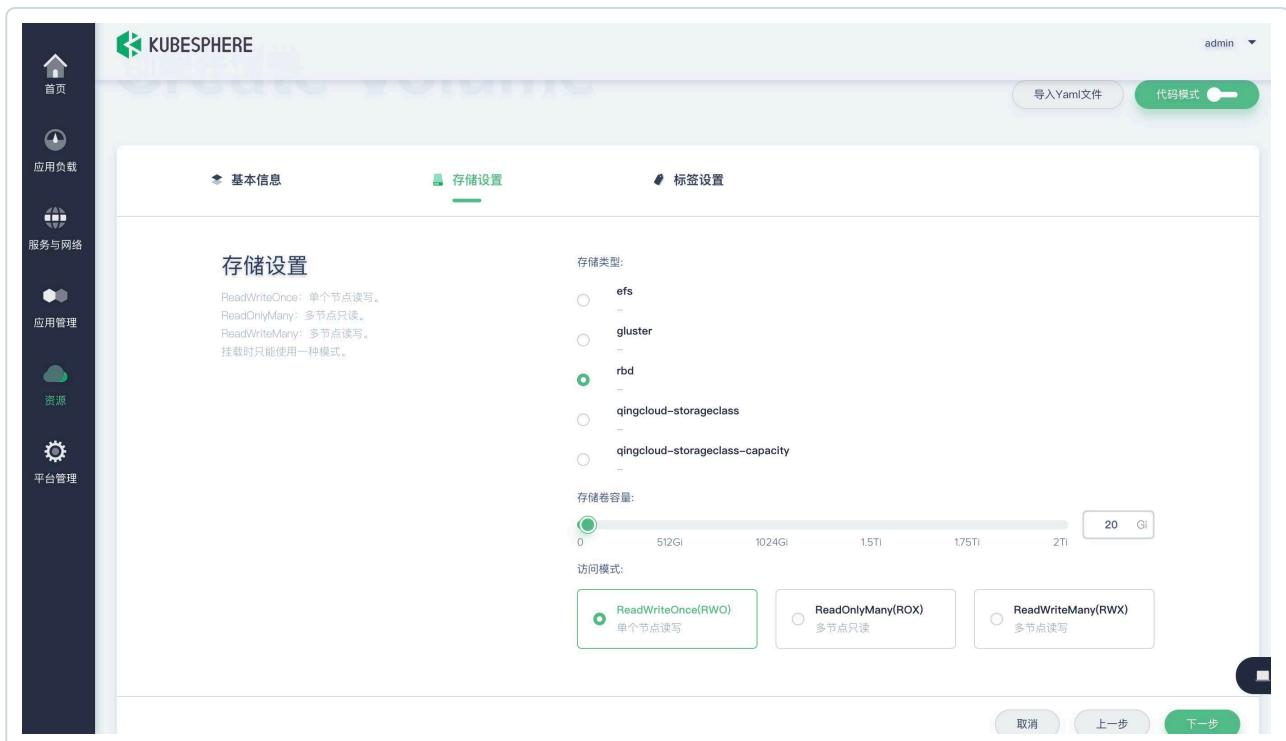
At the top right, there is a search bar with placeholder text '输入查询条件进行过滤' (Filter by input query conditions), a refresh button, and a green '创建存储卷' (Create Storage Volume) button.

注：使用 KubeSphere 创建 Local 存储类型存储卷，需预先创建 Local 类型 PV，参见 [Local Volume 使用方法。](#)

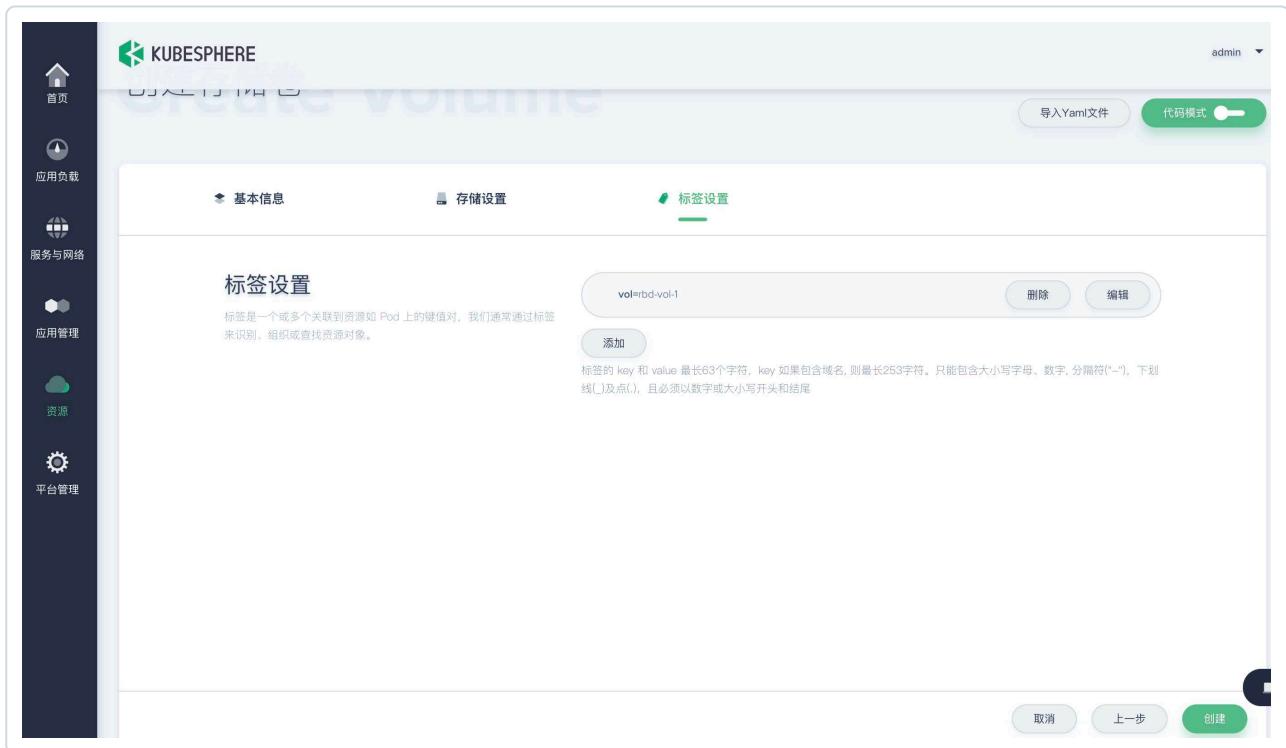
1. 点击存储卷列表页的 **创建存储卷** 按钮进入创建存储卷界面，填写存储卷基本信息，完成后点下一步：



2. 存储设置中，选择存储卷的存储类型，关于如何创建存储类型请参考 [存储类型管理](#)。按需填写存储卷的容量大小，存储卷大小和访问模式必须与存储类型和存储服务端能力相适应，访问模式通常选择为 RWO。各类型存储支持的访问模式参见 [Kubernetes 官方文档](#)。



3. 为存储卷设置标签，可通过标签来识别、组织和查找资源对象。



4. 存储卷创建成功。

存储卷							
共1个存储卷							
名称	状态	容量	访问模式	挂载状态	项目	创建时间	创建者
rbd-vol-1	Bound	15Gi	RWO	未挂载	default	2018年 7月 17日 11:45:54	admin

查看存储卷

点击列表页中的存储卷，即可查看存储卷的详细信息。存储卷详情页 **编辑 Yaml 文件** 和 **删除** 按钮分别可以编辑和删除存储卷；右侧显示挂载容器组的信息。

The screenshot shows the KubeSphere storage volume details page for 'rbd-vol-1'. The top navigation bar includes 'KUBESPHERE' and a user dropdown. The main header is 'rbd-vol-1'. Below it, there are tabs for '资源信息' (Resource Information) and '事件' (Events), with '资源信息' being active. On the left, there's a sidebar with links for '首页' (Home), '应用负载' (Application Load), '服务与网络' (Services & Networks), '应用管理' (Application Management), '资源' (Resources), and '平台管理' (Platform Management). The central content area has sections for '标签' (Labels) and '详情' (Details). The '标签' section shows a single label 'vol=rbd-vol-1'. The '详情' section lists various details about the storage volume, such as Project (default), Status (Bound), Capacity (15Gi), Storage Type (rbd), Provider (kubernetes.io/rbd), Access Mode (ReadWriteOnce), Creation Time (2018年 7月 17日 11:45), and Creator (admin). To the right, there's a table titled '挂载信息' (Mount Information) showing one pod mount: 'nginx-76f58c6b79-ghhwq' (Container ID), '运行中' (Running) status, 'kubesphere' (Host), and '10.10.148.97' (Pod IP).

