

企业级分布式应用服务 EDAS

应用部署

应用部署

应用部署概述

您的应用开发完成后，可以通过控制台或开发工具两种方式部署。

目前，各种类型的集群都支持部署 Spring Cloud、Dubbo 和 HSF 框架应用，区别仅仅是支持的部署方式不同，详情见下表。

应用框架	可选集群	打包方式
Spring Cloud、Dubbo 和 HSF	ECS 集群	WAR/JAR
	容器服务 Kubernetes 集群	WAR/JAR/镜像
	Swarm 集群（不推荐）	WAR/JAR/镜像
	自建 Kubernetes 集群	镜像

应用运行环境

Spring Cloud、Dubbo 和 HSF 应用都可以通过 WAR 或 JAR 包部署在 ECS 集群中，但所选择的应用运行环境会有所不同。

Spring Cloud 和 Dubbo 应用通过 WAR 包部署时，需要选择 **apache-tomcat** 相关版本的运行环境。

Spring Cloud 和 Dubbo 应用通过 JAR 包部署时，需要选择 **标准 Java 应用运行环境**。

HSF 应用通过 WAR 或 JAR 包部署时，需要选择 **EDAS-Container** 相关版本的运行环境。

控制台部署应用

建议您使用 Chrome 浏览器进行控制台操作。

使用以下资源来查找部署应用的相关操作教程，以便在 EDAS 中开始开发应用。

说明：使用 JAR 包部署应用的过程跟使用 WAR 包部署应用的操作过程类似，相关操作以 WAR 包部署应用的教程来示意。



部署 ECS 集群应用（WAR）



部署容器服务 Kubernetes 版集群应用（JAR 包）



部署容器服务 Kubernetes 版集群应用（镜像）



部署 Swarm 集群应用（WAR 包）



部署 Swarm 集群应用（镜像）

工具部署应用

除在控制台完成部署操作外，您还可通过以下工具来部署应用。



通过 Maven 插件自动化部署应用



使用 CLI 快速部署 EDAS 应用



使用 Eclipse 插件快速部署应用



使用 IntelliJ IDEA 插件快速部署应用



使用云效进行持续集成和发布



使用 Jenkins 创建持续集成

制作应用容器 Docker 镜像

您可以在本地的开发工具中使用命令将应用程序打成 WAR 包或 JAR 包，直接部署；也可以基于 WAR 包或 JAR 包制作镜像，再将镜像上传到阿里云镜像仓库来部署。本文介绍如何制作不同框架应用镜像的 Dockerfile 以及如何将镜像上传到阿里云镜像仓库。

注意：在制作应用镜像前，请查阅镜像创建规约并按照规约制作镜像。

创建标准 Dockerfile

Dockerfile 是一个文本格式的配置文件，您可以使用 Dockerfile 快速创建镜像。

您可以根据实际应用框架，通过 Dockerfile 分别制作 HSF、Spring Cloud 或 Dubbo 应用镜像。下面将通过示例分别介绍如何创建这些不同框架应用的 Dockerfile。

一个 EDAS 标准 Dockerfile 描述了 EDAS 创建应用运行环境的所有指令，包括下载、安装和启动 OpenJDK、Tomcat、WAR 包和 JAR 包等。您还可以通过改变 Dockerfile 内容，完成如 OpenJDK 版本替换、Tomcat 配置修改、改变运行时环境等操作。详情请参见自定义设置 Dockerfile。

下面通过示例介绍如何制作不同框架应用镜像的 Dockerfile。

- HSF 应用的 Dockerfile 示例（基于 WAR 包）
- HSF 应用的 Dockerfile 示例（基于 JAR 包）
- Spring Cloud 或 Dubbo 应用的 Dockerfile 示例（基于 WAR 包）
- Spring Cloud 或 Dubbo 应用的 Dockerfile 示例（基于 JAR 包）

HSF 应用的 Dockerfile 示例（基于 WAR 包）

```
FROM centos:7
MAINTAINER 企业级分布式应用服务EDAS研发团队 <edas-dev@list.alibaba-inc.com>

# 安装打包必备软件
RUN yum install -y wget unzip telnet lsof net-tools bind-utils

# 准备 JDK/Tomcat 系统变量与路径
ENV JAVA_HOME /usr/java/latest
ENV CATALINA_HOME /home/admin/taobao-tomcat
ENV PATH $PATH:$JAVA_HOME/bin:$CATALINA_HOME/bin

# 设置 EDAS-Container/Pandora 应用容器版本
ENV EDAS_CONTAINER_VERSION V3.5.4
LABEL pandora V3.5.4

# 下载安装 OpenJDK
RUN yum -y install java-1.8.0-openjdk-devel

# 下载安装 Ali-Tomcat 7.0.92 到 /home/admin/taobao-tomcat 目录
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/edas-container/7.0.92/taobao-tomcat-production-7.0.92.tar.gz -O /tmp/taobao-tomcat.tar.gz && \
mkdir -p ${CATALINA_HOME} && \
```

```

tar -xvf /tmp/taobao-tomcat.tar.gz -C ${CATALINA_HOME} && \
mv ${CATALINA_HOME}/taobao-tomcat-production-7.0.59.3/* ${CATALINA_HOME}/ && \
rm -rf /tmp/taobao-tomcat.tar.gz ${CATALINA_HOME}/taobao-tomcat-production-7.0.59.3 && \
chmod +x ${CATALINA_HOME}/bin/*sh

# 根据环境变量，下载安装 EDAS Container/Pandora 应用容器版本
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/edas-
plugins/edas.sar.${EDAS_CONTAINER_VERSION}/taobao-hsf.tgz -O /tmp/taobao-hsf.tgz && \
tar -xvf /tmp/taobao-hsf.tgz -C ${CATALINA_HOME}/deploy/ && \
rm -rf /tmp/taobao-hsf.tgz

# 下载部署 EDAS 演示 WAR 包
RUN wget http://edas.oss-cn-hangzhou.aliyuncs.com/demo/hello-edas.war -O /tmp/ROOT.war && \
unzip /tmp/ROOT.war -d ${CATALINA_HOME}/deploy/ROOT/ && \
rm -rf /tmp/ROOT.war

# 设定 Tomcat 安装目录为容器启动目录，并采用 run 方式启动 Tomcat，在标准命令行输出 catalina 日志。
WORKDIR ${CATALINA_HOME}
CMD ["catalina.sh", "run"]

```

HSF 应用的 Dockerfile 示例（基于 JAR 包）

```

FROM centos:7
MAINTAINER 企业级分布式应用服务EDAS研发团队 <edas-dev@list.alibaba-inc.com>

# 安装打包必备软件
RUN yum install -y wget unzip telnet lsof net-tools bind-utils

# 准备 JDK/Tomcat 系统变量与路径
ENV JAVA_HOME /usr/java/latest
ENV CATALINA_HOME /home/admin/taobao-tomcat
ENV PATH $PATH:$JAVA_HOME/bin

# 设置 EDAS-Container/Pandora 应用容器版本
ENV EDAS_CONTAINER_VERSION V3.5.4
LABEL pandora V3.5.4

# 下载安装 OpenJDK
RUN yum -y install java-1.8.0-openjdk-devel

# 根据环境变量，下载安装 EDAS Container/Pandora 应用容器版本
RUN mkdir -p ${CATALINA_HOME}/deploy/
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/edas-
plugins/edas.sar.${EDAS_CONTAINER_VERSION}/taobao-hsf.tgz -O /tmp/taobao-hsf.tgz && \
tar -xvf /tmp/taobao-hsf.tgz -C ${CATALINA_HOME}/deploy/ && \
rm -rf /tmp/taobao-hsf.tgz

# 下载部署 EDAS 演示 JAR 包
RUN mkdir -p /home/admin/app/ && wget http://edas.oss-cn-hangzhou.aliyuncs.com/demoapp/fatjar-test-case-
provider-0.0.1-SNAPSHOT.jar -O /home/admin/app/provider.jar

# 将启动命令写入启动脚本 start.sh
RUN echo 'JAVA_HOME/bin/java -jar $CATALINA_OPTS -Djava.security.egd=file:/dev/./urandom -
Dcatalina.logs=${CATALINA_HOME}/logs -Dpandora.location=${CATALINA_HOME}/deploy/taobao-hsf.sar

```

```
"/home/admin/app/provider.jar" --server.context-path=/ --server.port=8080 --server.tomcat.uri-encoding=ISO-8859-1 --server.tomcat.max-threads=400' > /home/admin/start.sh && chmod +x /home/admin/start.sh
```

```
WORKDIR $CATALINA_HOME  
CMD ["/bin/bash", "/home/admin/start.sh"]
```

Spring Cloud 或 Dubbo 应用的 Dockerfile 示例 (基于 WAR 包)

```
FROM centos:7  
MAINTAINER 企业级分布式应用服务EDAS研发团队 <edas-dev@list.alibaba-inc.com>  
  
# 安装打包必备软件  
RUN yum install -y wget unzip telnet lsof net-tools bind-utils  
  
# 准备 JDK/Tomcat 系统变量  
ENV JAVA_HOME /usr/java/latest  
ENV CATALINA_HOME /home/admin/apache-tomcat-7.0.91  
ENV ADMIN_HOME /home/admin  
ENV PATH $PATH:$JAVA_HOME/bin:$CATALINA_HOME/bin  
RUN mkdir -p /home/admin  
  
# 下载安装 OpenJDK  
RUN yum -y install java-1.8.0-openjdk-devel  
  
# 下载安装 Tomcat  
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/apache-tomcat-7.0.91.tar.gz -O /tmp/apache-tomcat-7.0.91.tar.gz && \  
tar -xvf /tmp/apache-tomcat-7.0.91.tar.gz -C /home/admin && \  
rm /tmp/apache-tomcat-7.0.91.tar.gz && \  
chmod +x ${CATALINA_HOME}/bin/*sh  
  
RUN mkdir -p ${CATALINA_HOME}/deploy/  
  
# 增加容器内中文支持  
ENV LANG="en_US.UTF-8"  
  
# 增强 Webshell 体验  
ENV TERM=xterm  
  
# 下载部署 EDAS 演示 WAR 包  
RUN wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/demo/1.0/hello-edas.war -O /tmp/ROOT.war && \  
unzip /tmp/ROOT.war -d ${CATALINA_HOME}/deploy/ROOT/ && \  
rm -rf /tmp/ROOT.war  
  
# 设定 Tomcat 安装目录为容器启动目录，并采用 run 方式启动 Tomcat，在标准命令行输出 catalina 日志  
WORKDIR $ADMIN_HOME  
CMD ["catalina.sh", "run"]
```

Spring Cloud 或 Dubbo 应用的 Dockerfile 示例 (基于 JAR 包)

```
FROM centos:7  
MAINTAINER 企业级分布式应用服务EDAS研发团队 <edas-dev@list.alibaba-inc.com>
```

```
# 安装打包必备软件
RUN yum install -y wget unzip telnet lsof net-tools bind-utils

# 准备 JDK/Tomcat 系统变量
ENV JAVA_HOME /usr/java/latest
ENV PATH $PATH:$JAVA_HOME/bin
ENV ADMIN_HOME /home/admin

# 下载安装 OpenJDK
RUN yum -y install java-1.8.0-openjdk-devel

# 下载部署 EDAS 演示 JAR 包
RUN mkdir -p /home/admin/app/ && \
wget http://edas-hz.oss-cn-hangzhou.aliyuncs.com/demo/1.0/hello-edas-0.0.1-SNAPSHOT.jar -O \
/home/admin/app/hello-edas-0.0.1-SNAPSHOT.jar

# 增加容器内中支持
ENV LANG="en_US.UTF-8"

# 增强 Webshell 使体验
ENV TERM=xterm

# 将启动命令写入启动脚本 start.sh
RUN mkdir -p /home/admin
RUN echo '$JAVA_HOME/bin/java -jar $CATALINA_OPTS /home/admin/app/hello-edas-0.0.1-SNAPSHOT.jar' > \
/home/admin/start.sh && chmod +x /home/admin/start.sh
WORKDIR $ADMIN_HOME
CMD ["/bin/bash", "/home/admin/start.sh"]
```

自定义设置 Dockerfile

您可以在创建的标准 Dockerfile 上根据实际需求进行自定义设置。

升级 OpenJDK

在 Dockerfile 中，可下载和安装新版本的 OpenJDK，以下示例为下载安装 OpenJDK 9。

```
# 下载安装 OpenJDK 9
RUN yum -y install java-1.9.0-openjdk-devel
```

升级 EDAS Java 应用容器版本（适用于 HSF 应用）

对于部署到容器服务 Kubernetes 集群中的 HSF 应用，为使用中间件的新特性或修复原有版本的问题，可以升级应用容器版本。升级步骤如下：

登录 EDAS 控制台。在左侧导航栏选择**应用管理** > **应用列表**，在**应用列表**页面右上角单击**创建应用**，在**应用基本信息**页面查看**应用运行环境**，查看最新的 EDAS-Container 版本号（3.X.X）。

参考容器版本说明，获取 Pandora 容器版本。

替换 Dockerfile 中的版本号信息，如 3.5.4

```
# 设置 EDAS-Container/Pandora 应用容器版本
ENV EDAS_CONTAINER_VERSION V3.5.4
```