使用存储卷

在创建工作负载时将用到存储卷，以创建部署并挂载存储卷为例：

注：工作负载挂载 Ceph RBD 类型存储卷时需确保工作负载所在项目必须有特定的 Secret，详情见 [Ceph RBD 无法挂载解决方案](#)。

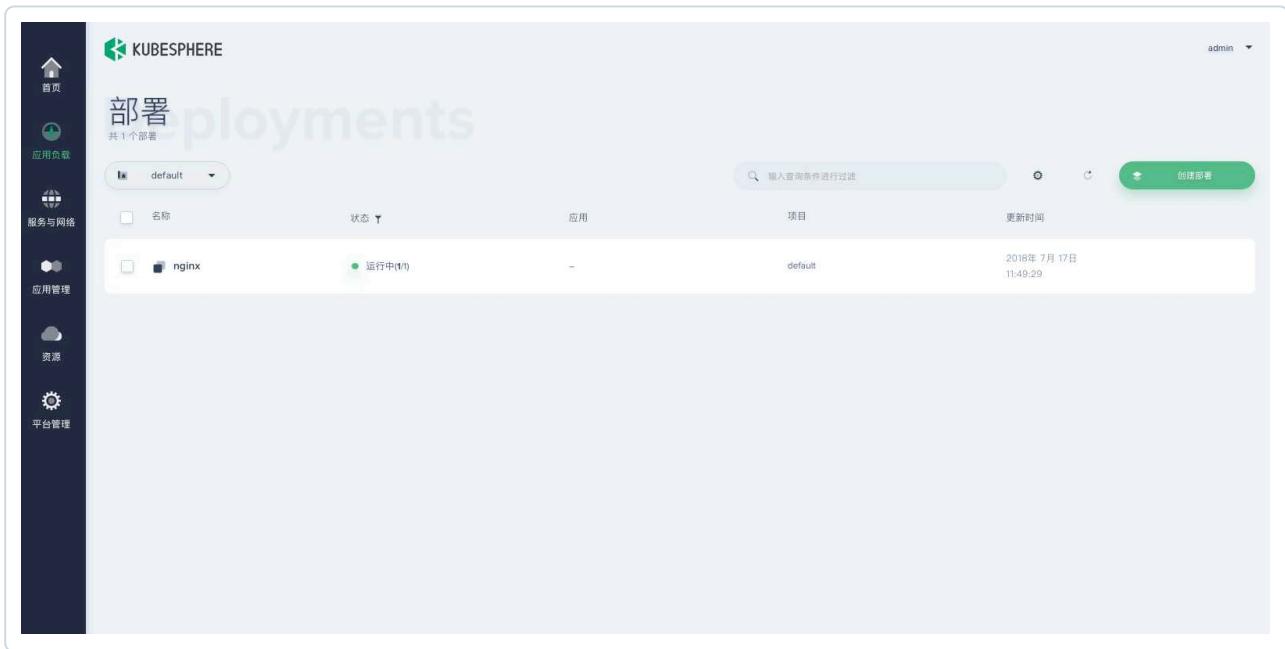
1. 在工作负载中选择创建部署，填写基本信息和容器组设置信息，可参考 [部署管理](#)。存储设置页面可挂载已有的存储卷或创建新的存储卷，在添加存储卷下选择第一项 **存储卷**，然后在右侧选择已经创建的存储卷：

The screenshot shows the KubeSphere Storage Volume Management interface. On the left sidebar, there are several navigation items: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area has tabs at the top: 基本信息 (Basic Information), 容器组设置 (Container Group Settings), 存储设置 (Storage Settings) (which is selected), 标签设置 (Label Settings), and 节点选择器 (Node Selector). Below these tabs, the page title is "存储卷" (Storage Volumes) and the sub-section title is "已有存储卷 新建" (Existing Storage Volumes, Create New). A button labeled "添加存储卷" (Add Storage Volume) is visible. There are three listed storage volumes: 1. "my-pvc": Status: 闲置 (Idle), Capacity: 10Gi, Type: qingcloud-storage... (RWO). 2. "data-mysql-mariadb-slave-0": Status: 已使用 (Used), Capacity: 10Gi, Type: qingcloud-storage... (RWO). 3. "data-mysql-mariadb-master-0": Status: 已使用 (Used), Capacity: 10Gi, Type: qingcloud-storage... (RWO).

2. 输入存储卷在容器内的挂载路径，完成后续步骤即可成功创建部署：

The screenshot shows the KubeSphere Deployment Creation interface. The title bar says "Create Deployment". On the left sidebar, there are several navigation items: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area has tabs at the top: 基本信息 (Basic Information), 容器组设置 (Container Group Settings), 存储设置 (Storage Settings) (which is selected), 标签设置 (Label Settings), and 节点选择器 (Node Selector). Below these tabs, the page title is "存储卷" (Storage Volumes) and the sub-section title is "正在添加存储卷 到容器中" (Adding storage volume to container). A button labeled "添加存储卷" (Add Storage Volume) is visible. There is one listed storage volume: "my-pvc": Status: 已创建 (Created), Capacity: 10Gi, Type: qingcloud-storage... (RWO). To the right of the storage volume list, there are input fields for "容器名" (Container Name) set to "mysql", "访问模式" (Access Mode) set to "读写" (ReadWrite), "挂载路径" (Mount Path) set to "/var/lib/mysql", and a "挂载" (Mount) button. At the bottom right, there are buttons for "取消" (Cancel), "上一步" (Previous Step), and "下一步" (Next Step) (which is highlighted in green).

3. 当存储卷挂载成功后，部署正常启动，说明此时存储卷已成功挂载：



删除存储卷

注：使用 KubeSphere 删除 Local 存储类型存储卷，需手动回收 Local 类型 PV，参见 [Local Volume 使用方法。](#)

在存储卷列表页可选中存储卷，点击删除按钮删除，删除存储卷前请确保存储卷挂载状态处于未挂载。

名称	状态	容量	访问模式	挂载状态	项目	创建时间	创建者
mysql-volume qingcloud-storagecl...	Bound	10Gi	RWO	已挂载	lab	2018年 8月 03日 23:28:47	userdemo
wordpress-volu... qingcloud-storagecl...	Bound	10Gi	RWO	已挂载	lab	2018年 8月 03日 23:25:27	userdemo
mysql qingcloud-storagecl...	Bound	10Gi	RWO	已挂载	demo-case	2018年 8月 01日 22:47:43	admin
xz qingcloud-storagecl...	Bound	10Gi	RWO	已挂载	xz22	2018年 8月 01日 17:42:09	admin
wp-pv-claim qingcloud-storagecl...	Bound	10Gi	RWO	已挂载	xz22	2018年 8月 01日 17:39:31	admin
stor-mysql-test-0 qingcloud-storagecl...	Bound	10Gi	RWO	未挂载	ucase01	2018年 8月 01日 15:56:38	-
test qingcloud-storagecl...	Bound	10Gi	RWO	未挂载	test1	2018年 8月 01日 15:51:52	yinfengliao
mysql-pv-u-mys... qingcloud-storagecl...	Bound	10Gi	RWO	未挂载		2018年 8月 01日	

Local Volume 使用方法

Local Volume 仅用于 all-in-one 单节点部署。

1. 创建 Local Volume 的存储类型详细步骤如下：

- 1.1. 通过 `sc.yaml` 文件定义 Local Volume 的存储类型：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: local
provisioner: kubernetes.io/no-provisioner
volumeBindingMode: WaitForFirstConsumer
```

- 1.2. 执行创建命令：

```
$ kubectl create -f sc.yaml
```

2. 创建 Local Volume 文件夹:

- 登录宿主机，创建文件夹，以文件夹 `vol-test` 为例，执行以下命令：

```
sudo mkdir -p /mnt/disks/vol-test
```

3. 创建 Local PV:

- 3.1. 通过 `pv.yaml` 文件定义 Local PV:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-local
spec:
  capacity:
    storage: 10Gi
  # volumeMode field requires BlockVolume Alpha feature gate to be enabled.
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Delete
  storageClassName: local
  local:
    path: /mnt/disks/vol-test
  nodeAffinity:
    required:
      nodeSelectorTerms:
        - matchExpressions:
          - key: node-role.kubernetes.io/master
            operator: Exists
```

- 3.2. 执行创建命令：

```
$ kubectl create -f pv.yaml
```

4. 执行以下命令验证创建结果：

```
$ kubectl get pv
NAME      CAPACITY   ACCESS MODES  RECLAIM POLICY  STATUS    CLAIM      STORED
pv-local   10Gi       RWO          Delete        Available  local      4s
```

上述工作完成后可在 KubeSphere 控制台创建存储卷，KubeSphere 控制台创建的存储卷容量不可大于预分配 PV 容量。

注：Local Volume 存储卷创建成功后为 Pending 属于正常状态，当创建工作负载调度 Pod 后存储卷状态即可变化为 Bound。

删除 Local Volume PV 和文件夹

1. 删除 Local Volume PV：

```
$ kubectl delete pv pv-local
```

2. 删除 Local Volume 文件夹，此操作也会删除 vol-test 文件夹里内容：

```
$ sudo cd /mnt/disks
$ sudo rm -rf vol-test
```

Ceph RBD 无法挂载解决方案

Ceph RBD 存储卷操作过程中，如果遇到 Ceph RBD 存储卷挂载至工作负载时因缺少密钥无法挂载，可参考如下解决方案：

1. 假设工作负载所在项目为 ns1，Ceph RBD 存储卷关联的存储类型为 rbd。
2. 通过 Kubectl 命令行工具向 kubernetes 发送以下命令，查询要创建 Secret 名称，得到应创建的 Secret 名称为 ceph-rbd-user-secret。

```
$ kubectl get sc rbd | grep userSecretName  
userSecretName: ceph-rbd-user-secret
```

3. 在 Ceph 集群执行以下命令，得到密钥：

```
$ ceph auth get-key client.admin  
AQAnwihbXo+uDxAAD0HmWziVgTaAdai90IzZ6Q==
```

则密钥为 `AQAnwihbXo+uDxAAD0HmWziVgTaAdai90IzZ6Q==`。

4. 通过 Kubectl 命令行工具执行以下命令，创建 Secret：

```
kubectl create secret generic ceph-rbd-user-secret --type="kubernetes.io/rbd" --from-literal
```

其中，以下的三个字段根据不同环境的实际状况会有所不同，应根据您的环境替换成对应的字段：

- `ceph-rbd-user-secret`
- `AQAnwihbXo+uDxAAD0HmWziVgTaAdai90IzZ6Q==`
- `ns1`

存储类型

存储类型（StorageClass）是由 [集群管理员](#) 配置存储服务端参数，并按类型提供存储给集群用户使用。KubeSphere 目前支持 [Ceph RBD](#) 和 [GlusterFS](#) 存储类型以及青云研发的块存储插件 [QingCloud CSI](#)（更多的存储类型持续更新中），并且支持各种存储类型的展示。

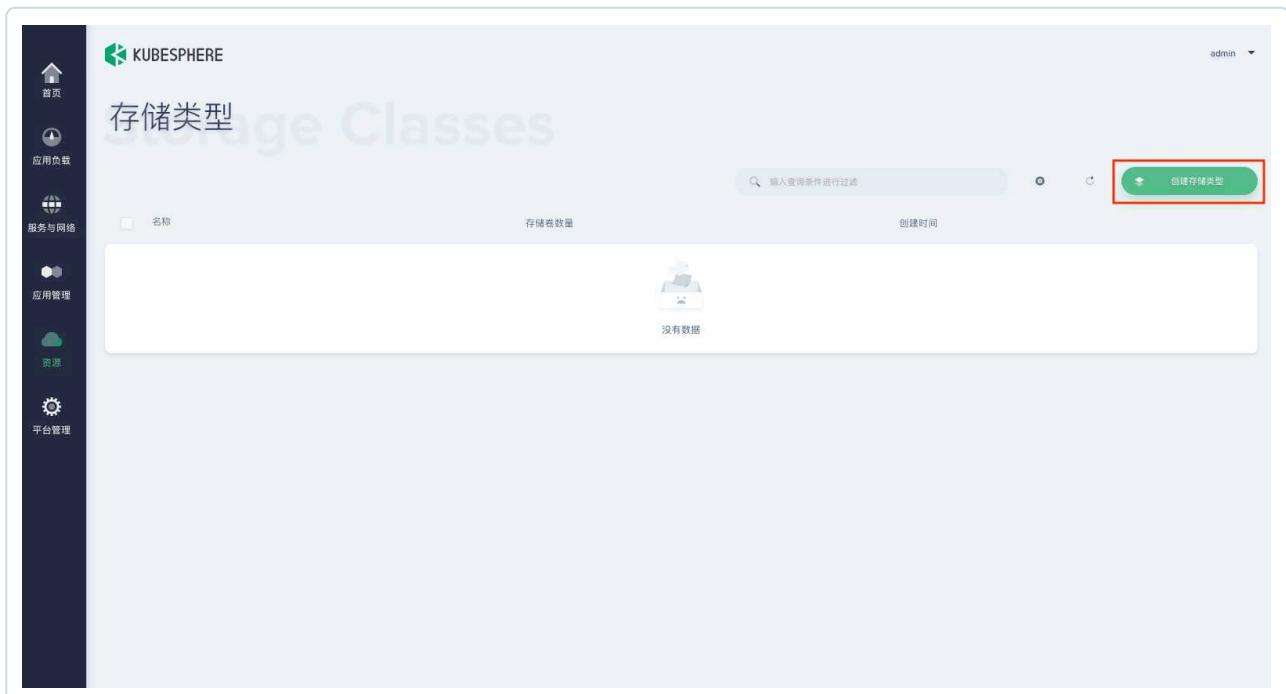
本节通过以下几个方面介绍如何管理存储类型：

- 创建存储类型
- 查看存储类型
- 设置默认存储类型
- 删除存储类型

创建存储类型

首先登录 KubeSphere 管理控制台，访问左侧菜单栏，在 [资源](#) 菜单下，点击 [存储类型](#) 按钮，进入存储类型列表页面。作为集群管理员，可以查看当前集群下所有的存储类型和详细信息。

1. 在存储类型列表页，点击 [创建存储类型](#) 按钮：

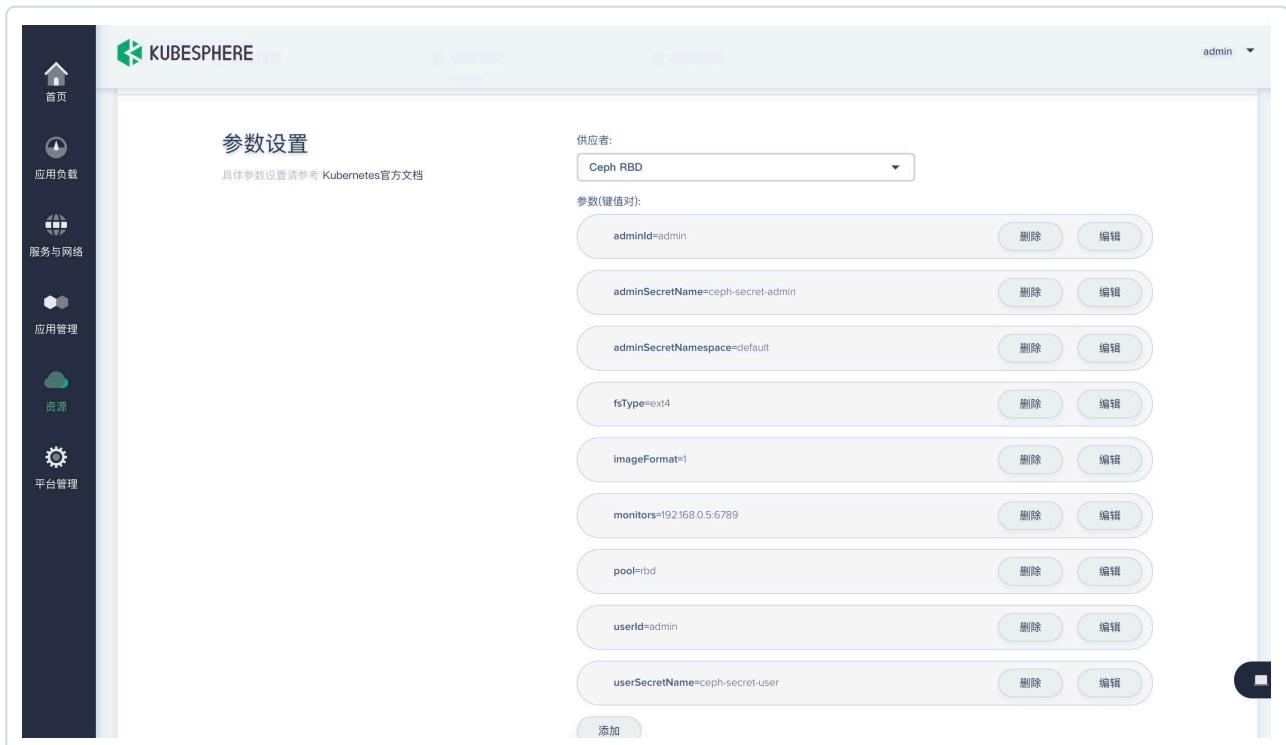


2. 在基本信息页面填入参数，以创建 rbd-sc 存储类型为例。

- 2.1. 填入存储类型名和描述信息：



- 2.2. 填入参数键值对，可参考 [Kubernetes 官方文档](#)。



- 2.3. 为存储类型设置标签，点击创建：



查看存储类型

1. 存储类型列表页显示已经创建的存储类型，点击可查看和编辑该存储类型。

2. 进入存储类型详情页。

选择 **编辑 Yaml 文件** 按钮直接编辑存储类型对象文件；选择 **删除** 按钮删除存储类型；存储卷列表展示与此存储类型相关的存储卷，存储卷介绍请见 [存储卷](#)。



The screenshot shows the KubeSphere storage type details page for the RBD storage type. The left sidebar includes links for Home, Application Load Balancer, Services & Networks, Application Management, Resources (highlighted), and Platform Management. The main content area has a header 'KUBESPHERE' and '存储类型 / rbd'. Below this, it shows the 'rbd' storage type icon and the '资源信息' tab selected. A table lists storage volumes, with one entry: 'rbd-vol-1' (Status: Bound, Capacity: 15Gi, Access Mode: RWO, Mount Status: 已挂载, Project: default, Created Time: 2018年 7月 17日 11:45:54, Creator: admin). There are also '编辑Yaml文件' and '删除' buttons.

设置默认存储类型

一个 Kubernetes 集群中仅允许设置一个默认存储类型。

1. 存储类型列表页点击存储类型名称，进入存储类型详情页。点击编辑 Yaml 文件按钮。

The screenshot shows the KubeSphere interface for managing storage resources. On the left, a sidebar lists various management categories: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The '资源' (Resources) category is currently selected and highlighted in blue.

In the main content area, the title is 'KUBESPHERE' with a logo. Below it, the path is '存储类型 / rbd'. A sub-section titled 'rbd' is shown, featuring a cylinder icon. A red box highlights the '编辑Yaml文件' (Edit YAML file) button, which is located next to a '新增' (Add) button. To the right, a table header for '资源信息' (Resource Information) is displayed with columns: 名称 (Name), 状态 (Status), 容量 (Capacity), 访问模式 (Access Mode), 挂载状态 (Mount Status), 项目 (Project), 创建时间 (Creation Time), and 创建者 (Creator).