重新制作并发布应用镜像。

在 Tomcat 启动参数中添加 EDAS 运行时环境

参考应用运行时环境变量，EDAS 提供了 JVM 环境变量 EDAS_CATALINA_OPTS，包含运行所需的最小参数。同时 Ali-Tomcat 提供了自定义 JVM 参数配置选项 JAVA_OPTS，可以设置 Xmx、Xms 等参数。

```
# 设置 EDAS 应用 JVM 参数
ENV CATALINA_OPTS ${EDAS_CATALINA_OPTS}
# 设置 JVM 参数
ENV JAVA_OPTS="\
-Xmx3550m \
-Xms3550m \
-Xmn2g \
-Xss128k"
```

在本地构建镜像

从本地命令行进入 Dockerfile 所在的目录，执行 docker build 命令构建镜像：

```
docker build -t [标签名称，最好取应用名]:[版本号] .
或
docker build -t [标签名称，最好取应用名]:[版本号] -f /path/to/custom_dockerfile_name . #假如您创建好的 Dockerfile 在其他位置或名称不为 Dockerfile 时适用。
```

例如：

```
docker build -t hsf-provider:1.0.0 .
```

然后使用 docker images | grep <镜像标签名称> 命令查看本地打包好的镜像。

上传镜像到镜像仓库

您在本地生成构建好的应用镜像就可以上传到阿里云提供的容器镜像仓库（需要您提前访问阿里云镜像仓库控制台：<https://cr.console.aliyun.com>，并根据您 EDAS 中已经创建或尚未创建的应用所在的 Region，在阿

里云镜像仓库控制台切换到对应的 Region，查看镜像仓库列表，选择一个已经存在或者新创建的镜像仓库来存放您打包好的应用镜像，仓库名称最好是应用名称，便于识别）。

执行指定镜像仓库基本信息页面提供的 “将镜像推送到 Registry ” 相关命令，将本地镜像上传到该镜像仓库。

```
docker login --username=[当前登录的账号名] registry.[region_id].aliyuncs.com #这一步输入的密码为阿里云容器镜像服务默认实例 -> 访问凭证页面设置的固定或者临时密码，并非阿里云账号密码。
docker tag [本地应用镜像ID] registry.[region_id].aliyuncs.com/[命令空间名称]/[镜像仓库名称]:[镜像版本号]
docker push registry.[region_id].aliyuncs.com/[命令空间名称]/[镜像仓库名称]:[镜像版本号]
```

例如：

```
docker login --username=tdy218@gmail.com registry.cn-hangzhou.aliyuncs.com
docker tag 2b64f63601a7 registry.cn-hangzhou.aliyuncs.com/webapps/hsf-provider:1.0.0
docker push registry.cn-hangzhou.aliyuncs.com/webapps/hsf-provider:1.0.0
```

- 阿里云镜像仓库网络加速：<https://cr.console.aliyun.com/#/accelerator>

镜像创建规约

通过 Dockerfile 制作自定义镜像时，需要遵循以下规范及限制。

租户/加密信息

EDAS 应用用户鉴权与加密凭证的信息。

资源

资源类型	资源名称	描述
Secret	edas-certs	加密字典，保存 EDAS 租户信

		息。
--	--	----

环境变量

环境变量 Key	类型	描述
tenantId	String	EDAS 租户 ID
accessKey	String	鉴权 Access Key ID
secretKey	String	鉴权 Access Key Secret

本地文件

路径	类型	描述
/home/admin/.spas_key/default	File	EDAS 租户鉴权信息，包含上述 ENV 信息

服务信息

包含运行时所需连接的 EDAS 域名、端口等信息。

资源

资源类型	资源名称	描述
ConfigMap	edas-envs	EDAS 服务信息

环境变量

环境变量 Key	类型	描述
EDAS_ADDRESS_SERVER_DOMAIN	String	配置中心服务域名或 IP
EDAS_ADDRESS_SERVER_PORT	String	配置中心服务端口
EDAS_CONFIGSERVER_CLIENT_PORT	String	CS 服务端口

应用运行时环境变量（强制）

EDAS 部署时会提供以下环境变量 Env，保证应用运行正常，请勿覆盖配置。

环境变量 Key	类型	描述
POD_IP	String	POD IP
EDAS_APP_ID	String	EDAS 应用 ID

EDAS_ECC_ID	String	EDAS ECC ID
EDAS_PROJECT_NAME	String	同 EDAS_APP_ID，用于调用链解析
EDAS_JM_CONTAINER_ID	String	同 EDAS_ECC_ID，用于调用链解析
EDAS_CATALINA_OPTS	String	中间件运行时所需 CATALINA_OPTS 参数
CATALINA_OPTS	String	同 EDAS_CATALINA_OPTS，默认 TOMCAT 启动参数

控制台部署

在 ECS 集群中部署 Java Web 应用

一个 ECS 只能部署一个 ECS 集群应用。发布 ECS 集群应用是指在 ECS 实例上安装 EDAS Container，然后将应用通过 WAR 包或 JAR 包部署到 EDAS Container 中。

本文档将指导您在 EDAS 控制台上创建并用 WAR 包来部署更新和查看管理一个仅包含欢迎页面的 Java Web 应用。

通过此文档，将了解如何：

- 在 EDAS 控制台创建 ECS 集群应用
- 将 ECS 集群应用部署到网站上
- 更新已发布的 ECS 集群应用
- 验证已发布应用
- 管理应用

前提准备

完成本教程中的步骤需要完成以下前提任务：

1. 开通 EDAS 服务
2. 创建 VPC
3. 创建命名空间
4. 创建 ECS 集群

5. 同步 SLB 到 EDAS：仅当您需要配置负载均衡时需完成该配置。

创建 ECS 集群应用

登录 EDAS 控制台。

在左侧导航栏，单击**应用管理** > **应用列表**，在**应用列表**页面上方选择命名空间。

在应用列表页面右上角，单击**创建应用**。

在创建应用对话框中，输入应用相关信息。设置完成后单击**创建空应用**则创建了一个未包含实例的应用，单击**下一步**则进入应用部署配置页面。

- **命名空间**：在左侧下拉列表中选择**地域**，在右侧下拉列表选择**命名空间**。如果不选择会自动设置命名空间为**默认**。
- **集群类型**：在左侧下拉列表中选择集群类型为 **ECS 集群**，在右侧下拉列表选择一个具体的 ECS 集群。
- **应用名称**：输入应用名称。
- **应用部署方式**：选择 ECS 集群后，您可以选择 **WAR 包部署**或 **JAR 包部署**。
- **应用运行环境**：
 - **WAR 包部署**：如果您创建的是原生应用，请选择 **Apache Tomcat**；如果您创建的是 HSF 应用，请选择 **EDAS-Container**。
 - **JAR 包部署**：如果您创建的是原生应用，请选择 **标准 Java 应用运行环境**；如果您创建的是 HSF 应用，请选择 **EDAS-Container**。
- **Java 环境**：选择 **Open JDK 8**或 **Open JDK 7**。
- **应用描述**：填写应用的基本情况，输入的描述信息不超过 128 个字符。

在**应用配置**页面，添加**实例**，按照页面指示进行配置。完成设置后单击**确认创建**。

实例来源：在 ECS 集群中，您有三种方式添加实例。（若您没有选择实例，单击**创建空应用**，可以创建一个未包含实例的应用，后续可以通过应用扩容来添加实例后进行部署。）

`http://127.0.0.1:8080/_etc.html`

批次：设置批次，只有当您选择的实例数 ≥ 2 时，您才可以选择分批次数并分批发布。

- **分批方式**：选择**自动**或**手动**。当您选择自动时，您需设置**分批时间**，分批时间即为分批部署的时间间隔。

应用创建可能需要几分钟，请您耐心等待。创建完成后可以前往应用详情页查看应用。在应用详情页中**实例部署信息**页签查看实例的运行状态。如果运行状态/时间为**正常运行**，说明应用发布成功。

部署应用和更新应用

如果在创建应用时只创建了一个空应用，则可以在应用详情页单击**部署应用**进行发布，如果在创建应用时已完成部署，则可通过**部署应用**来更新应用。

在 EDAS 控制台左侧导航栏中单击**应用管理** > **应用列表**，在应用列表页面单击您要部署的应用名称，进入应用详情页面。

在**实例部署信息**查看应用是否有可用的实例，如果没有可运行的实例，则单击**应用扩容**为应用添加至少一个实例。

单击**部署应用**，根据界面指示配置部署参数并单击**发布**。

- **应用部署方式**：
 - 全新部署的应用：您可选择 **WAR 包部署**、**JAR 包部署**或**历史版本**来进行部署。
 - 部署升级的应用：您可选择第一次的部署方式或**历史版本**来进行部署。
- **文件上传方式**：（此处以 WAR 包部署示例）可选择**上传 WAR 包**或 **WAR 包地址**两种方式上传。
 - **上传 WAR 包**：单击**选择文件**，选择您要部署的 WAR 包上传。
 - **WAR 包地址**：将 WAR 包的保存地址粘贴在 WAR 包地址栏中。
- **请填写版本**：设置版本（如：1.2.0），建议不用时间戳作为版本号。
- **分组**：选择分组。
- **批次**：设置批次，如果选择2次以上的批次，需要设置**分批时间**。
- **分批方式**：选择自动。
- **Java 环境（可选）**：选择 **JRE 8** 或 **JRE 7**。
- **应用描述**（可选）：描述可以为空，描述此次发布的目的。限制在128字符以内。
- **生成 Maven 插件配置**：您可生成 Maven 插件配置然后根据通过 toolkit-maven-plugin 插件自动化部署应用进行应用部署。

验证应用发布结果

应用发布后，你可以通过查看实例运行状态或登录负载均衡配置网址来验证应用已成功发布。

查看应用实例运行状态

在应用详情页中**实例部署信息**页签查看 ECS 实例的运行状态。如果运行状态/时间为**运行正常**，说明应用发布成功。

配置公网负载均衡并访问应用

由于是在专有网络内创建发布的应用，如果没有特别配置，该应用没有公网 IP 地址。如果您的应用部署在多个 ECS 实例上，并且希望将您的应用对外开放，建议您配置公网负载均衡，以便将应用的访问流量根据转发策略分发到 ECS 实例中，增强应用的服务能力，提升应用的可用性。

在基本信息页面的应用设置区域，点击**负载均衡（公网）**右侧的**添加**。

在**添加 SLB 与应用的绑定**对话框中，设置负载均衡参数，然后点击**配置负载均衡**完成配置。

添加SLB与应用的绑定

开启SLB端口监听后，会自动在SLB上新增端口监听。
请勿在SLB控制台上删除该监听，否则将影响应用访问。

负载均衡(公网):

使用虚拟服务器组 ☒

虚拟服务器组（外网）:

新建虚拟服务器组

虚拟服务器组名称:

监听（外网）:

创建新监听: ☒

SLB 前端协议:

TCP

SLB 前端端口:

应用端口:

8080

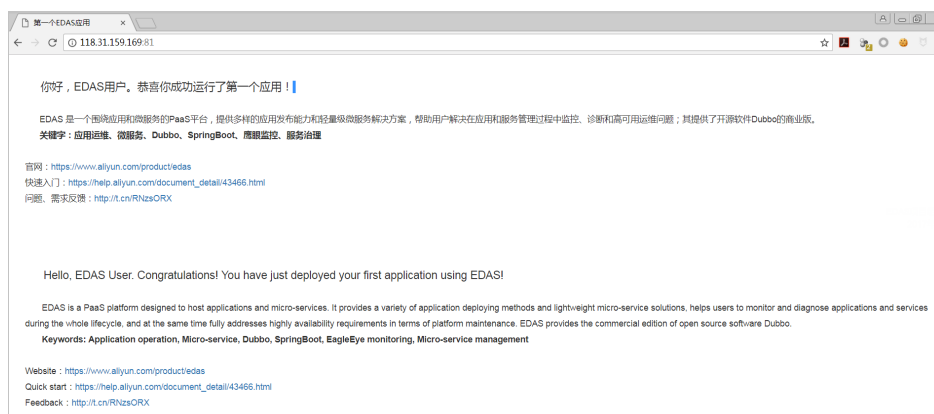
配置负载均衡

取消

- **负载均衡（公网）**：在右侧的下拉菜单中，根据实际需求，选择内网或公网的 SLB 地址。
- **使用虚拟服务器组**：虚拟服务器组是一组处理负载均衡分发的前端请求的 ECS 实例。不同的监听可以关联不同的虚拟服务器组，实现监听维度的请求转发。如果您勾选了**使用虚拟服务器组**，则需要配置虚拟服务器组参数。

- **虚拟服务器组名称**：如果您选择了**新建虚拟服务器组**，则需要在此处输入虚拟服务器组名称。系统会按照您输入的名称为您创建虚拟服务器组。
- **监听（外网）**：负载均衡服务监听规定了如何将请求转发给后端服务器。一个负载均衡实例至少添加一个监听。您可以在监听右侧的下拉菜单中选择已创建的监听端口。如果您没有创建监听，单击**创建新监听**。请勿在服务均衡管理控制台上删除该监听，否则将影响应用访问。
- **SLB 前端协议**：默认为 TCP，不可配置。
- **SLB 前端端口**：输入 SLB 的前端端口，可自行设置端口数值。
- **应用端口**：默认为 8080，不可配置。