The table body is titled '存储卷' (Storage Volumes) and contains one entry:

	名称	状态	容量	访问模式	挂载状态	项目	创建时间	创建者
	无数据	无	无	无	无	无	无	无

A search bar at the top right allows users to filter results by inputting search terms.

2. 进入 Yaml 文件编辑界面，添加 annotation，并更新。

```
1 kind: StorageClass
2 apiVersion: storage.k8s.io/v1
3 metadata:
4   name: rbd
5   selfLink: /apis/storage.k8s.io/v1/storageclasses/rbd
6   uid: 18b3d293-8973-11e8-9c40-5254d9c0c0d8
7   resourceVersion: '8108'
8   creationTimestamp: '2018-07-17T03:40:06Z'
9   labels:
10    sc: rbd
11   annotations:
12     creator: admin
13     storageclass.beta.kubernetes.io/is-default-class: 'true'
14   provisioner: kubernetes.io/rbd
15   parameters:
16     adminId: admin
17     adminSecretName: ceph-rbd-admin-secret
18     adminSecretNamespace: kube-system
19     fsType: ext4
20     monitors: '192.168.0.5:6789'
21     pool: rbd
22     userId: admin
23     userSecretName: ceph-rbd-user-secret
24   reclaimPolicy: Delete
25
```

3. 存储类型详情页中，存储类型名称后添加五角星标识为默认存储类型。

The screenshot shows the KubeSphere Storage Classes management interface. On the left is a dark sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area has a light gray header with the KubeSphere logo and the text "Storage Classes". Below the header, it says "共 1 个存储类型" (1 storage class). There is a search bar with placeholder text "输入查询条件进行过滤" (Filter by input query conditions) and a green button labeled "创建存储类型" (Create storage type). A table lists the storage classes:

名称	存储卷数量	创建时间
rbd *	0	2018年 7月 17日 11:40:06

删除存储类型

- 存储类型删除前需确保当前集群无此存储类型创建的存储卷。

The screenshot shows the KubeSphere Storage Classes management interface. The sidebar and header are identical to the previous screenshot. The main area now displays "共 5 个存储类型" (5 storage classes). A red button labeled "Delete Storage Class" is visible. A green button labeled "取消选择" (Cancel Selection) is also present. The table lists the storage classes:

名称	存储卷数量	创建时间
efs	0	2018年 7月 19日 15:28:15
gluster	7	2018年 6月 20日 16:31:42
rbd	18	2018年 6月 20日 10:22:03
qingcloud-storageclass *	42	2018年 6月 12日 18:21:15
qingcloud-storageclass-capacity	0	2018年 6月 12日 18:21:15

主机管理

Kubernetes 集群中的计算能力由主机提供，Kubernetes 集群中的 Node 是所有 Pod 运行所在的工作主机，可以是物理机也可以是虚拟机。

本节通过以下几个方面介绍如何管理主机：

- Taints 管理
- 查看主机详情
- 停用/启用主机
- 更新主机的标签

Taints 管理

首先登录 KubeSphere 管理控制台，访问左侧菜单栏，在 **资源** 菜单下，点击 **主机管理** 按钮进入列表页。作为集群管理员，可以查看当前集群下所有主机信息。

名称	状态	角色	CPU(Core)	内存(GiB)	容器组	创建时间
i-3gr5u2pa 192.168.0.41	运行中	node	0.6/8.0	5.8/7.7	45/60	2018年 6月 12日 06:21:21
i-78nzdqal 192.168.0.39	运行中	master	0.6/4.0	2.2/3.8	14/60	2018年 6月 12日 06:20:54
i-7liraqs 192.168.0.40	运行中	node	1.1/8.0	4.6/7.7	53/60	2018年 6月 12日 06:21:21
i-dhpvdilm 192.168.0.9	运行中	node	0.2/8.0	3.4/7.7	17/60	2018年 8月 01日 04:37:08
i-ohb602zs 192.168.0.3	运行中	node	0.1/8.0	2.9/7.7	18/60	2018年 8月 01日 04:37:07
i-xdo05x6f 192.168.0.42	运行中	node	1.1/8.0	5.7/7.7	47/60	2018年 6月 12日 06:21:21

1. 点击右上角 **Taints 管理** 按钮，进入集群主机 Taints 统一管理页面。

2. 在左侧节点列表中可以看到各个主机节点已有的 Taints 状态，选中相应节点，在右侧可以对其进行完成 Taints 的添加、删除、更新操作。

节点名	CPU 使用	内存使用	Pod 使用	Taint
i-3ejm3bfq	0.5/4.0 核	3.3/3.8 GiB	6/100	dedicated=log:NoSchedule
i-87s3jhh	0.5/4.0 核	3.1/3.8 GiB	4/100	dedicated=log:NoSchedule
i-muws35zt	0.4/4.0 核	2.5/3.8 GiB	13/100	node-role.kubernetes.io/master=:NoSchedule
i-qipv7cif	0.2/4.0 核	3.6/7.7 GiB	28/100	-
i-rylgnorp	0.6/4.0 核	4.1/7.7 GiB	46/100	-

参数解释:

- NoSchedule: 表示不允许调度，已调度的资源不受影响。
- PreferNoSchedule: 表示尽量不调度。
- NoExecute: 表示不允许调度，已调度的在 tolerationSeconds (定义在 Toleration 上) 后删除 Node 和 Pod 上都可以定义多个 Taints 和 Toleration，Scheduler 会根据具体定义进行筛选，Node 筛选 Pod 列表的时候，会保留 Toleration 定义匹配的，过滤掉没有 Toleration 定义的，过滤的过程是这样的：
 - 如果 Node 中不存在影响策略为 NoSchedule 的 Taint，但是存在一个或多个影响策略为 PreferNoSchedule 的 Taint，该 Pod 会尽量不调度到该 Node。
 - 如果 Node 中存在一个或多个影响策略为 NoExecute 的 Taint，该 Pod 不会被调度到该 Node，并且会驱逐已经调度到该 Node 的 Pod 实例。

3. 当对所需一个或者多个主机节点完成相应 Taints 编辑操作后，点击 应用 按钮，完成 Taints 配置的更新。

查看主机详情

在主机列表页，点击某个主机节点打开其详情页，可以看到当前主机下所有资源的概况，点击相应资源，可以打开对应资源的详情页面查看更多细节信息。

The screenshot shows the KubeSphere interface for viewing a node detail. On the left is a sidebar with navigation links: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area has a header with the node name 'i-3ejm3bfq' and a back button. It includes tabs for 概览 (Overview), 资源信息 (Resource Information), and 事件 (Events), with '概览' being active. Below the tabs are sections for 资源状态 (Resource Status) showing CPU (12%), 内存 (86%), and Pod (6%) usage, and Taints (dedicated log NoSchedule). A table titled 'Conditions' lists two entries: OutOfDisk (False) and MemoryPressure (False). The bottom section displays basic information about the node, such as state (运行中), IP address (192.168.0.10), and creation time (2018年 6月 23日 07:00).

停用或启用主机

在主机详情页面，点击左侧 停用 (cordon) 按钮，主机状态变为 无法调度，当前按钮变为 启用 (uncordon)，当有新的工作负载被创建时将不会被调度到此节点，如想回复为可调度状态，点击 启用 按钮。

KUBESPHERE

nodes / i-3ejm3bfq

概览 资源信息 事件

资源状态

CPU使用情况: 12% (已用: 0.5 核, 共计: 4.0 核)

内存使用情况: 84% (已用: 3.2 GiB, 共计: 3.8 GiB)

Pod使用情况: 6% (已用: 6, 共计: 100)

标签

beta.kubernetes.io/arch=amd64
beta.kubernetes.io/instance-type=custom
beta.kubernetes.io/os=linux
failure-domain.beta.kubernetes.io/region=ap2a
kubernetes.io/hostname=i-3ejm3bfq
node_id=cin-7n7trif
role=log

基本信息

状态: 无法调度 (red dot)

IP地址: 192.168.0.10

无法调度: true

角色: log

操作系统: linux

内核版本: 4.4.0-12-generic

容器版本: docker://18.3.1

Kubelet 版本: v1.10.4

Kube-Proxy 版本: v1.10.4

Taints

key	value	effect
dedicated	log	NoSchedule
node.kubernetes.io/unschedulable		NoSchedule

Conditions

类型	状态	最近更新	原因	消息
OutOfDisk	False	2018年 6月 23日 07:14:39	KubeletHasSufficientDisk	kubelet has sufficient disk space available

更新主机标签

进入项目详情页面，点击左侧项目操作菜单，点击 **编辑标签** 按钮编辑当前主机上的标签 (Labels)，最后点击 **确认** 按钮完成修改。

The screenshot shows the KubeSphere web interface. On the left is a sidebar with navigation links: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area shows a node details page for 'i-3ejm3bfq'. The node is labeled as 'log'. A modal window titled 'Edit Labels' is open, listing several labels:

- beta.kubernetes.io/arch=amd64
- beta.kubernetes.io/instance-type=custom
- beta.kubernetes.io/os=linux
- failure-domain.beta.kubernetes.io/hostname=i-3ejm3bfq
- role=log
- kubernetes.io/hostname=i-3ejm3bfq
- node_id=c1n-7n7tirf

At the bottom of the modal are '取消' (Cancel) and '确定' (Confirm) buttons. Below the modal, there are status indicators for MemoryPressure, Disk, and KubeletHasSufficientMemory.

节点信息

状态: 运行中

IP地址: 192.168.0.10

无法调度: false

角色: log

操作系统: linux

内核版本: 4.4.0-127-generic

容器版本: docker://18.03.1~ce~3-stretch

Kubelet 版本: v1.10.4

Kube-Proxy 版本: v1.10.4

创建时间: 2018年 6月 23日 07:01

标签

beta.kubernetes.io/arch=amd64
beta.kubernetes.io/instance-type=custom
beta.kubernetes.io/os=linux
failure-domain.beta.kubernetes.io/hostname=i-3ejm3bfq
kubernetes.io/hostname=i-3ejm3bfq
node_id=c1n-7n7tirf
role=log

基本信息

MemoryPressure: False

2018年 6月 23日 07:00:48

KubeletHasSufficientMemory

Pod 使用情况: 6% 已用: 6 共计: 100%

消息: kubelet has sufficient disk available

kubelet has sufficient memory available

项目管理

KubeSphere 中的项目对应的是 Kubernetes 的 namespace，是对一组资源和对象的抽象集合，常用来隔离不同的用户。

本节通过以下几个方面介绍如何管理项目：

- 创建项目
- 查看项目详情
- 管理项目成员
- 创建成员角色
- 删除项目

创建项目

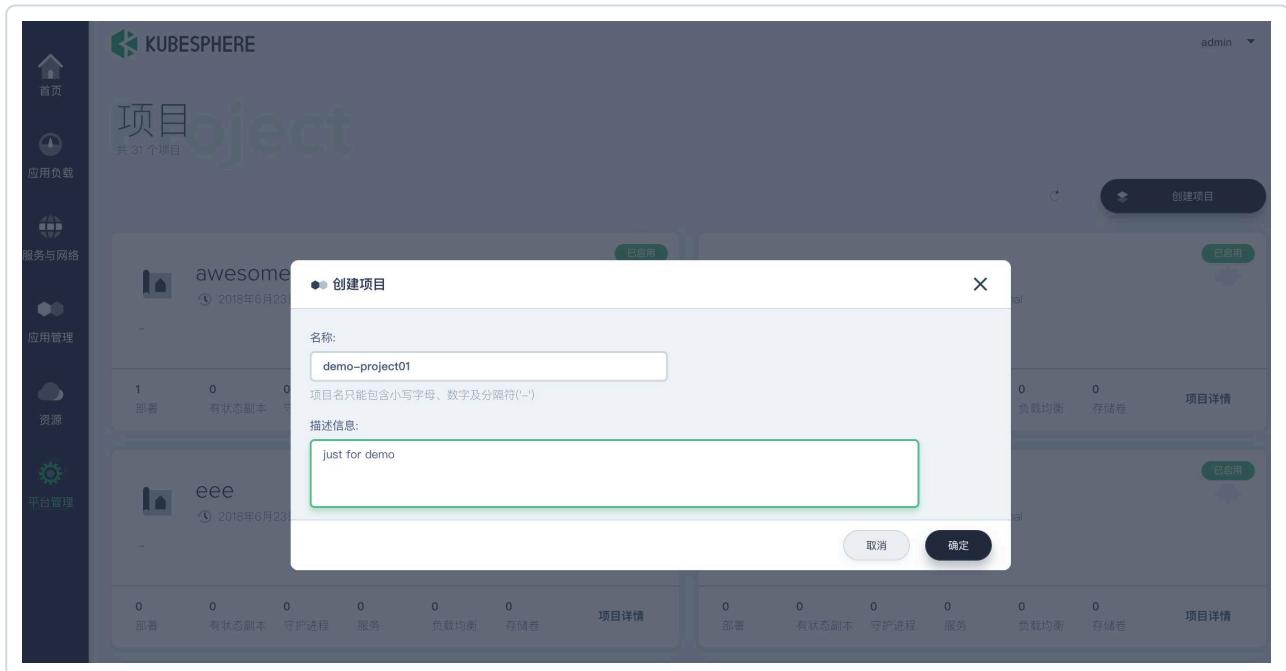
首先登录 KubeSphere 管理控制台，访问左侧菜单栏，在 **平台管理** 菜单下，点击 **项目管理** 按钮进入列表页。作为集群管理员，可以查看当前集群下所有项目，普通用户只能看到自己能访问的项目。

1. 点击右上角 **创建项目** 按钮，新建一个项目，项目之间的资源是相互隔离的。

The screenshot shows the KubeSphere management interface with the 'Project Management' section selected. On the left is a sidebar with icons for Home, Application Load, Services & Networks, Application Management, Resources, and Platform Management. The main area displays a grid of project cards. Each card contains the project name, creation date, owner, status (e.g., '已启用'), and resource counts (Deployments, StatefulSets, Services, Ingresses, Volumes). A 'Create Project' button is located in the top right corner of the main area.

项目名称	创建时间	所有者	状态	资源
zyrs	2018年 8月 03日 17:49	zyrs	已启用	0 部署, 0 有状态副本集, 0 守护进程集, 0 服务, 0 应用路由, 0 存储卷
lab	2018年 8月 02日 07:35	userdemo	已启用	2 部署, 0 有状态副本集, 0 守护进程集, 2 服务, 1 应用路由, 2 存储卷
demo-case	2018年 8月 01日 21:17	admin	已启用	1 部署, 0 有状态副本集, 0 守护进程集, 2 服务, 1 应用路由, 1 存储卷
demosadfasfsafas	2018年 8月 01日 17:43	xinyi	已启用	1 部署, 0 有状态副本集, 0 守护进程集, 0 服务, 0 应用路由, 0 存储卷
ucase01	2018年 7月 31日 18:21	admin	已启用	-
demo001	2018年 7月 26日 14:26	flora	已启用	-

2. 在弹出窗口填入项目基本信息并点击确定按钮。



3. 项目创建后，可以进入详情页对项目进行管理，可参考后续步骤。

查看项目详情

在项目列表页，点击某个项目的 **项目详情** 按钮，打开项目详情页，可以看到当前项目下所有资源的概况。

管理项目成员

项目创建之后可以通过邮箱地址邀请其他成员参与到项目的协作中，请合理的为项目成员分配角色。

1. 项目创建之后，进入项目详情页面，点击左侧项目操作菜单，点击 **成员** 按钮，进入成员管理页面。
2. 输入邮箱搜索用户，选中用户并分配合适的成员角色。注：成员角色仅作用于所属项目，请区分集群角色。默认的成员角色有：

- admin，有权管理项目内包括成员和成员角色在内的所有资源；
- editor，有权管理项目内除成员和成员角色之外的所有资源；

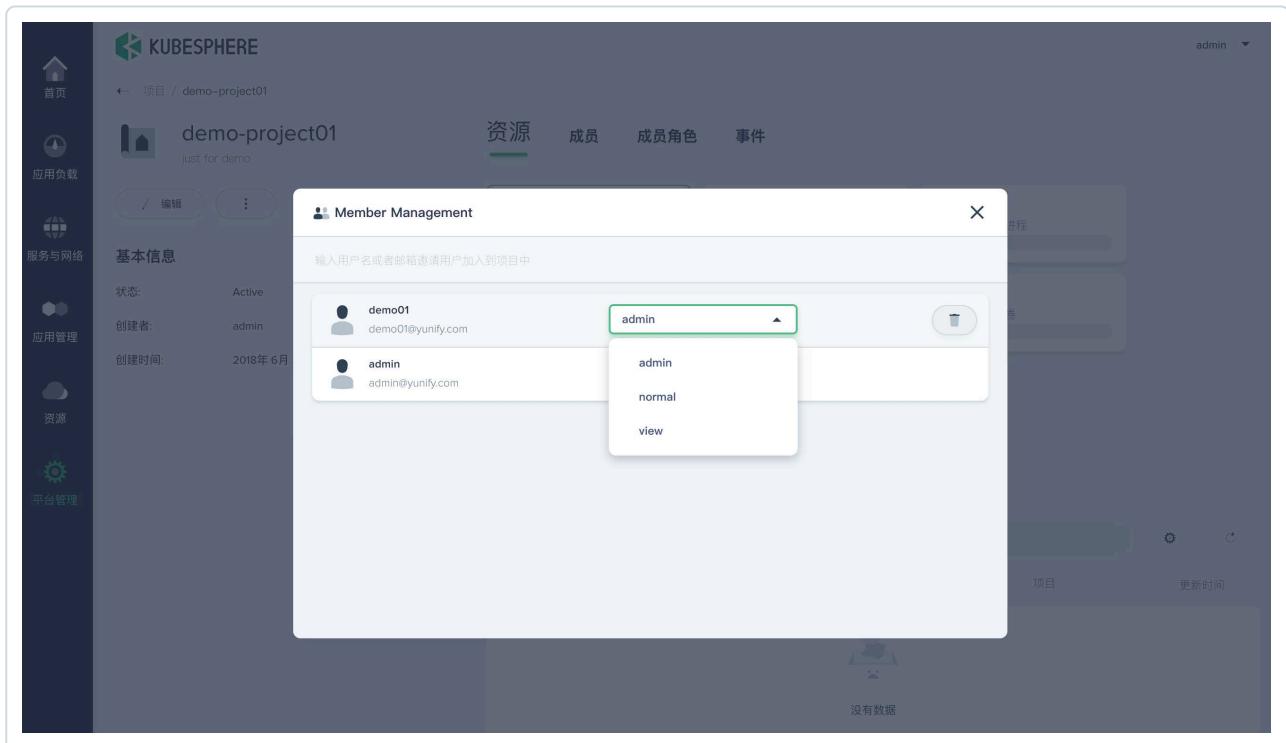
- view, 仅有权查看项目内的应用负载、服务和应用路由。

The screenshot shows the KubeSphere interface for the 'demo-project01'. On the left sidebar, there are several navigation items: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area displays the project details for 'demo-project01', which is described as 'just for demo'. Below the project name are buttons for 编辑 (Edit), 三个点 (More), 成员 (Members), 成员角色 (Member Roles), and 删除 (Delete). A red arrow points to the '成员' button. To the right, there are tabs for 资源 (Resources), 成员 (Members), 成员角色 (Member Roles), and 事件 (Events). Under the 资源 tab, there are six resource cards: 部署 (Deployments) with 0 items, 有状态副本 (Stateful Sets) with 0 items, 守护进程 (DaemonSets) with 0 items, 服务 (Services) with 0 items, 应用路由 (Ingresses) with 0 items, and 存储卷 (Storage Volumes) with 0 items. Below these cards is a 'Pods' section with one card showing 0 items. At the bottom, there is a search bar with the placeholder '输入查询条件进行过滤' (Filter by query conditions) and columns for 名称 (Name), 状态 (Status), 应用 (Application), 项目 (Project), and 更新时间 (Last updated).