复制配置的 SLB IP 及端口，如 `118.31.159.169:81`，在浏览器的地址中粘贴并回车，即可进入应用的欢迎页面。

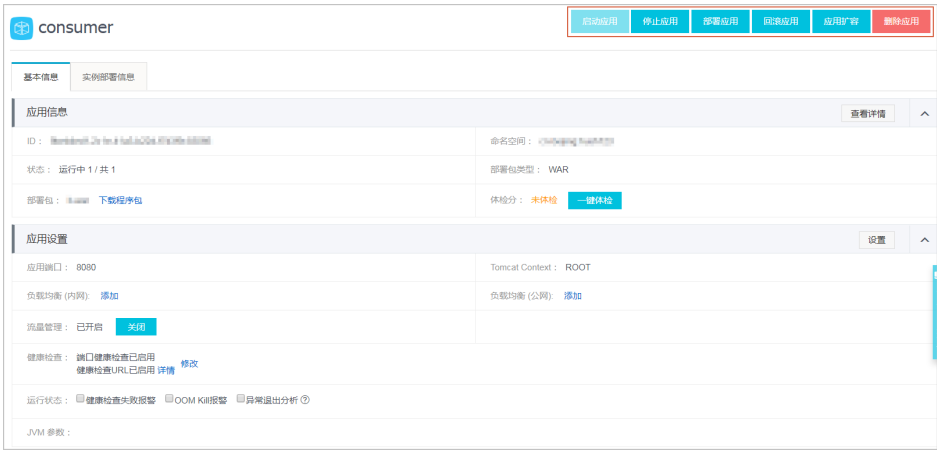


管理 ECS 集群应用

登录 EDAS 控制台。

在左侧导航栏中单击**应用管理** > **应用列表**，在应用列表页面单击具体应用名称，进入应用详情页面。

在该页面可以执行基本的管理任务，例如浏览、设置、停止、启动、扩容、回滚和删除。



在容器服务 K8S 集群中部署应用（镜像）

容器服务 Kubernetes 版提供高性能可伸缩的容器应用管理能力，支持企业级 Kubernetes 容器化应用的全生命周期管理。容器服务 Kubernetes 版简化集群的搭建和扩容等工作，整合阿里云虚拟化、存储、网络和安全能力，打造云端极佳的 Kubernetes 容器化应用运行环境。

EDAS 支持通过镜像部署容器服务 Kubernetes 版集群应用，您需准备好镜像，在容器服务 Kubernetes 版控制台中创建容器服务 Kubernetes 版集群，并将该集群导入到 EDAS 控制台，然后创建应用并完成部署。

前提条件

- 您的阿里云账号同时开通了 EDAS 服务和容器服务 Kubernetes 版。
- 在容器服务 Kubernetes 版完成角色授权。
- 制作应用镜像（容器服务 Kubernetes 集群）。

步骤一：创建容器服务 Kubernetes 集群

登录容器服务 Kubernetes 控制台。

在左侧导航栏单击**集群**，在集群列表页面右上角单击**创建 Kubernetes 集群**。

容器服务中可以创建 **Kubernetes**、**Kubernetes 托管版**和**多可用区 Kubernetes**三种类型的集群：

- 创建 **Kubernetes 集群**：您购买、添加的实例中有 3 个实例需要作为管控（Master）节点，这 3 个实例不能部署应用。您只能在另外购买的实例（Worker）上部署应用。
- 创建 **Kubernetes 托管版集群**：您购买、添加的实例中都是 Worker 节点，即可以用于部

创建多可用区 Kubernetes 集群：跟 Kubernetes 集群的区别是把节点部署在不同的可用区上，当一个区不可用时，还可在其他可用区上使用。您购买、添加的实例中有 3 个实例需要作为管控（Master）节点，这 3 个实例不能部署应用，您只能在另外购买的 Worker 实例上部署应用。

应用基本信息

应用配置

应用创建完成

* 命名空间：

华北2

-

默认

* 集群类型：

容器服务K8S集群

-

选择集群

K8S Namespace：

default

* 应用名称：

应用名称

应用描述：

请输入应用描述。

0/128

下一步：应用配置

- **命名空间**：在左侧下拉选择框选择地域；在右侧下拉选择框选择命名空间，如果不做选择命名空间则设置为默认。
- **集群类型**：在左侧下拉选择框中选择集群类型为 **容器服务 K8S 集群**，右侧下拉选择框内选择具体的集群。
- **K8S Namespace**：K8S Namespace 通过将系统内部的对象分配到不同的 Namespace 中，形成逻辑上分组的不同项目、小组或用户组，便于不同的分组在共享使用整个集群的资源的同时还能被分别管理。
 - **default**：没有其他命名空间的对象的默认命名空间。
 - **kube-system**：系统创建的对象命名空间。
 - **kube-public**：此命名空间是自动创建的，并且可供所有用户（包括未经过身份验证的用户）读取。
- **应用名称**：输入应用名称。
- **应用描述**：填写应用的基本情况。

在应用配置页面配置镜像。应用部署方式默认选择为**镜像**。在**我的镜像**区域选择您创建的镜像。

应用基本信息

应用配置

应用创建完成

应用部署方式：

镜像

配置镜像

我的镜像

华北2

	image/redis	类型：PRIVATE	来源：ALL_HUB	请选择
	image/nginx	类型：PRIVATE	来源：ALL_HUB	请选择
	image/centos	类型：PRIVATE	来源：ALL_HUB	请选择
	image/centos7	类型：PRIVATE	来源：ALL_HUB	请选择
	image/centos8	类型：PRIVATE	来源：ALL_HUB	请选择

<

1

2

3

4

5

6

...

21

>

前往

1

页共 102 条

设置 Pod。

Pod 是应用最小的部署单元。应用可以有多个 Pod，在负载均衡下，请求会被随机分配给某个 Pod 处理。

The image shows a configuration interface for Pod settings. At the top, there is a label 'Pod总数: *' followed by a numeric input field containing '1' and '+'/'-' buttons. Below this, there is a label '单Pod资源配额:' followed by two input fields: 'Cpu cores' with '0' and 'Memory (MB)' with '0'. A small note below these fields states: '单Pod的CPU和Mem系统默认显示为0（即不做配额限制）。如果您需要限制配额，请填写具体的数字。'

设置 Pod 总数。

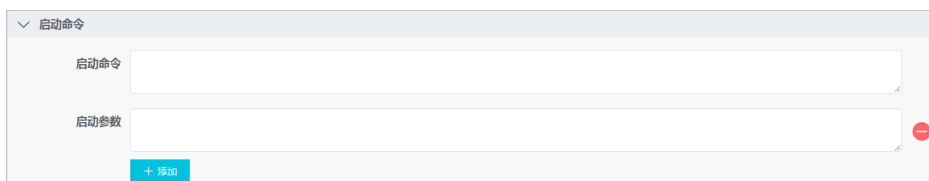
Pod 在运行失败或出现故障时，可以自动重启或者快速迁移，保证应用的高可用。有状态应用如果使用了持久化存储，能保存实例数据；无状态应用重新部署时不保存实例数据。您最多可以设置 Pod 总数为 50。

设置单 Pod 资源配额。

系统默认不做配额限制，即单 Pod 的 CPU 和 Memory 显示为 0。如果需要限制配额，请填写设置数字。

（可选）设置启动命令和启动参数。

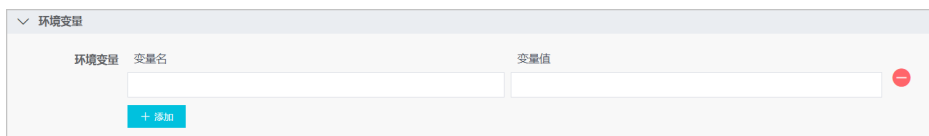
注意：若不了解原 Dockerfile 镜像的 CMD 和 ENTRYPOINT 内容，不建议修改自定义启动命令和启动参数，错误的自定义命令将导致应用创建失败。

The image shows a configuration interface for startup settings. It has a title '启动命令' (Startup Command). Below it are two text input fields: '启动命令' (Startup Command) and '启动参数' (Startup Parameters). At the bottom right of the '启动参数' field is a red minus sign. At the bottom left is a blue button labeled '+ 添加' (Add).

- **启动命令：**输入启动命令，如命令 CMD `["/usr/sbin/sshd","-D"]`，则在输入框内填写 `/usr/sbin/sshd -D`。
- **启动参数：**一个参数写一行。如 `args:["-c"; "while sleep 2"; "do echo date"; "done"]` 中包含 4 个参数，需要分为 4 行来填写。

（可选）设置环境变量。

在创建应用过程中，将所填环境变量注入到即将生成的容器中，这样可以避免常用环境变量的重复添加。

The image shows a configuration interface for environment variables. It has a title '环境变量' (Environment Variables). Below it is a table with two columns: '环境变量' (Environment Variable) and '变量值' (Variable Value). There are two empty input fields for these columns. At the bottom right is a red minus sign. At the bottom left is a blue button labeled '+ 添加' (Add).

假如您使用 MySQL 镜像时，可以参考如下环境变量：

- **MYSQL_ROOT_PASSWORD (必选项):** 用于设置 MySQL 的 root 密码，必选项。
- **MYSQL_USER 和 MYSQL_PASSWORD (可选项):** 用于添加除 root 之外的账号并设置密码。
- 。

- MYSQL_DATABASE (可选项): 用于设置生成容器时需要新建的数据库。

如使用其它类型的镜像，请根据实际需求进行配置。

(可选) 设置持久化存储。

在阿里云容器服务 Kubernetes 集群中，原生的 Volume 对象所对应的物理储存为非持久化的，它的生命周期与 Kubernetes Pod 一致，是一个暂态的储存对象。使用持久化的储存服务—文件储存，可以永久保存实例数据，应用在升级或迁移后不丢失。

说明：设置持久化存储前，请确保您的 EDAS 账号已开通 NAS 服务。同时，由于 NAS 服务的计费方式为按量付费，请确保当前账户拥有足够的余额或计费方式为后付费。

存储类型：默认为 NAS 文件存储，不可配置。

存储服务类型：目前仅支持 SSD 性能型，不可配置。

- **NAS 选择：**

- **新购一个 NAS：**选择 NAS 挂载目录和本地挂载目录。单一 Region 内 NAS 文件系统上限为 10 个，若已超过 10 个会创建失败。如有特殊需求，请提交工单。
- **使用已有 NAS：**选择一个已有的 NAS。您最多可以创建两个挂载点。不符合要求的 NAS 将不会显示在下拉列表中。

- **挂载目录：**设置挂载目录命令。

(可选) 设置本地存储。

您可以将宿主机的文件系统的一部分映射到容器内部，以满足您的需求。在使用该功能之前，建议阅读 `hostpath`，并考虑该方案的合理性。

文件类型说明：

名称	Value	含义
----	-------	----

默认	空字符串	不执行类型检查，直接挂载
（新建）文件目录	DirectoryOrCreate	文件目录，如果不存在，则新建
文件目录	Directory	文件目录，不存在则容器启动失败
（新建）文件	FileOrCreate	文件，如果不存在，则新建
文件	File	文件，不存在则容器启动失败
Socket	Socket	标准Unix Socket文件，不存在则容器启动失败
CharDevice	CharDevice	字符设备文件，不存在则容器启动失败
BlockDevice	BlockDevice	块储存设备文件，不存在则容器启动失败

说明：此步骤中，您无需关注 Value 列的内容。但在应用创建完成后，可能会在 API 中使用。

（可选）设置应用生命周期管理。

容器服务 Kubernetes 版集群应用有两种状态：

无状态应用：支持多副本部署。重新部署时不保存实例数据。适用于以下使用场景：

- Web 应用，应用升级或迁移时，实例内数据不保留。
- 需要灵活水平扩展，以应对业务量骤然变化的应用。

有状态应用：区别于无状态应用，有状态应用会存储需要持久化的数据，在应用升级或迁移时，实例内数据不会丢失。适用于以下使用场景：

- 需要频繁通过 SSH 到容器进行操作。
- 数据需要持久化存储（如数据库应用 MySQL 等），或者集群之间有选举特性，服务发现的应用，如 ZooKeeper，etcd等。

有状态应用可以选择设置应用生命周期管理。

应用生命周期管理

PostStart配置

PreStop配置

Liveness配置

Readiness配置

选择模式

Exec

HttpGet

TcpSocket

命令脚本

+

添加

生命周期管理脚本说明：

Poststart 脚本：一种容器钩子。该钩子在容器被创建后立刻触发，通知容器它已经被创

PreStop 脚本：一种容器钩子。该钩子在容器被删除前触发，其所对应的 hook handler 必须在删除该容器的请求发送给 Docker daemon 之前完成。在该钩子对应的 hook handler 完成后不论执行的结果如何，Docker daemon 会发送一个 SIGTERM 信号量给 Docker daemon 来删除该容器。参考 Container Lifecycle Hooks

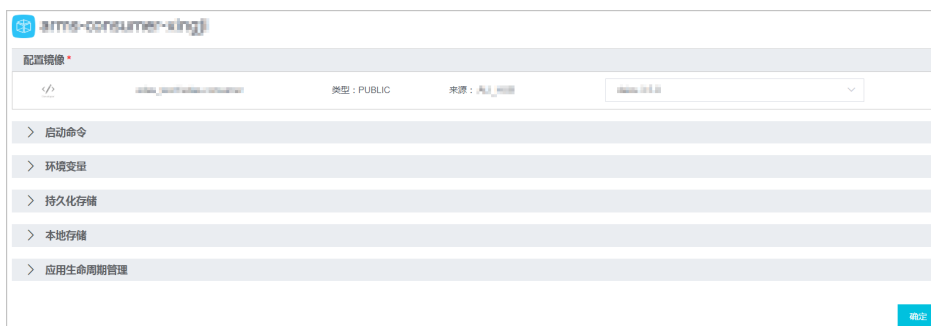
Readiness 脚本：一种探测容器状态的探针，探测应用是否启动完成并且处于正常服务状态。如果不正常，则更新容器的状态。参考 Pod Lifecycle