- 输入需要添加的成员名称，如存在并检索发现后，点击加号添加。

The screenshot shows the 'Member Management' dialog box over the KubeSphere interface. The dialog has a header 'Member Management' with a close button 'X'. It contains a list of members: 'demo01' (with email 'demo01@yunify.com'). A green '+' button is located at the bottom right of the dialog. A red arrow points to this '+' button. The background shows the same project details and resource cards as the previous screenshot.

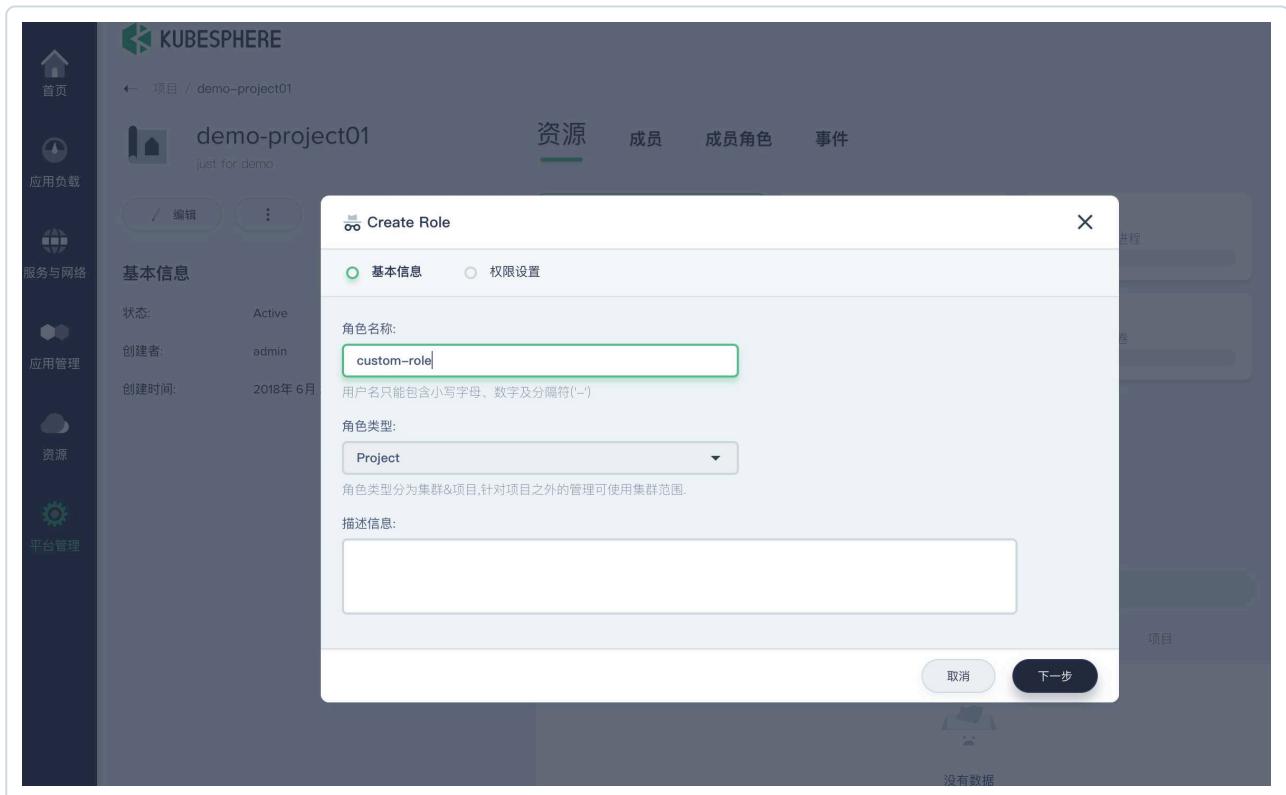
- 为新添加的成员选择合适的成员角色，选中后会立即生效。



创建成员角色

如果项目内预置的成员角色无法满足特定权限限制的要求，则需创建自定义的成员角色。

1. 进入项目详情页面，点击左侧项目操作菜单，点击 **成员角色** 按钮，进入角色创建页面。



2. 输入角色名称后点击下一步，依次勾选权限规则，点击创建。

成员角色只在当前项目下生效，是为项目成员赋予权限的一种方式，即此角色授权的覆盖范围为当前项目下的资源和操作，请区别于通过角色管理创建的集群角色。

The screenshot shows the 'Create Role' dialog box in KubeSphere. The title bar says 'Create Role'. There are two tabs: '基本信息' (Basic Information) and '权限设置' (Permission Settings), with '权限设置' selected. The main area contains four sections: '项目管理' (Project Management), '部署管理' (Deployment Management), '有状态副本集' (Stateful Set), and '守护进程集' (DaemonSet). Each section has several checkboxes for various operations like '查看' (View), '创建' (Create), '编辑' (Edit), and '删除' (Delete). At the bottom right of the dialog are three buttons: '取消' (Cancel), '上一步' (Previous Step), and '创建' (Create).

删除项目

进入项目详情页面，点击左侧项目操作菜单，点击 **删除** 按钮删除项目。项目在删除之后，项目下的资源也会被释放掉。

The screenshot shows the 'demo-project01' project details page. On the left, there is a sidebar with icons for Home, Application Load, Services & Networks, Application Management, Resources, and Platform Management. The main content area has tabs for '资源' (Resources), '成员' (Members), '成员角色' (Member Roles), and '事件' (Events). The '资源' tab is selected. It displays resource counts: 0 Deployments, 0 Stateful Sets, 0 DaemonSets, 0 Services, 0 Application Routes, 0 Storage Volumes, and 0 Pods. Below these counts is a '部署' (Deployment) section with a search bar and a button to '输入查询条件进行过滤' (Input filtering conditions). A red arrow points to the '删除' (Delete) button in the '成员' (Members) section of the sidebar.

角色管理

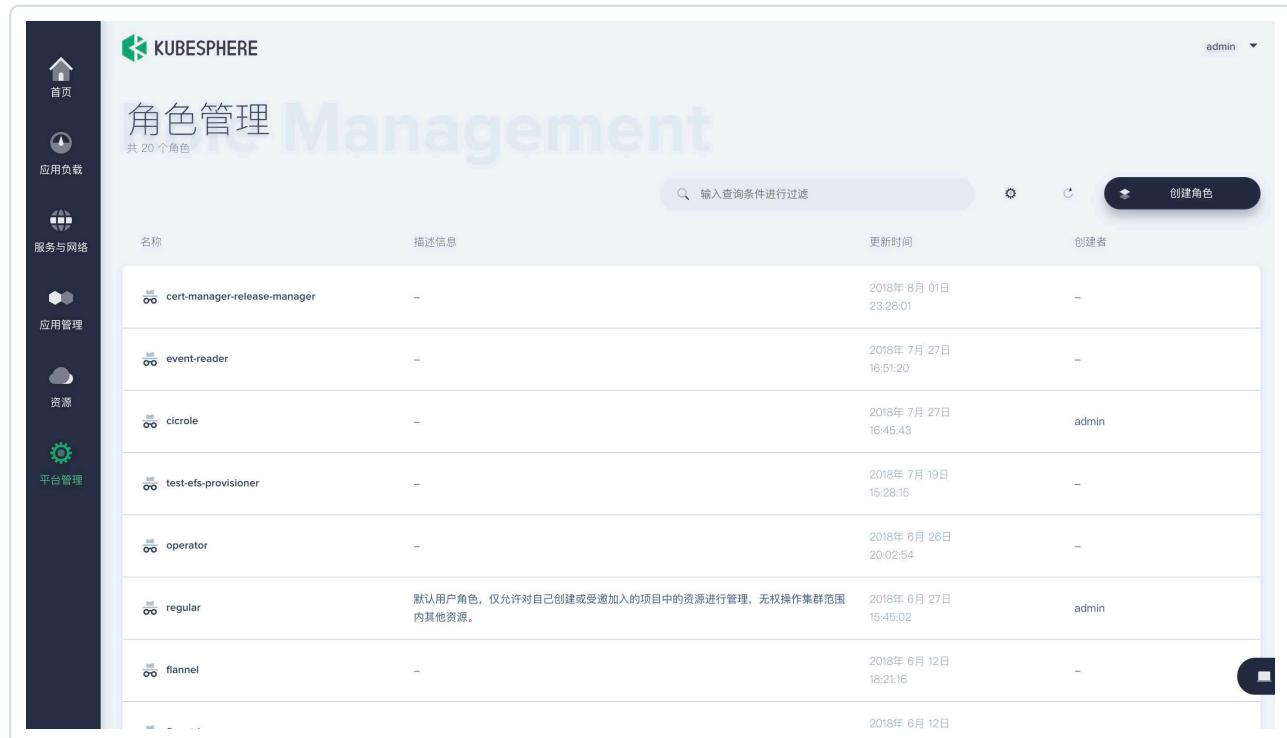
用户的权限管理依赖角色定义，角色标识了用户的身份，定义了用户和可访问/操作的资源之间的关系。当 KubeSphere 预置角色不满足使用要求的时候，可以根据实际情况，为用户创建自定义角色。