应用创建可能需要几分钟，创建过程中，可以通过应用变更单跟踪创建的过程。Kubernetes 应用无需部署，创建完成即部署完成，创建完成后，返回应用详情页面查看实例部署信息中 Pod 状态若为**正常运行**则说明应用发布成功。

容器服务 K8S 集群中的应用在创建时已经进行了部署，故部署应用功能用来更新应用，您可以更新镜像版本和重新设置环境变量等高级设置选项来升级应用。

在页面右上角单击部署应用。

在**部署应用**页面，您可更改**配置镜像**，并配置**启动命令**、**环境变量**、**持久化存储**、**本地存储**和**应用生命周期管理**等高级设置选项。完成设置后单击**确定**。



升级之后，应用详情页面上的镜像版本会变化为升级后的版本。在**实例部署信息**页面查看应用的运行状态，如果显示为**正常运行**则表示应用更新成功。

后续操作

应用创建完毕后，可以通过添加公网 SLB 实现公网访问，添加内网 SLB 实现同 VPC 内所有节点能够通过私网负载均衡访问您的应用。相关操作请参阅**绑定 SLB 到 K8S 集群中的应用**。

问题反馈

如果在使用容器服务 Kubernetes 集群过程中有任何疑问，欢迎您扫描下面的二维码加入钉钉群进行反馈。



在容器服务 K8S 集群中部署应用（WAR 或 JAR）

容器服务 Kubernetes 版提供高性能可伸缩的容器应用管理能力，支持企业级 Kubernetes 容器化应用的全生命周期管理。容器服务 Kubernetes 版简化集群的搭建和扩容等工作，整合阿里云虚拟化、存储、网络和安全能力，打造云端极佳的 Kubernetes 容器化应用运行环境。

您需在容器服务 Kubernetes 版控制台中创建容器服务 Kubernetes 版集群，并将该集群导入到 EDAS 控制台，然后将应用通过 WAR 或 JAR 部署到容器服务 Kubernetes 集群中。您可以通过镜像部署容器服务 Kubernetes 版集群应用，详情请参见在容器服务K8S集群中部署应用（镜像）

前提条件

- 您的阿里云账号同时开通了 EDAS 服务和容器服务 Kubernetes 版。
- 在容器服务 Kubernetes 版完成角色授权。

- 制作应用镜像（容器服务 Kubernetes 集群）。

步骤一：创建容器服务 Kubernetes 集群

登录容器服务 Kubernetes 控制台。

在左侧导航栏单击**集群**，在集群列表页面右上角单击**创建 Kubernetes 集群**。

容器服务中可以创建 **Kubernetes**、**Kubernetes 托管版**和**多可用区 Kubernetes**三种类型的集群：

- 创建 **Kubernetes 集群**：您购买、添加的实例中有 3 个实例需要作为管控（Master）节点，这 3 个实例不能部署应用。您只能在另外购买的实例（Worker）上部署应用。
- 创建 **Kubernetes 托管版集群**：您购买、添加的实例中都是 Worker 节点，即可以用于部署应用的实例。
- 创建**多可用区 Kubernetes 集群**：跟 **Kubernetes 集群**的区别是把节点部署在不同的可用区上，当一个区不可用时，还可在其他可用区上使用。您购买、添加的实例中有 3 个实例需要作为管控（Master）节点，这 3 个实例不能部署应用，您只能在另外购买的 Worker 实例上部署应用。

步骤二：在 EDAS 控制台中导入容器服务 K8S 集群

登录 EDAS 控制台。

在左侧导航栏中选择**资源管理** > **集群**。

在**集群列表**页面单击**容器服务 K8S 集群**。在集群列表中选择已创建的容器服务 Kubernetes 集群，在该集群**操作列**单击**导入**，然后在**导入 Kubernetes 集群**的对话框上单击**导入**。



集群名称/ID	网络类型	集群类型	实例数	集群状态	命名空间	创建时间	操作
hmqgaw001a-1-bus	VPC 网络	容器服务K8S集群	6	运行中	华北2	2019-01-23 10:47:40	导入
hmqgaw001a-1-bus	VPC 网络	容器服务K8S集群	6	运行中	华北2	2018-12-27 17:58:57	删除
hmqgaw001a-1-bus	VPC 网络	容器服务K8S集群	0	删除失败	华北2	2019-01-23 10:47:40	导入

- 当创建的容器服务 K8S 集群的**操作列**的选项按钮变为**删除**，且集群状态为**运行中**，则表示容器服务的 Kubernetes 版集群已成功导入到 EDAS。

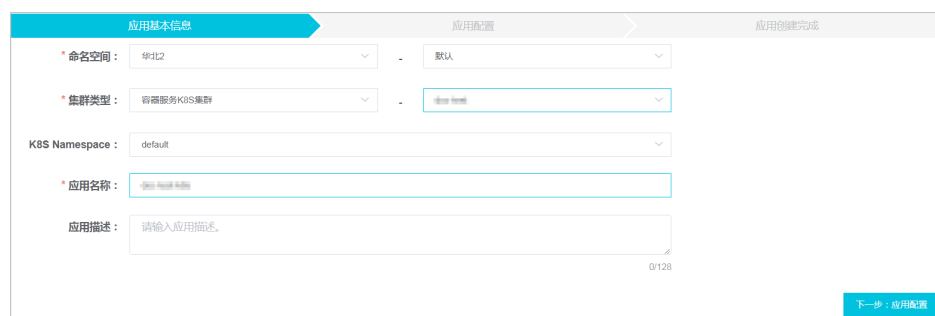
步骤三：在容器服务 Kubernetes 版集群中部署应用

说明：WAR 和 JAR 的部署步骤一致，本文以通过 JAR 为例介绍如何部署应用。

登录 EDAS 控制台。在左侧导航栏中单击**应用管理**，进入应用列表页面。

在应用列表页面选择**地域**（Region）和**命名空间**，然后在页面右上角单击**创建应用**。

在**应用基本信息**页面中设置应用的基本信息和参数，然后单击**下一步:应用配置**。



- **命名空间**：在左侧下拉选择框选择地域；在右侧下拉选择框选择命名空间，如果不做选择命名空间则设置为**默认**。
- **集群类型**：在左侧下拉选择框中选择集群类型为 **容器服务 K8S 集群**，右侧下拉选择框内选择具体的集群。
- **K8S Namespace**：K8S Namespace 通过将系统内部的对象分配到不同的 Namespace 中，形成逻辑上分组的不同项目、小组或用户组，便于不同的分组在共享使用整个集群的资源的同时还能被分别管理。
 - **default**：没有其他命名空间的对象的默认命名空间。
 - **kube-system**：系统创建的对象命名空间。
 - **kube-public**：此命名空间是自动创建的，并且可供所有用户（包括未经过身份验证的用户）读取。
- **应用名称**：输入应用名称。
- **应用描述**：填写应用的基本情况。

在**应用配置**页面**应用部署方式**右侧选择 **JAR 包部署**并设置部署参数。

- **应用运行环境**：在下拉列表中选择。HSF 应用请选择具体版本的 **EDAS-Container**，Spring Cloud 或 Dubbo 应用请选择 **标准 Java 运行环境**。
- **Java 环境**：在下拉列表中选择 **Open JDK 8** 或 **Open JDK 7**。
- **文件上传方式**：在下拉列表中选择 **上传 JAR 包**。
- **上传 JAR 包**：单击**选择文件**，在弹出的对话框中选择应用的 JAR 包文件。
- **版本**：填写该应用的版本。
- **Pod 总数**：设置该应用要部署的 Pod 个数。
- **单 Pod 资源配额**：设置单个 Pod 的 CPU 和 Memory。

（可选）设置启动命令和启动参数。

注意：若不了解原 Dockerfile 镜像的 CMD 和 ENTRYPOINT 内容，不建议修改自定义启动命令和启动参数，错误的自定义命令将导致应用创建失败。

- **启动命令**：输入启动命令，如命令 CMD `["/usr/sbin/sshd","-D"]`，则在输入框内填写 `/usr/sbin/sshd -D`。
- **启动参数**：一个参数写一行。如 `args:["-c"; "while sleep 2"; "do echo date"; "done"]` 中包含 4 个参数，需要分为 4 行来填写。

（可选）设置环境变量。

在创建应用过程中，将所填环境变量注入到即将生成的容器中，这样可以避免常用环境变量的重复添加。

如果您需要设置 JVM 堆内存、JVM 属性参数、javaagent等参数，可以在设置环境变量这一步添加

相关参数：

变量名：CATALINA_OPTS ， 变量值：\${EDAS_CATALINA_OPTS} [您需要添加的参数]

企业级分布式应用服务

概览

资源管理

应用管理

命名空间

应用列表

配置管理

灰度发布

微服务管理

组件中心

系统管理

应用部署方式：☐ 镜像 ☒ WAR包部署 ☐ JAR包部署

应用运行环境：

Java环境：

文件上传方式：

上传WAR包：

选择文件 http://edas-hz.oss-cn-hangzhou-internal.aliyuncs.com/apps/K8S_APP_ID/891cat8c-d845-453b-adab-c191af18df32/hsf-provider.war

hsf-provider.war

下载WAR样例包

版本：

Pod总数：

单Pod资源配置：

Cpu cores

Memory (MB)

单Pod的CPU和Mem系统默认为0（即不做配额限制）。如果您需要限制配额，请填写具体的数字。

启动命令

环境变量

环境变量	变量名	变量值
	CATALINA_OPTS	\${EDAS_CATALINA_OPTS} -Xmx768m -Dhsf.publish.default.tid=false -Dhsf.sui

例如：

假如您使用 MySQL 镜像时，可以参考如下环境变量：

- MYSQL_ROOT_PASSWORD (必选项): 用于设置 MySQL 的 root 密码，必选项。
- MYSQL_USER 和 MYSQL_PASSWORD (可选项): 用于添加除 root 之外的账号并设置密码。
- MYSQL_DATABASE (可选项): 用于设置生成容器时需要新建的数据库。

如使用其它类型的镜像，请根据实际需求进行配置。

（可选）设置持久化存储。

在阿里云容器服务 Kubernetes 集群中，原生的 Volume 对象所对应的物理储存为非持久化的，它的生命周期与 Kubernetes Pod 一致，是一个暂态的储存对象。使用持久化的储存服务—文件储存，可以永久保存实例数据，应用在升级或迁移后不丢失。

说明：设置持久化存储前，请确保您的 EDAS 账号已开通 NAS 服务。同时，由于 NAS 服务的计费方式为按量付费，请确保当前账户拥有足够的余额或计费方式为后付费。

持久化存储

存储类型 ☒ NAS文件存储

存储服务类型 ☒ SSD性能型

NAS选择 ☒ 新购一个NAS ☐ 使用已有NAS

使用NAS文件存储服务时,按实际使用量付费。详细收费标准请参考[NAS计费说明](#)

挂载目录 本地挂载目录

+

添加

存储类型：默认为 NAS 文件存储，不可配置。

存储服务类型：目前仅支持 SSD 性能型，不可配置。

- **NAS 选择**：

- **新购一个 NAS**：选择 NAS 挂载目录和本地挂载目录。单一 Region 内 NAS 文件系统上限为 10 个，若已超过 10 个会创建失败。如有特殊需求，请提交工单。
- **使用已有 NAS**：选择一个已有的 NAS。您最多可以创建两个挂载点。不符合要求的 NAS 将不会显示在下拉列表中。

- **挂载目录**：设置挂载目录命令。

（可选）设置本地存储。

您可以将宿主机的文件系统的一部分映射到容器内部，以满足您的需求。在使用该功能之前，建议阅读 `hostpath`，并考虑该方案的合理性。

文件类型说明：

名称	Value	含义
默认	空字符串	不执行类型检查，直接挂载
（新建）文件目录	DirectoryOrCreate	文件目录，如果不存在，则新建
文件目录	Directory	文件目录，不存在则容器启动失败
（新建）文件	FileOrCreate	文件，如果不存在，则新建
文件	File	文件，不存在则容器启动失败
Socket	Socket	标准 Unix Socket 文件，不存在则容器启动失败
CharDevice	CharDevice	字符设备文件，不存在则容器启动失败
BlockDevice	BlockDevice	块储存设备文件，不存在则容器启动失败

说明：此步骤中，您无需关注 Value 列的内容。但在应用创建完成后，可能会在 API 中使用。

（可选）设置应用生命周期管理。

容器服务 Kubernetes 版集群应用有两种状态：

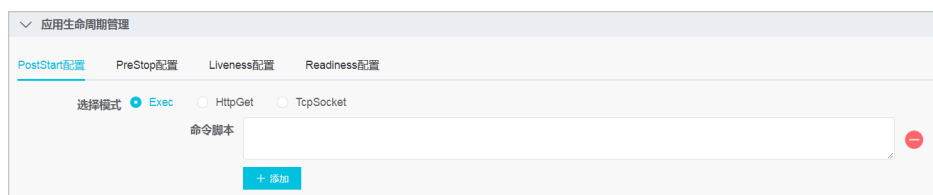
无状态应用：支持多副本部署。重新部署时不保存实例数据。适用于以下使用场景：

- Web 应用，应用升级或迁移时，实例内数据不保留。
- 需要灵活水平扩展，以应对业务量骤然变化的应用。

有状态应用：区别于无状态应用，有状态应用会存储需要持久化的数据，在应用升级或迁移时，实例内数据不会丢失。适用于以下使用场景：

- 需要频繁通过 SSH 到容器进行操作。
- 数据需要持久化存储（如数据库应用 MySQL 等），或者集群之间有选举特性，服务发现的应用，如 ZooKeeper，etcd 等。

有状态应用可以选择设置应用生命周期管理。



生命周期管理脚本说明：

Poststart 脚本：一种容器钩子。该钩子在容器被创建后立刻触发，通知容器它已经被创建。该钩子不需要向其所对应的 hook handler 传入任何参数。如果该钩子对应的 hook handler 执行失败，则该容器会被杀死，并根据该容器的重启策略决定是否要重启该容器。参考 Container Lifecycle Hooks

PreStop 脚本：一种容器钩子。该钩子在容器被删除前触发，其所对应的 hook handler 必须在删除该容器的请求发送给 Docker daemon 之前完成。在该钩子对应的 hook handler 完成后不论执行的结果如何，Docker daemon 会发送一个 SIGTERM 信号量给 Docker daemon 来删除该容器。参考 Container Lifecycle Hooks

Liveness 脚本：一种探测容器状态的探针，探测应用是否处于健康状态。如果不健康，则删除重建容器。参考 Pod Lifecycle

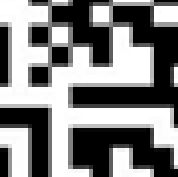
Readiness 脚本：一种探测容器状态的探针，探测应用是否启动完成并且处于正常服务状态。如果不正常，则更新容器的状态。参考 Pod Lifecycle

设置完成后，单击**确认创建**。

应用创建可能需要几分钟，创建过程中，可以通过**应用变更单**跟踪创建的过程。Kubernetes 应用无需部署，创建完成即部署完成，创建完成后，返回应用详情页面查看实例部署信息中 Pod 状态若为**正常运行**则说明应用发布成功。



问题反馈



部署 Swarm 集群应用

部署 Swarm 集群应用 (WAR 包)

在 EDAS 中，Swarm 集群应用是特指在 Swarm 集群内创建的应用。Swarm 集群支持 Docker 镜像、WAR 包或 JAR 包部署的应用。一个 ECS 可以部署多个运行在不同的 Docker 容器中的应用。发布 Swarm 集群应用是指在 ECS 实例上的 Docker 中安装 EDAS container，然后将应用通过 WAR 包、JAR 包或镜像部署到 EDAS container 中。

本文档将指导您在 EDAS 控制台来创建并用 WAR 包来部署更新和查看管理一个 Swarm 集群应用。(JAR 包部署应用的过程与 WAR 包部署应用过程类似，我们以 WAR 包部署方式为例说明)。

通过此文档，将了解如何：

- 创建 Swarm 集群应用
- 部署 Swarm 集群应用
- 更新应用
- 验证已发布应用
- 管理应用

前提准备

完成本教程中的步骤需要完成以下前提任务：

1. 开通 EDAS 服务
2. 创建 VPC
3. 创建 ECS 实例
4. 创建命名空间
5. 创建 Swarm 集群
6. 同步 SLB 到 EDAS：仅当你需要配置负载均衡时需完成该配置。

创建 Swarm 集群应用

在 EDAS 控制台中，在创建的 Swarm 集群下创建一个 Swarm 集群应用。

登录 EDAS 控制台。

在左侧导航栏中单击**应用管理** > **应用列表**，在应用列表页面选择**地域** (Region) 和**命名空间**。

在应用列表页面右上角，单击**创建应用**。

在**应用基本信息**页面中设置应用的基本信息和参数，设置完成后单击**创建空应用**则创建了一个未包含实例的应用，单击**下一步**则进入应用部署配置页面。

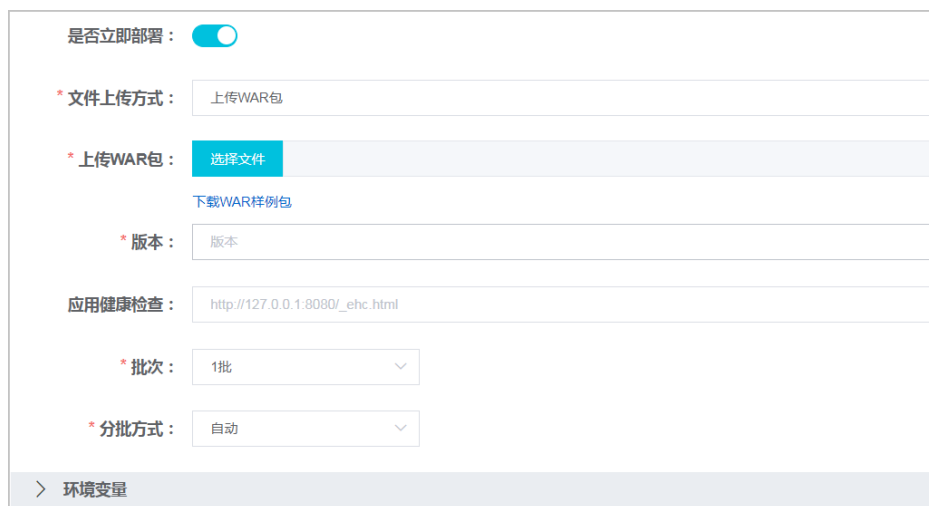
- **命名空间**：在左侧下拉选择框选择**地域**；在右侧下拉选择框选择**命名空间**，如果不做选择命名空间则设置为**默认**。
- **集群类型**：在左侧下拉列表中选择集群类型为 **Swarm 集群**，在右侧下拉列表选择一个具体的 Swarm 集群。
- **应用名称**：输入应用名称。
- **应用部署方式**：选择 Swarm 集群后，您可以选择**镜像**、**WAR 包部署**或 **JAR 包部署**。
- **应用运行环境**：选择应用运行环境，例如 EDAS-Container 3.5.1 [支持FatJar部署]。
- **Java 环境**：选择 **Open JDK 8**或 **Open JDK 7**。
- **CPU 规格**：设置 CPU 规格。如果您在创建 Swarm 集群时设置了 **CPU 共享比例**为 1:2 或 1:4，则此处规则默认为**共享模式**；如果创建时未设置 **CPU 共享比例** 或设置 **CPU 共享比例**为 1:1，则此处您可以手动设置 CPU 规格。
- **规格**：设置内存规格。
- **应用描述**：填写应用的基本情况，输入的描述信息不超过 128 个字符。

在**应用配置**页面，添加**实例个数**并**选定实例列表**，按照页面指示进行配置。完成设置后单击**确认创建**。

- **请选择实例个数**：在右侧微调输入框内设置实例个数，实例个数由您所选择集群内的实例数决定。
 - 如果没有设置实例个数（默认为 0），单击**创建空应用**，可以创建一个空应用。后续再通过**应用扩容**进行实例的添加，通过**部署应用**完成应用部署。
 - 如果设置了大于 0 的实例个数，未对应用进行部署即单击**确认创建**，可以创建一个包含实例的空应用。之后可通过**部署应用**完成应用部署。

选定实例列表：在选择好实例个数后，此处会自动选定好实例。

是否立即部署：选择实例后才可点击打开。点击打开后按照界面进行配置。



文件上传方式：可选择**上传 WAR 包**或 **WAR 包地址**两种方式上传。

- **上传 WAR 包**：单击**选择文件**，选择您要部署的 WAR 包上传。

WAR 包地址：复制 WAR 包的存放地址，将该地址粘贴在 WAR 包地址栏中。

注意：应用部署程序包名仅允许字母、数字，及中划线(-)、下划线(_)两个特殊符号。

版本：设置版本（如：1.1.0），不建议用时间戳作为版本号。

应用健康检查（可选）：设置应用健康检查的 URL。应用的健康会在容器启动后/运行时检查应用的状态是否正常，会根据应用的健康检查结果来执行服务路由。设置参考示例为 `http://127.0.0.1:8080/_etc.html`

批次：设置批次，只有当您选择的实例数 ≥ 2 时，您才可以选择分批次数并分批发布。

分批方式：选择**自动**或**手动**。当您选择自动时，您需设置**分批时间**，分批时间即为分批部署的时间间隔。

- **环境变量**（可选）：单击展开**环境变量**。配置**环境变量的变量名和变量值**。单击**添加**，可以设置多个环境变量。

说明：如果未勾选**是否立即部署**，在应用创建完成后，在应用详情页单击部署应用，也可以完成应用的部署。

在**创建完成**页面，单击**应用详情页**，查看应用的基本信息和实例部署信息。在应用详情页中**实例部署信息**页签查看实例的运行状态，如果运行状态/时间为**正常运行**，说明应用发布成功。

部署应用和更新应用

如果在创建应用时只创建了一个空应用，则可以在应用详情页单击**部署应用**进行发布，如果在创建应用时已完成部署，则可通过**部署应用**来更新应用。

登录 EDAS 控制台，在左侧导航栏中单击**应用管理**，在应用列表页面单击具体应用名称，进入**基本信息**页面。

在**实例部署信息**查看应用是否有可用的实例，如果没有可运行的实例，则单击**应用扩容**为应用添加至少一个实例。

单击**部署应用**，根据界面指示配置部署参数并单击**发布**。

部署应用

应用部署方式：☐ 镜像 ☒ **WAR包部署** ☐ JAR包部署 ☐ 历史版本

* 应用运行环境：

* 文件上传方式： [下载样例工程](#)

* 上传war包： [选择文件](#)

* 请填写版本： [用时间戳作为版本号](#)

* 分组：

* 批次：

* 分批方式：

Java环境：

应用描述：

> 环境变量

> 生成Maven插件配置

[了解更多快速部署方式](#)

[发布](#) [取消](#)

- **应用部署方式：**

- 未部署过应用：您可选择**镜像**、**WAR 包部署**、**JAR 包部署**或**历史版本**来进行部署。
- 部署升级的应用：您可选择第一次的部署方式或**历史版本**来进行部署。

- **应用运行环境：**选择 EDAS Container 版本。如 EDAS Container 3.5.1 [支持 FatJar 部署]。

- **文件上传方式**：（此处以 WAR 包部署示例）可选择**上传 WAR 包**或**WAR 包地址**两种方式上传。
 - **上传 WAR 包**：单击**选择文件**，选择您要部署的 WAR 包上传。
 - **WAR 包地址**：将该您要部署的 WAR 包保存地址粘贴在 WAR 包地址栏中。
- **请填写版本**：设置版本（如：1.2.0），建议不用时间戳作为版本号。
- **分组**：选择分组。
- **批次**：设置批次，如果选择2次以上的批次，需要设置**分批时间**。
- **分批方式**：选择自动。
- **Java 环境（可选）**：选择 **JRE 8** 或 **JRE 7**。
- **应用描述（可选）**：描述可以为空，描述此次发布的目的。限制在128字符以内。
- **环境变量（可选）**：参考如何设置环境变量进行设置。
- **生成 Maven 插件配置**：您可生成 Maven 插件配置然后根据通过 toolkit-maven-plugin 插件自动化部署应用进行应用部署。

应用在重新部署后，会跳转到**变更详情**页面，查看部署流程和执行日志。部署流程执行完成后，执行状态变为**执行成功**则表示部署成功。

验证应用发布结果

应用发布后，你可以通过查看实例运行状态或登录负载均衡配置网址来验证应用已成功发布。

查看应用实例运行状态

在应用详情页中**实例部署信息**页签查看 ECS 实例的运行状态。如果运行状态/时间为**运行正常**，说明应用发布成功。

配置公网负载均衡并访问应用

由于是在专有网络内创建发布的应用，如果没有特别配置，该应用没有公网 IP 地址。如果您的应用部署在多个 ECS 实例上，并且希望将您的应用对外开放，建议您配置公网负载均衡，以便将应用的访问流量根据转发策略分发到 ECS 实例中，增强应用的服务能力，提升应用的可用性。