本节通过以下几个方面介绍如何管理角色：

- 创建角色
- 查看角色详情
- 修改角色权限
- 删除角色

创建角色

首先登录 KubeSphere 管理控制台，访问左侧菜单栏，在 **平台管理** 菜单下，点击 **角色管理** 按钮进入列表页。作为集群管理员，可以里查看当前集群下所有角色信息。



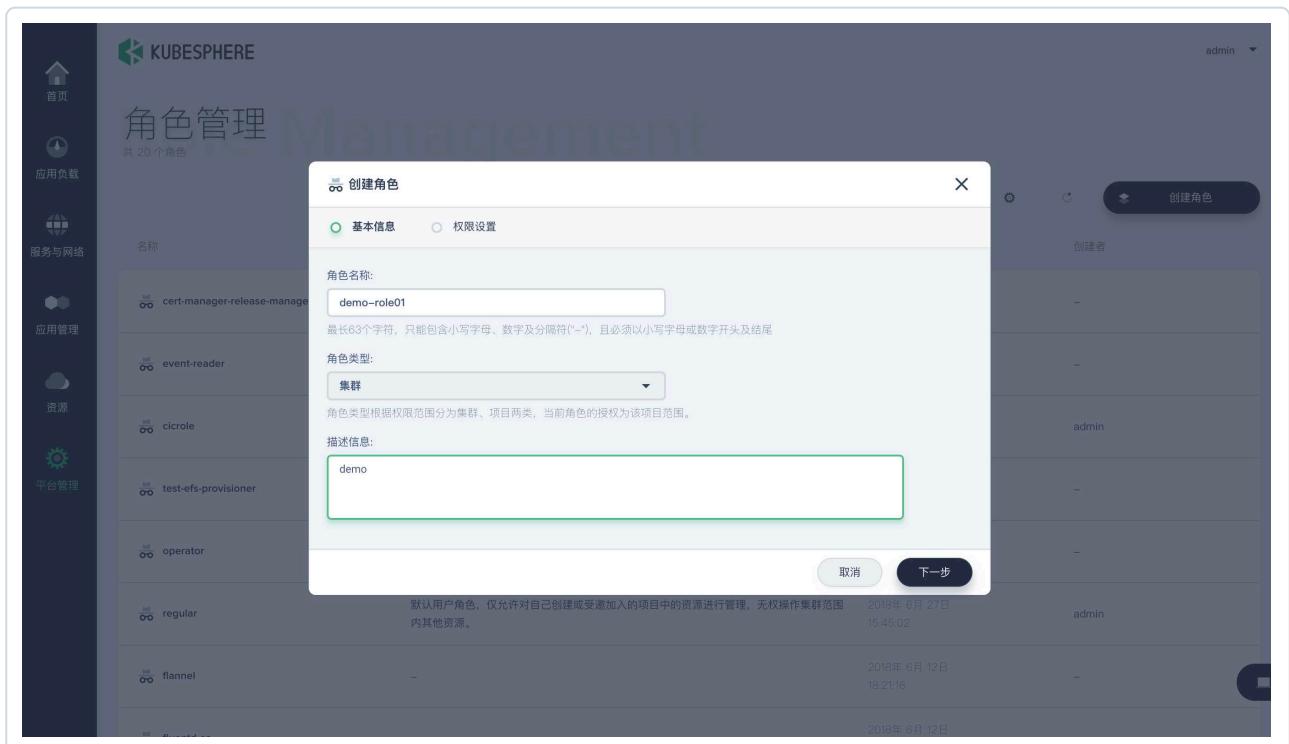
The screenshot shows the KubeSphere Management Console with the 'Role Management' page open. The left sidebar has 'Platform Management' selected, with 'Role Management' highlighted. The main area has a title 'Role Management' with a subtitle '共 20 个角色'. There's a search bar and filter buttons. A table lists 20 roles, each with a preview icon, name, description, last updated time, and creator. The 'regular' role is expanded to show its description: '默认用户角色，仅允许对自己创建或受邀加入的项目中的资源进行管理。无权操作集群范围内其他资源。' The 'operator' role is also partially visible.

名称	描述信息	更新时间	创建者
cert-manager-release-manager	-	2018年 8月 01日 23:28:01	-
event-reader	-	2018年 7月 27日 16:51:20	-
cicrole	-	2018年 7月 27日 16:45:43	admin
test-efs-provisioner	-	2018年 7月 19日 15:28:15	-
operator	-	2018年 6月 26日 20:02:54	-
regular	默认用户角色，仅允许对自己创建或受邀加入的项目中的资源进行管理。无权操作集群范围内其他资源。	2018年 6月 27日 15:45:02	admin
flannel	-	2018年 6月 12日 18:21:16	-

1. 角色类型根据权限范围分为集群、项目两类，分别是集群用户角色和项目成员角色。集群用户角色的管理入口在平台管理下，项目成员角色的管理入口在项目下。

2. 点击右上角创建 **创建角色** 按钮创建角色。用户角色的授权范围为整个集群，请区分于项目成员角色。
3. 在弹出的角色详细信息页面，填写基本的角色信息之后点击下一步。

注：被创建的角色权限范围 <= 创建者的权限范围



- 依次勾选所需权限规则，点击创建：



查看角色详情

在角色列表页，点击某个角色，打开角色详情页，可以看到当前角色授权的相关资源和操作的详细信息以及与它相关联的所有用户。

修改角色权限

进入角色详情页面，点击左侧 **编辑** 按钮角色相关信息以及授权资源和操作。

The screenshot shows the 'Edit Role' dialog in the KubeSphere interface. The dialog title is 'Edit Role' with a close button 'X'. It has two tabs: '基本信息' (Basic Information) and '权限设置' (Permission Settings), with '权限设置' selected. The main area is divided into four sections: '项目管理' (Project Management), '用户管理' (User Management), '角色管理' (Role Management), and '镜像管理' (Image Management). Each section contains four checkboxes for '查看' (View), '创建' (Create), '编辑' (Edit), and '删除' (Delete). In '项目管理', all four checkboxes are checked. In the other three sections, only '查看' is checked, while others are unchecked. At the bottom right of the dialog are buttons for '取消' (Cancel), '上一步' (Previous Step), and '保存' (Save).

删除角色

进入角色详情页面，点击左侧项目操作菜单，点击 **删除** 按钮删除角色，删除角色之前需要解绑和该角色相关的用户，使用中的角色无法删除。

The screenshot shows the KubeSphere web interface. On the left is a dark sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The main area has a light background. At the top, it says "KUBESPHERE" with a logo, a back arrow, and the path "角色 / event-reader". Below this, there's a title "event-reader" with a gear icon. To the right are two tabs: "权限列表" (selected) and "授权用户". Under "权限列表", there are two buttons: "/ 编辑" and "...". A red box highlights the "删除" button, which has a trash icon. Below these buttons is a section labeled "类型: 集群角色". Under "权限列表", there are two columns: "功能模块" and "可执行操作". The "功能模块" column lists "集群角色". The "可执行操作" column shows a user icon with the text "没有数据".