在基本信息页面的应用设置区域，点击**负载均衡（公网）**右侧的**添加**。

在**添加 SLB 与应用的绑定**对话框中，设置负载均衡参数，然后点击**配置负载均衡**完成配置。

添加SLB与应用的绑定

开启SLB端口监听后，会自动在SLB上新增端口监听。
请勿在SLB控制台上删除该监听，否则将影响应用访问。

负载均衡(公网)：

使用虚拟服务器组 ☒

虚拟服务器组（外网）：

新建虚拟服务器组

虚拟服务器组名称：

监听（外网）：

创建新监听：☒

SLB 前端协议：

TCP

SLB 前端端口：

应用端口：

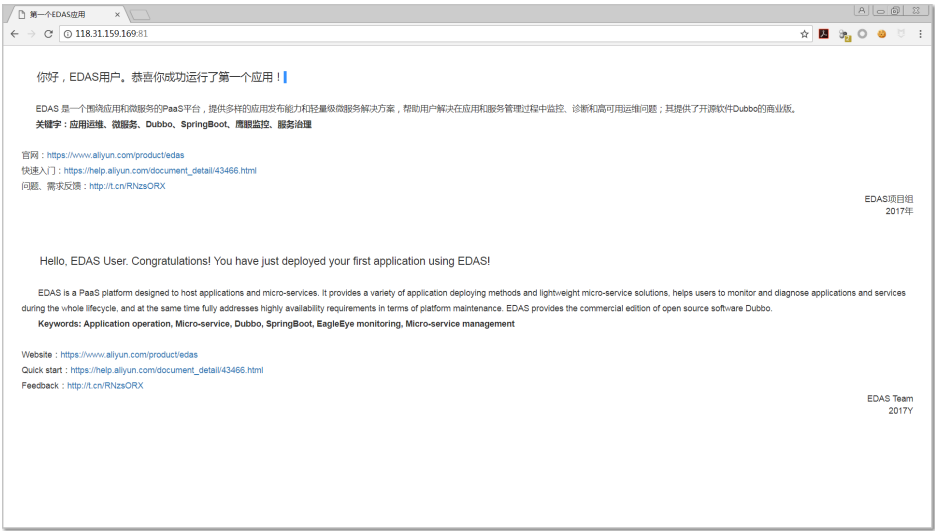
8080

配置负载均衡

取消

- **负载均衡（公网）**：在右侧的下拉菜单中，根据实际需求，选择内网或公网的 SLB 地址。
- **使用虚拟服务器组**：虚拟服务器组是一组处理负载均衡分发的前端请求的 ECS 实例。不同的监听可以关联不同的虚拟服务器组，实现监听维度的请求转发。如果您勾选了**使用虚拟服务器组**，则需要配置虚拟服务器组参数。
- **虚拟服务器组名称**：如果您选择了**新建虚拟服务器组**，则需要在此处输入虚拟服务器组名称。系统会按照您输入的名称为您创建虚拟服务器组。
- **监听（外网）**：负载均衡服务监听规定了如何将请求转发给后端服务器。一个负载均衡实例至少添加一个监听。您可以在监听右侧的下拉菜单中选择已创建的监听端口。如果您没有创建监听，单击**创建新监听**。请勿在服务均衡管理控制台上删除该监听，否则将影响应用访问。
- **SLB 前端协议**：默认为 TCP，不可配置。
- **SLB 前端端口**：输入 SLB 的前端端口，可自行设置端口数值。
- **应用端口**：默认为 8080，不可配置。

复制配置的 SLB IP 及端口，如 118.31.159.169:81，在浏览器的地址中粘贴并回车，即可进入应用的欢迎页面。

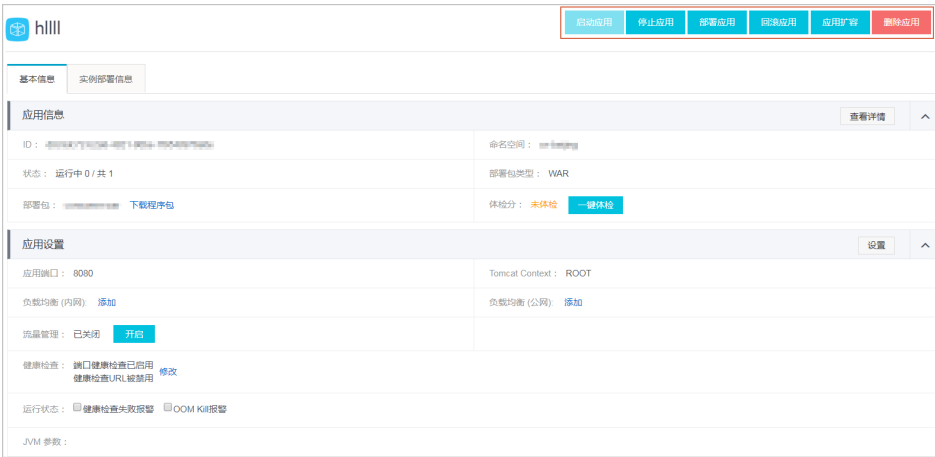


管理 Swarm 集群应用

登录 EDAS 控制台。

在左侧导航栏中单击**应用管理**，在应用列表页面单击具体 Swarm 集群应用名称，进入**基本信息**页面。

在该页面可以执行基本的管理任务，例如浏览、设置、停止、启动、扩容、回滚和删除。



部署 Swarm 集群应用（镜像）

在 EDAS 中，Swarm 集群应用是特指在 Swarm 集群内创建的应用。Swarm 集群支持 Docker 镜像、WAR 包或 JAR 包部署的应用。一个 ECS 可以部署多个运行在不同的 Docker 容器中的应用。发布 Swarm 集群应用是指在 ECS 实例上的 Docker 中安装 EDAS container，然后将应用通过 WAR 包、JAR 包或镜像部署到 EDAS container 中。

本文档将指导您在 EDAS 控制台来创建并用镜像来部署更新和查看管理一个 Swarm 集群应用。

通过此文档，将了解如何：

- 创建 Swarm 集群应用
- 部署 Swarm 集群应用
- 更新应用
- 验证已发布应用
- 管理应用

前提准备

完成本教程中的步骤需要完成以下前提任务：

1. 开通 EDAS 服务
2. 创建 VPC
3. 创建 ECS 实例
4. 创建命名空间
5. 创建 Swarm 集群
6. 创建镜像
7. 同步 SLB 到 EDAS：仅当您需要配置负载均衡时需完成该配置。

创建 Swarm 集群应用

登录 EDAS 控制台。

在左侧导航栏中单击**应用管理**，进入应用列表页面。

在应用列表页面选择**地域**（Region）和**命名空间**（可选），然后在右上角单击**创建应用**。

在**应用基本信息**页面中设置应用的基本信息和参数，然后单击**下一步:应用配置**。

说明：设置完应用基本信息后，您也可以单击**创建空应用**，然后进入应用管理页面通过**应用扩容**或**添加实例**和**部署应用**进行发布。

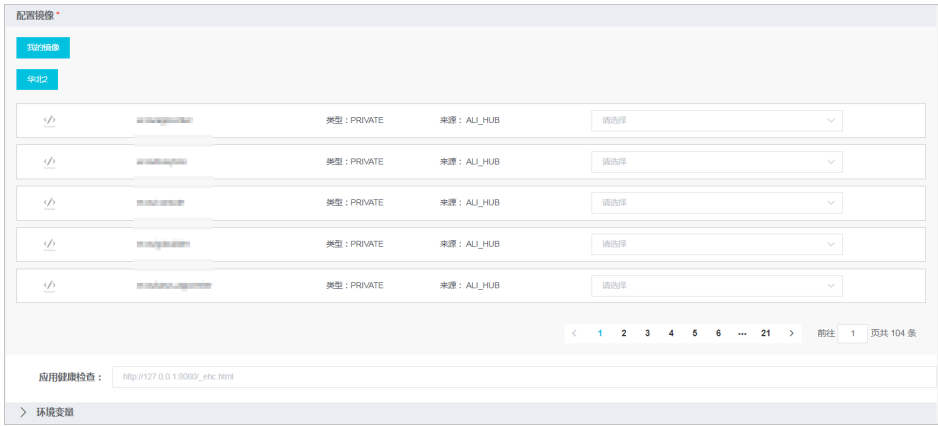
- **命名空间**：在左侧下拉选择框选择**地域**；在右侧下拉选择框选择**命名空间**，如果不做选择命名空间则设置为**默认**。
- **集群类型**：在左侧下拉列表中选择集群类型为 **Swarm 集群**，在右侧下拉列表选择一个具体的 Swarm 集群。
- **应用名称**：输入应用名称。
- **应用部署方式**：选择 Swarm 集群后，您可以选择**镜像**、**WAR 包部署**或**JAR 包部署**。此处选择**镜像**。
- **应用运行环境**：选择应用运行环境，例如 EDAS-Container 3.5.1 [支持FatJar部署]。
- **Java 环境**：选择 **Open JDK 8**或 **Open JDK 7**。
- **CPU 规格**：设置 CPU 规格。如果您在创建 Swarm 集群时设置了 **CPU 共享比例**为 1:2 或 1:4，则此处规则默认为**共享模式**；如果创建时未设置 **CPU 共享比例** 或设置 **CPU 共享比例**为 1:1，则此处您可以手动设置 CPU 规格。
- **规格**：设置内存规格。
- **应用描述**：填写应用的基本情况，输入的描述信息不超过 128 个字符。

在**应用配置**页面，添加**实例个数**并**选定实例列表**，按照页面指示进行配置。完成设置后单击**确认创建**。

- **请选择实例个数**：在右侧微调输入框内设置实例个数，实例个数由您所选择集群内的实例数决定。
 - 如果没有设置实例个数（默认为 0），单击**创建空应用**，可以创建一个空应用。后续再通过**应用扩容**进行实例的添加，通过**部署应用**完成应用部署。
 - 如果设置了大于 0 的实例个数，未对应用进行部署即单击**确认创建**，可以创建一个包含实例的空应用。之后可通过**部署应用**完成应用部署。

选定实例列表：在选择好实例个数后，此处会自动选定好实例。

是否立即部署：选择实例后才可点击打开。点击打开后按照界面进行配置。



配置镜像：选择镜像仓库，然后在镜像仓库右侧的下拉列表中选择具体镜像，选择完成后会在**配置镜像**右侧显示您所选择的镜像。

应用健康检查（可选）：设置应用健康检查的 URL。应用的健康会在容器启动后/运行时检查应用的状态是否正常，会根据应用的健康检查结果来执行服务路由。设置参考示例为 `http://127.0.0.1:8080/_etc.html`

- **环境变量（可选）**：展开**环境变量**。配置**变量名**和**变量值**。单击**添加**，可以设置多个环境变量。

说明：如果未勾选**是否立即部署**，在应用创建完成后，在应用详情页单击**部署应用**，也可以完成应用的部署。

应用创建可能需要几分钟，请您耐心等待。创建完成后可以前往应用详情页查看应用的基本信息和实例部署信息，在应用详情页中**实例部署信息**页签查看实例的运行状态，如果运行状态/时间为**正常运行**，说明应用发布成功。

部署应用和更新应用

如果在创建应用时只创建了一个空应用，则可以在应用详情页单击**部署应用**进行发布，如果在创建应用时已完成部署，则可通过**部署应用**来更新应用。

登录 EDAS 控制台。

在左侧导航栏中单击**应用管理**，在应用列表页面单击具体应用名称，进入**基本信息**页面。

单击**部署应用**，根据界面指示配置部署参数并单击**发布**。如果已部署过该应用，则**文件上传方式**可选方式由第一次部署的方法决定。（下面以镜像部署应用示例说明。）

部署应用

应用部署方式：☒ 镜像 ☐ WAR包部署 ☐ 历史版本

Docker 镜像地址：

registry.cn-beijing.aliyuncs.com/edas_test1/alpine

首次使用？

* 请填写版本：

请填写版本

用时间戳作为版本号

* 分组：

分组：默认分组

镜像所属地域：

cn-beijing

镜像：

edas_test1

alpine

* 镜像版本：

请选择

* 批次：

1批

* 分批方式：

自动

应用描述：

例如：“本次发布修复漏洞：...”。限制在128字符以内。

> 环境变量

发布

取消

- 应用部署方式：

- 全新部署的应用：您可选择**镜像**、**WAR 包部署**、**JAR 包部署**或**历史版本**来进行部署。
- 部署升级的应用：部署方式会根据您第一次部署的方式决定，选择您想重新部署的方式。

- **Docker 镜像地址**：此处会生成一个默认 Docker 镜像地址，不可设置。

- **请填写版本**：设置版本（如：1.2.0），建议不用时间戳作为版本号。

- **分组**：选择分组。

- **镜像所属地域**：选择镜像所存在的地域。

- **镜像**：在右侧下拉框内选择镜像仓库和用来部署的镜像。

- **镜像版本**：选择镜像的版本。

- **批次**：设置批次，如果选择2次以上的批次，需要设置**分批时间**。

- **分批方式**：选择自动。

- **应用描述**（可选）：描述可以为空，描述此次发布的目的。限制在128字符以内。

- **环境变量**（可选）：参考如何设置环境变量进行设置。

应用在重新部署后，会跳转到**变更详情**页面，查看部署流程和执行日志。部署流程执行完成后，执行状态变为**执行成功**则表示部署成功。

验证应用发布结果

应用发布后，你可以通过查看实例运行状态或登录负载均衡配置网址来验证应用已成功发布。

44

查看应用实例运行状态

在应用详情页中**实例部署信息**页签查看实例的运行状态。如果运行状态/时间为**运行正常**，说明应用发布成功。

配置公网负载均衡并访问应用

由于是在专有网络内创建发布的应用，如果没有特别配置，该应用没有公网 IP 地址。如果您的应用部署在多个 ECS 实例上，并且希望将您的应用对外开放，建议您配置公网负载均衡，以便将应用的访问流量根据转发策略分发到 ECS 实例中，增强应用的服务能力，提升应用的可用性。

在基本信息页面的应用设置区域，点击**负载均衡（公网）**右侧的**添加**。

在**添加 SLB 与应用的绑定**对话框中，设置负载均衡参数，然后点击**配置负载均衡**完成配置。

添加SLB与应用的绑定

开启SLB端口监听后，会自动在SLB上新增端口监听。
请勿在SLB控制台上删除该监听，否则将影响应用访问。

负载均衡(公网):

使用虚拟服务器组 ☒

虚拟服务器组（外网）:

新建虚拟服务器组

虚拟服务器组名称:

监听（外网）:

创建新监听: ☒

SLB 前端协议:

TCP

SLB 前端端口:

应用端口:

8080

配置负载均衡

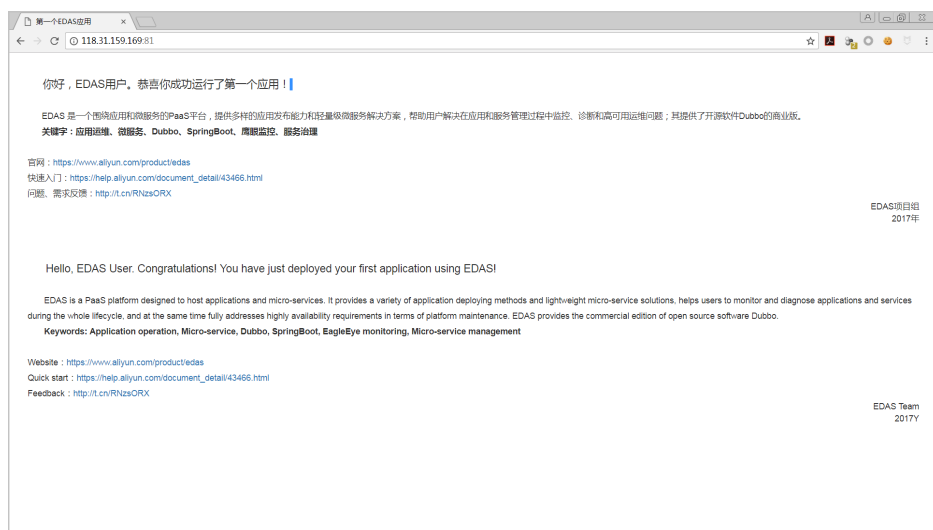
取消

- **负载均衡（公网）**：在右侧的下拉菜单中，根据实际需求，选择内网或公网的 SLB 地址。
- **使用虚拟服务器组**：虚拟服务器组是一组处理负载均衡分发的前端请求的 ECS 实例。不同的监听可以关联不同的虚拟服务器组，实现监听维度的请求转发。如果您勾选了**使用虚拟服务器组**，则需要配置虚拟服务器组参数。
- **虚拟服务器组名称**：如果您选择了**新建虚拟服务器组**，则需要在此处输入虚拟服务器组名称。系统会按照您输入的名称为您创建虚拟服务器组。
- **监听（外网）**：负载均衡服务监听规定了如何将请求转发给后端服务器。一个负载均衡实

例至少添加一个监听。您可以在监听右侧的下拉菜单中选择已创建的监听端口。如果您没有创建监听，单击**创建新监听**。请勿在服务均衡管理控制台上删除该监听，否则将影响应用访问。

- **SLB 前端协议**：默认为 TCP，不可配置。
- **SLB 前端端口**：输入 SLB 的前端端口，可自行设置端口数值。
- **应用端口**：默认为 8080，不可配置。

复制配置的 SLB IP 及端口，如 `118.31.159.169:81`，在浏览器的地址中粘贴并回车，即可进入应用的欢迎页面。



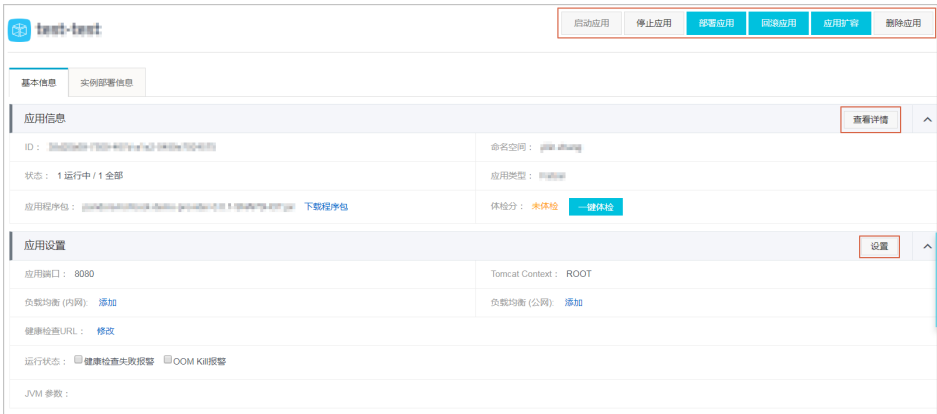
管理 Swarm 集群应用

登录 EDAS 控制台。

在左侧导航栏中单击**应用管理**，在应用列表页面单击具体 Swarm 集群应用名称，进入**基本信息**页面。

。

在该页面可以执行基本的管理任务，例如浏览、设置、停止、启动、扩容、回滚和删除。



使用工具部署应用

通过 toolkit-maven-plugin 插件自动化部署应用

在 ECS 集群或 Swarm 集群中，基于 HSF、Dubbo 或 Spring Cloud 框架开发的应用均可以通过 toolkit-maven-plugin 插件来自动化部署。

自动化部署应用

在您的打包工程的 pom.xml 文件中增加如下的插件依赖。

```
<build>
<plugins>
<plugin>
<groupId>com.alibaba.cloud</groupId>
<artifactId>toolkit-maven-plugin</artifactId>
<version>1.0.3</version>
</plugin>
</plugins>
</build>
```

在工程目录下创建一个文件格式为 yaml 的 .edas_config.yaml 文件。如果部署的工程为 Maven 的子模块，那么在子模块的目录下创建该文件。（关于多模块工程部署更多方式，请参考多模块工程部

署。)

```
env:
region_id: cn-beijing
app:
app_id: eb20****-e6ee-4f6d-a36f-5f6a5455****
```

以上配置项中，**region_id** 为应用机器所在的区域 ID，**app_id** 为应用 ID，以上的配置参数为示例参数，请替换成您的实际的应用参数。更多配置参数请参见[更多配置项](#)。

若您不知如何设置以上参数值，请根据如下步骤获取：

登录 EDAS 控制台。

在左侧导航栏中单击**应用管理**，进入应用列表页面点击您要进行的部署应用进入应用管理界面，单击**部署应用**。

单击**部署应用**页面下面的**生成 Maven 插件配置**获取参数信息。

部署应用

* 应用部署方式：☒ WAR包部署 ☐ 历史版本

* 文件上传方式：上传WAR包

下载样例工程

* 上传WAR包：

选择文件

* 版本：请输入版本。

用时间戳作为版本号

* 分组：默认分组

* 批次：1批

* 分批方式：自动

Java环境：JDK 8

描述：例如：“本次发布修复漏洞：...”。限制在128字符以内。

生成Maven插件配置

```
# 此配置根据您的输入自动生成，关于配置使用详情参考如下文档：
# //help.aliyun.com/document_detail/92242.html
---
env:
  region_id: cn-beijing
app:
  app_id: 

XXXXXXXXXX


  group_id: 

XXXXXXXXXX


  batch: 1
  batch_wait_time: 0
```

[了解更多快速部署方式](#)

部署

取消

新建一个账号文件，使用 yaml 格式配置 Access Key ID 和 Access Key Secret，Access Key ID 和 Access Key Secret 可登录阿里云用户信息管理查看，推荐使用 RAM 子账号（需已授予应用管理权限）来控制应用权限以提高应用安全。配置示例如下：

```
access_key_id: abc
access_key_secret: 1234567890
```

说明：以上配置中的 abc 和 1234567890 只是示例，请替换成您的 Access Key ID 和 Access Key Secret。该配置中 Access Key ID 和 Access Key Secret 只会用来生成请求签名，不会用于网络传输等其他用途。

进入您的工程目录（如果为 Maven 多模块则进入子模块目录），执行以下打包命令。

```
mvn clean package toolkit:deploy -Daccess_key_file={账号文件路径}
```

49

以上参数的意义为：

toolkit:deploy：使用 toolkit-maven-plugin 在打包成功后进行应用部署。只有添加此参数才会进行应用部署。

access_key_file：阿里云账号文件。（更多指定密钥对的方式可参考账号配置）

执行以上打包命令后，看到如下输出表示您已经成功使用 toolkit-maven-plugin 部署了应用。

```
[INFO] Sending deploy request to EDAS...
[INFO] Deploy request sent, changeOrderId is: 71548877-5bb3-4503-88e5-04761c0ac688
[INFO] Begin to trace change order: 71548877-5bb3-4503-88e5-04761c0ac688
[INFO] PipelineName:Batch: 1, PipelineId:9d7b8038-534c-4891-8f70-e63db6606297
[INFO] StageName:Workflow Start, StageId:cda21377-260d-43f0-9c82-98b55c3fd1ec
[INFO] ServiceStageName:Workflow Start, ServiceStageId:cda21377-260d-43f0-9c82-98b55c3fd1ec
[INFO] StageName:SLB Offline, StageId:1060b156-38b3-47aa-990d-5b5925256a57
[INFO] ServiceStageName:SLB Offline, ServiceStageId:1060b156-38b3-47aa-990d-5b5925256a57
[INFO] StageName:Deploy, StageId:035c7222-b84a-4c3a-bec2-3aeb3a5ac90e@co_virtual_stage
[INFO] InstanceName:nahai-20180920, InstanceIp:47.93.157.199(Public)<br>10.30.137.182(Inner)
[INFO] InstanceStageName:RPC Offline, InstanceStageId:3a0fbf0b-1dda-4368-8b67-b2e8bf7c7de7
[INFO] Waiting...
[INFO] Waiting...
[INFO] Waiting...
[INFO] Waiting...
[INFO] Waiting...
[INFO] Waiting...
[INFO] InstanceStageName:Stop Application, InstanceStageId:f8df6de7-adea-47c0-a2cd-6a18b49ebb2f
[INFO] InstanceStageName:Environment initial, InstanceStageId:033ca419-7d0e-4996-ba91-0b4e67877548
[INFO] InstanceStageName:Start Application, InstanceStageId:634e45bc-a411-429d-a549-b693794a41ac
[INFO] Waiting...
[INFO] Waiting...
[INFO] InstanceStageName:Health Check, InstanceStageId:77f179ee-f8ac-40ad-85a5-86cad41b1e4a
[INFO] Waiting...
[INFO] Waiting...
[INFO] Waiting...
[INFO] Waiting...
[INFO] StageName:SLB Online, StageId:035c7222-b84a-4c3a-bec2-3aeb3a5ac90e
[INFO] ServiceStageName:SLB Online, ServiceStageId:035c7222-b84a-4c3a-bec2-3aeb3a5ac90e
[INFO] StageName:Workflow Complete, StageId:510d5110-d430-4508-93c4-864dc506b9ae
[INFO] ServiceStageName:Workflow Complete, ServiceStageId:510d5110-d430-4508-93c4-864dc506b9ae
[INFO] Deploy application successfully!
[INFO] -----
[INFO] BUILD SUCCESS
```

更多配置项

部署应用的配置项可分成三大类：

- 基本环境（env）
- 应用配置（app）
- 存储配置（oss）

目前支持的配置项如下表所示：

类型	key	是否必须	说明
env	region_id	是	应用所在的区域ID
	endpoint	否	用于专有云设置 POP 网关接入点

app	app_id	是	应用ID
	package_version	否	部署包的版本。默认为 pom 文件的 version 加上当前机器构建的时间，格式如：“1.0 (2018-09-27 19:00:00)”。
	desc	否	部署的描述
	group_id	否	部署分组 ID。默认为所有分组。
	batch	否	部署分批。默认为 1 批，最大为 5 批。
	batch_wait_time	否	部署分批之间的等待时间，单位为分钟。默认不等待。
	stage_timeout	否	展示每个变更流程 stage 状态的超时时间，单位为分钟，默认为 5 分钟。如果同时设置了 batch_wait_time，那么此参数在计算时会自动加上 batch_wait_time。在运行时，如果某个 stage 等待时间超过此阈值，那么此插件会自动退出。
oss	region_id	否	目标存储桶所在的区域 ID。默认使用应用所在的区域 ID。
	bucket	否	目标存储桶名称。默认使用 EDAS 提供的免费 OSS 存储空间。若指定了 OSS 配置，则必须指定 bucket 参数，否则使用 EDAS 自动分配的免费 OSS 存储空间。
	key	否	应用包上传到 OSS 的自定义路径，默认使用 EDAS 提供的免费 OSS 存储空间。若使用指定的 OSS 存储，则可通过该参数指明包存储路径，同时可以使用以下变量来进行参数化的路径配置 {region_id}，{app_id}，{version}，例如：pkgs/petstore/{version}/store.war 该配置默认为 {region_id}/{app_id}/{version}
	access_key_id	否	应用包上传到 OSS 的自定义账号 ID。

	access_key_secret	否	应用包上传到 OSS 的自定义账号密钥。
--	-------------------	---	----------------------

配置示例一：指定分组和部署包的版本

例如我们在北京 Region 有一个 ID 为 eb20dc8a-e6ee-4f6d-a36f-5f6a545**** 的应用，希望部署的分组 ID 为 06923bb9-8c5f-4508-94d8-517b692f****，部署包版本为 1.2。那么配置如下：

```
env:
region_id: cn-beijing
app:
app_id: eb20dc8a-e6ee-4f6d-a36f-5f6a545****
package_version: 1.2
group_id: 06923bb9-8c5f-4508-94d8-517b692f****
```