用户管理

本节通过以下几个方面介绍如何管理用户：

- 创建用户
- 查看用户详情
- 修改密码
- 删除用户

创建用户

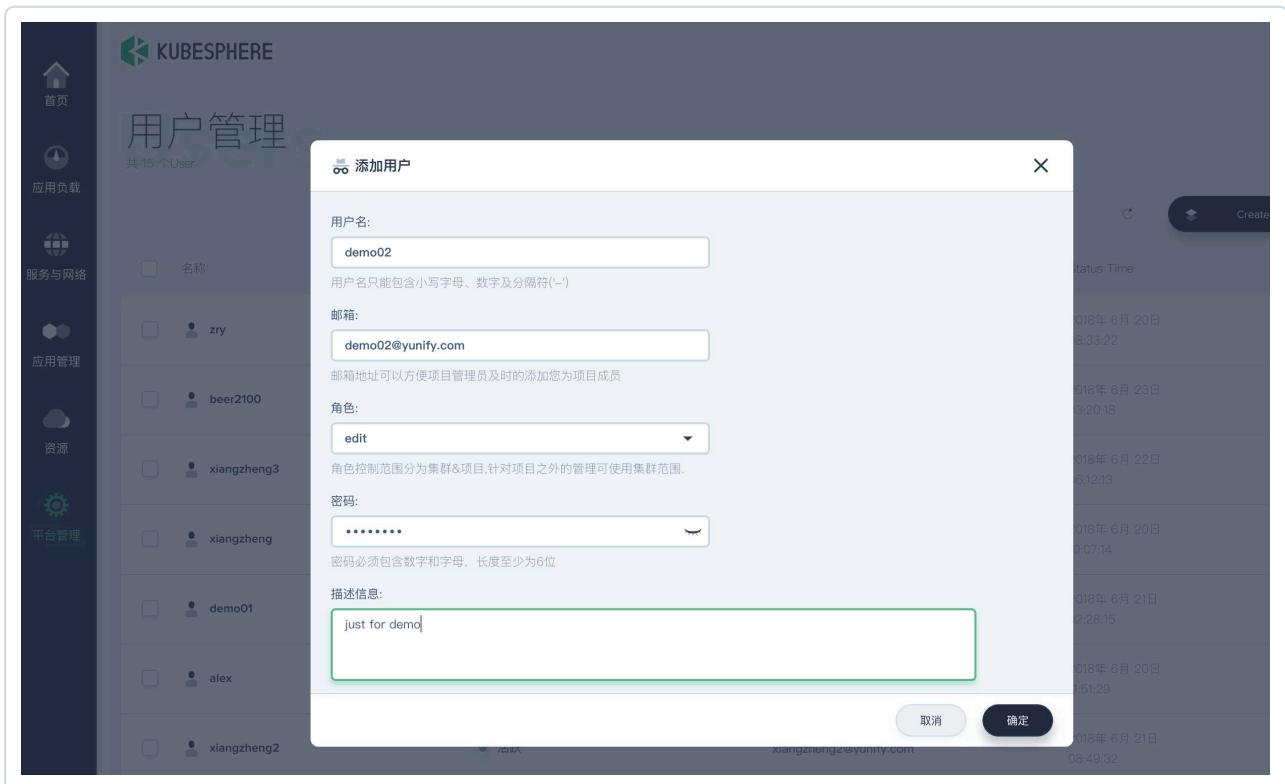
首先登录 KubeSphere 管理控制台，访问左侧菜单栏，在 **平台管理** 菜单下，点击 **用户管理** 按钮进入列表页。作为集群管理员，可以查看当前集群下所有用户信息。

The screenshot shows the KubeSphere management interface with the 'User Management' page open. On the left, there's a vertical sidebar with icons for Home, Application Load Balancer, Services & Networks, Application Management, Resources, and Platform Management. The 'Platform Management' section is currently selected. The main area has a header 'User Management' with a sub-count '共 43 个用户'. Below the header is a search bar with placeholder text '输入查询条件进行过滤' and a 'Create User' button. A table lists seven users with columns: '名称' (Name), '状态' (Status), '邮箱' (Email), and '更新时间' (Last Update). Each user row includes a checkbox and a small profile icon. The users listed are: liyuanjin, 1474682303, ubc, user, dd, 4, and 5. All users are marked as '活跃' (Active) in the status column.

名称	状态	邮箱	更新时间
liyuanjin	活跃	lyj@unify.com	2018年 7月 01日 17:30:07
1474682303	活跃	1474682303@qq.com	2018年 7月 09日 10:23:34
ubc	活跃	ubc@163.com	2018年 7月 09日 11:53:21
user	活跃	user@163.com	2018年 7月 09日 10:32:28
dd	活跃	dd@163.com	2018年 7月 09日 13:15:40
4	活跃	4@163.com	2018年 7月 09日 11:40:21
5	活跃	5@163.com	2018年 7月 09日 10:30:33

1. 点击右上角创建 **创建用户** 按钮创建用户。
2. 在弹出的用户详细信息页面，依次填写基本的用户信息之后，需要为即将创建的用户指定角色，系统默认角色为 **普通用户（regular）**，该角色仅允许对自己创建或受邀加入的项目中的资源进行管理，无权操作集群范围内其他资源。除默认角色之外，系统内还会预置以下四个角色：

- cluster-admin，有权管理集群范围内包括用户和角色在内的所有资源。
- admin，有权管理集群范围内除用户和角色之外的所有资源。
- edit，有权管理集群范围内所有的工作负载、服务和应用路由。
- view，仅允许查看集群范围内所有的工作负载、服务和应用路由。



查看用户详情

在用户列表页，点击某个用户打开用户详情页，可以看到和当前用户相关的资源信息、项目授权列表。

The screenshot shows the KubeSphere user interface for a user named 'usercase'. The left sidebar includes links for 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main content area displays the user's profile details: 状态 (Status) - 活跃 (Active), 角色 (Role) - regular, 邮箱 (Email) - uc@demo.com, 上次登录时间 (Last Login Time) - 2018年 8月 02 日 07:25, and 创建时间 (Creation Time) - 2018年 8月 02 日 07:13. Below this is a section titled '资源信息' (Resource Information) showing counts for Deployments (1), Services (2), Ingresses (1), and ConfigMaps (1). A '部署' (Deployment) table lists one entry: 'wordpress...' with status '运行中(1/1)' (Running (1/1)), created by 'admin' on '2018年 8月 01 日' at '22:48:26'. The top right corner shows the user 'admin'.

修改密码

进入用户详情页面，点击左侧项目操作菜单，点击 **修改密码** 按钮更改用户密码。

The screenshot shows the KubeSphere user interface for a user named 'demo02'. The left sidebar is identical to the previous screenshot. The main content area displays the user's profile details: 状态 (Status) - just for demo, 角色 (Role) - edit, Email: demo02@yunify.com, 上次登录时间 (Last Login Time) - 2018年 6月 23 日 09:22, and 创建时间 (Creation Time) - 2018年 6月 23 日 09:22. A '操作' (Operation) menu on the left includes '编辑' (Edit), '删除' (Delete), and '修改密码' (Change Password). Below this is a section titled '资源信息' (Resource Information) showing counts for Deployments (0), Services (0), Ingresses (0), and Pods (0). A '部署' (Deployment) table lists one entry: 'demo-project01' with status '待部署' (Pending Deployment), created by 'demo02' on '2018年 6月 23 日' at '09:22'. The top right corner shows the user 'admin'.

删除用户

进入用户详情页面，点击左侧项目操作菜单，点击 **删除** 按钮删除用户。

The screenshot shows the KubeSphere user management interface. On the left sidebar, there are several navigation items: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The current page is 'User / demo02'. The main content area displays the user profile for 'demo02' with the name 'just for demo'. Below the profile, there are two buttons: '编辑' (Edit) and a context menu icon. The context menu is open, showing options: '修改密码' (Change Password) and '删除' (Delete). To the right of the user profile, there are two tabs: '资源信息' (Resource Information) and '授权项目' (Authorized Projects). Under '资源信息', there are six resource counts: 0 Deployments, 0 StatefulSets, 0 DaemonSets, 0 Services, 0 Application Routes, and 0 Storage Classes. Below this, there is a '部署' (Deployment) section with a table header: '名称' (Name), '状态' (Status), '应用' (Application), '项目' (Project), and '更新时间' (Last Updated). A search bar at the top of the table allows filtering by name. The table body is currently empty, displaying the message '没有数据' (No Data).

服务组件

服务组件提供 KubeSphere、Kubernetes 和 OpenPitrix 集群内各项服务组件的健康状态监控，可以查看当前集群的健康状态和运行时间，能够帮助用户监测集群的状况和及时定位问题。

查看服务组件

登录 KubeSphere 管理控制台，访问左侧菜单栏，在平台管理菜单下，点击 **服务组件** 进入列表页。作为集群管理员，可以查看当前集群下所有的服务组件：

The screenshot shows the KubeSphere management console with the sidebar menu open. Under the 'Platform Management' section, the 'Service Components' option is selected. The main area displays a table of service components:

名称	健康状态	运行时间
elasticsearch-logging	健康	a month
fluentbit-logging	健康	a month
heapster	健康	a month
kibana-logging	健康	a month
kube-dns	健康	a month
kubernetes-dashboard	健康	a month
prometheus	健康	a month
openpitrix-api-gateway	健康	12 天
openpitrix-app-manager	健康	12 天
openpitrix-category-manager	健康	12 天

服务组件的作用

服务组件可以查看当前集群健康状态，当集群出现异常时，管理员可以查看是否存在某个服务组件出现异常。比如使用应用模板部署应用时，应用没有部署成功，那么就可以查看是不是 Openpitrix 的组件出现异常，管理员就可以快速定位问题，然后根据出现异常的组件进行修复。当某个服务组件出现异常时，KubeSphere、OpenPitrix 和 Kubernetes 的 Tab 显示为异常组件的数目。如下图所示，当 KubeSphere 的“ks-website”显示状态为“异常中”，在 KubeSphere 的 Tab 则会显示异常的组件数目：

The screenshot shows the KubeSphere platform interface. On the left is a dark sidebar with navigation icons and labels: 首页 (Home), 应用负载 (Application Load Balancer), 服务与网络 (Services & Networks), 应用管理 (Application Management), 资源 (Resources), and 平台管理 (Platform Management). The main area is titled "服务组件" (Service Components) and displays a list of services. At the top of the list are four tabs: 全部服务组件 (22), KubeSphere (1), Kubernetes (2), and OpenPitrix (11). The main table has columns for 名称 (Name), 健康状态 (Health Status), and 运行时间 (Run Time). The data is as follows:

名称	健康状态	运行时间
ks-account	健康	a month
ks-apiserver	健康	a month
ks-console	健康	a month
ks-website	异常中	8 天