配置示例二：指定 OSS 存储

例如我们希望部署一个 ID 为 eb20dc8a-e6ee-4f6d-a36f-5f6a545**** 的应用，并将部署包上传到自己在北京 Region 的名为 release-pkg 的存储桶，文件对象名为 my.war，OSS 账号 ID 为 ABC，OSS 账号密钥为 1234567890。那么配置如下：

```
env:
region_id: cn-beijing
app:
app_id: eb20dc8a-e6ee-4f6d-a36f-5f6a545****
oss:
region_id: cn-beijing
bucket: release-pkg
key: my.war
access_key_id: ABC
access_key_secret: 1234567890
```

指定配置文件

当未指定配置文件时，此插件会默认使用工程目录下的 .edas_config.yaml 文件。如果当前工程为一个 Maven 工程的子模块，那么默认配置文件应该存放在该当前工程目录下而不是整个 Maven 工程的根目录下。

您也可以通过设置参数 -Dedas_config=xxx 来指定配置文件。

如果既存在默认配置文件，也通过参数指定配置文件，那么此插件会使用参数指定的配置文件。

账号配置及账号优先级说明

当使用此插件部署应用时，您需提供部署应用的阿里云账号密钥对。目前此插件支持多种配置方式，当存在重复配置的情况时，优先级高的配置方式会覆盖优先级低的配置方式。优先级从高到低的配置方式罗列如下：

命令行指定 ak/sk 参数：您可以通过如下任一方式来指定 access_key_id/access_key_secret：

在使用 Maven 命令打包时，通过 `-Daccess_key_id=xx -Daccess_key_secret=xx` 来指定。

在 pom 中配置此插件时，配置 ak/sk 参数如下所示：

```
<plugin>
<groupId>com.alibaba.cloud</groupId>
<artifactId>toolkit-maven-plugin</artifactId>
<version>1.0.3</version>
<configuration>
<accessKeyId>abc</accessKeyId>
<accessKeySecret>1234567890</accessKeySecret>
</configuration>
</plugin>
```

命令行指定账号文件（推荐）：在使用 Maven 命令打包时，通过 `-Daccess_key_file={账号文件路径}` 来指定 yaml 格式的账号文件。例如：

```
access_key_id: abc
access_key_secret: 1234567890
```

使用默认的阿里云账号文件：如果没有通过以上两种方式指定账号，那么此插件会使用您曾经配置过的阿里云账号进行应用部署。

aliyuncli：如果您使用过最新的 aliyuncli 工具并且配置过阿里云账号，那么阿里云会在您当前 Home 目录下生成一个 .aliyuncli 目录，并在 .aliyuncli 目录下创建 credentials 文件来保存您的账号信息。以 Mac 系统为例，假如系统用户为 jack，那么会在 `/Users/jack/.aliyuncli/credentials` 文件中保存如下信息：

```
[default]
aliyun_access_key_secret = 1234567890
aliyun_access_key_id = abc
```

此插件会使用此账号文件作为部署应用的账号。

aliyun：如果您使用过老的 aliyun 工具并且配置过阿里云账号，那么 aliyun 工具会在您当前 Home

目录下生成一个 .aliyun 目录，并且在 .aliyun 目录下创建一个 config.json 文件来保存您的账号信息。以Mac系统为例，假如系统用户为jack，那么会在 /Users/jack/.aliyun/config.json 文件中保存账号信息，例如：

```
{
  "current": "",
  "profiles": [{
    "name": "default",
    "mode": "AK",
    "access_key_id": "",
    "access_key_secret": "",
    "sts_token": "",
    "ram_role_name": "",
    "ram_role_arn": "",
    "ram_session_name": "",
    "private_key": "",
    "key_pair_name": "",
    "expired_seconds": 0,
    "verified": "",
    "region_id": "",
    "output_format": "json",
    "language": "en",
    "site": "",
    "retry_timeout": 0,
    "retry_count": 0
  }, {
    "name": "",
    "mode": "AK",
    "access_key_id": "abc",
    "access_key_secret": "xxx",
    "sts_token": "",
    "ram_role_name": "",
    "ram_role_arn": "",
    "ram_session_name": "",
    "private_key": "",
    "key_pair_name": "",
    "expired_seconds": 0,
    "verified": "",
    "region_id": "cn-hangzhou",
    "output_format": "json",
    "language": "en",
    "site": "",
    "retry_timeout": 0,
    "retry_count": 0
  }],
  "meta_path": ""
}
```

- **系统环境变量**：最后，此插件会尝试通过系统环境变量来获取 access_key_id 和 access_key_secret 的值。即通过 Java 代码的 System.getenv("access_key_id") 和 System.getenv("access_key_secret") 来获取相应的值。

多模块工程部署

多模块工程是常见的项目组织形式，此 Maven 插件支持以下两种方式来部署您的多模块工程。

方式一：在父工程中执行打包及部署命令

此方式需要 toolkit-maven-plugin 版本为 1.0.3 及以上。

在父工程中执行 `toolkit:deploy` 命令时，增加 `-Ddeploy_artifacts` 参数来指定需要部署的子模块 Artifact ID。如果希望部署多个子模块，请以半角逗号分隔。

例如：一个 CarShop 工程存在以下示例的子模块：

- carshop
 - itemcenter-api
 - itemcenter
 - detail

如果想同时部署 itemcenter 和 detail 子模块，您可以在 carshop 目录下，执行以下命令：

```
mvn clean package toolkit:deploy -Ddeploy_artifacts=itemcenter,detail
```

默认情况下，此插件会根据 itemcenter 和 detail 模块下的 `.edas_config.yaml` 文件来分别部署对应的应用。您也可以通过增加 `-Dedas_config` 参数来指定配置文件，详见指定配置文件。

方式二：在子模块中执行打包及部署命令

此方式支持 toolkit-maven-plugin 所有版本。

首先，在父工程中执行 `install` 命令，将子模块依赖安装到本地的 Maven 仓库。然后，进入到需要部署的子模块目录下执行 `toolkit:deploy` 命令即可。

使用 CLI 快速部署 EDAS 应用

命令行界面 CLI (Command Line Interface)，是在图形用户界面得到普及之前使用最为广泛的用户界面，它通常不支持鼠标，用户通过键盘输入指令，计算机接收到指令后，予以执行。使用 CLI，您可以精确操控系统并可以高效而可靠地完成一些复杂操作。

阿里云命令行工具 (Alibaba Cloud Command Line Interface) 是在阿里云 GO SDK 之上构建的开源工具。阿里云 CLI 可直接访问 EDAS 的 Open API，请确保您已经开通了 EDAS 服务，并掌握了如何使用 SDK 调用 EDAS 的 API，调用 API 的具体操作方式请参见 SDK 接入指南。

在 EDAS 的 ECS 集群或 Swarm 集群中，基于 HSF、Dubbo 或 Spring Cloud 框架开发的应用均可以通过阿里云 CLI 来快速部署。

前提条件

完成本教程中的步骤需要完成以下前提任务：

1. 开通 EDAS 服务
2. 创建 ECS 实例

安装 CLI

阿里云 CLI 工具下载解压后即可使用，支持 Mac、Linux 和 Windows 平台（x64 版本）终端。

各操作系统的下载链接如下：

- Mac
- Linux
- Windows (64 bit)

解压后，您可以将 **aliyun** 文件移至 `/usr/local/bin` 目录下，或添加到 `$PATH` 环境变量中。

配置 CLI

在使用阿里云 CLI 前，您需要运行 `aliyun configure` 命令配置调用阿里云账号的 AccessKey、地域、语言等信息。

您可以在[安全信息管理](#)页面创建和查看您的 AccessKey，或者联系您的系统管理员获取 AccessKey。

```
$ aliyun configure
Configuring profile 'default' ...
Aliyun Access Key ID [None]: <Your AccessKey ID>
Aliyun Access Key Secret [None]: <Your AccessKey Secret>
Default Region Id [None]: cn-hangzhou
Default output format [json]: json
Default Language [zh]: zh
```

使用 CLI 快速创建应用

参考以下脚本，快速创建一个应用。

```
#!/bin/bash

# 希望部署的region
REGION="cn-beijing"
```

```

# ECS实例ID
ECS_ID="i-2z*****b6"
# ECS所在的VPCID
VPC_ID="vpc-t*****c"
# 命名空间名（若不存在会自动创建）
NAMESPACE="myNamespace"
# 集群名（会自动创建）
CLUSTER_NAME="myCluster"
# 应用名
APP_NAME="myApp"

# Step1: 创建命名空间
aliyun edas InsertOrUpdateRegion --RegionTag $REGION:$NAMESPACE --RegionName $NAMESPACE --region
$REGION --endpoint "edas.cn-beijing.aliyuncs.com" >> /dev/null

# Step2: 创建集群
CLUSTER_ID=`aliyun edas InsertCluster --ClusterName $CLUSTER_NAME --ClusterType 2 --NetworkMode 2 --VpcId
$VPC_ID --logicalRegionId $REGION:$NAMESPACE --region $REGION --endpoint "edas.cn-beijing.aliyuncs.com" |
sed -E 's/.*"ClusterId":("[a-z0-9-"]*)"/\1/g`

# Step3: 转化机器（需要花费一段时间）
aliyun edas TransformClusterMember --InstanceIds $ECS_ID --TargetClusterId $CLUSTER_ID --Password Hello1234
>> /dev/null
for i in `seq 300`
do
OUT=`aliyun edas ListClusterMembers --ClusterId $CLUSTER_ID | grep EcuId` && break
sleep 1
done
ECU_ID=`echo $OUT | sed -E 's/.*"EcuId":("[a-z0-9-"]*)"/\1/g`

# Step4: 创建应用
APP_ID=`aliyun edas InsertApplication --ApplicationName $APP_NAME --BuildPackId 51 --EcuInfo $ECU_ID --
ClusterId $CLUSTER_ID --logicalRegionId $REGION:$NAMESPACE | sed -E 's/.*"AppId":("[a-z0-9-"]*)"/\1/g`

printf "An application is created by CLI, App ID:"$APP_ID"\n"

```

使用 CLI 快速部署应用

使用阿里云 CLI 参考如下代码快速部署应用。

```

#!/bin/bash

# 待部署应用ID（需要提前创建）
APP_ID="87a6*****4d1"
# 应用分组ID
GROUP_ID="54b*****f27"
# 上传的OSS存储桶名（该桶需要公共可读）
OSS_BUCKET="eda*****mo"
# 安装包文件（由您的CI系统构建生成）
PACKAGE="hello-edas.war"

# Step1: 上传部署包到OSS
aliyun oss cp -f $PACKAGE oss://$OSS_BUCKET/$PACKAGE >> /dev/null
PKG_URL=`aliyun oss sign oss://$OSS_BUCKET/$PACKAGE|head -1`

```

```
# Step2: 发起部署请求
CO_ID=`aliyun edas DeployApplication --AppId $APP_ID --PackageVersion $VERSION --DeployType url --WarUrl
"${PKG_URL}" --GroupId $GROUP_ID | sed -E 's/.*"ChangeOrderId":"([a-z0-9-]*)".*/\1/g`

# Step3: 等待部署完成
for i in `seq 300`
do
STATUS=`aliyun edas GetChangeOrderInfo --ChangeOrderId $CO_ID | sed -E 's/.*"Status":(.).*/\1/g`
[ 2 = ${STATUS} ] && break
sleep 1
done
```

以上配置项中，APP_ID 和 GROUP_ID 为应用配置参数，以上代码内的参数均为示例参数，请替换成您的应用参数。

若您不知如何设置以上参数值，请根据如下步骤获取：

登录 EDAS 控制台。

在左侧导航栏中单击**应用管理**，进入应用列表页面点击您要进行的部署应用进入应用管理界面，单击**部署应用**。

在**部署应用**页面下方单击展开**生成 Maven 插件配置**获取参数信息。

部署应用

* 应用部署方式：☒ WAR包部署 ☐ 历史版本

* 文件上传方式：上传WAR包

下载样例工程

* 上传WAR包：

选择文件

* 版本：请输入版本。

用时间戳作为版本号

* 分组：默认分组

* 批次：1批

* 分批方式：自动

Java环境：JDK 8

描述：例如：“本次发布修复漏洞：...”。限制在128字符以内。

生成Maven插件配置

```
# 此配置根据您的输入自动生成，关于配置使用 详情参考如下文档：
# //help.aliyun.com/document_detail/92242.html
---
env:
  region_id: cn-beijing
app:
  app_id: 

XXXXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX


  group_id: 

XXXXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX


  batch: 1
  batch_wait_time: 0
```

[了解更多快速部署方式](#)

部署

取消

使用 Eclipse 插件快速部署应用

您在本地完成应用的开发、调试和测试后，可以使用在本地 IDE（现已支持 Eclipse、IntelliJ IDEA）中安装的 Cloud Toolkit 插件连接到云端部署环境并将应用快速部署到 EDAS 的 ECS 集群或 Swarm 集群中。本文档将向您介绍如何在 Eclipse 中安装 Cloud Toolkit，并使用 Cloud Toolkit 快速部署一个应用到 EDAS。

前提条件

- 下载并安装 JDK 1.8 或更高版本。
- 下载并安装适用于 Java EE 开发人员的 Eclipse IDE、4.5.0（代号：Mars）或更高版本。
- 已在 EDAS 控制台创建了应用并完成了首次部署，相关操作请参见在 ECS 集群创建应用或在 Swarm

集群创建应用。

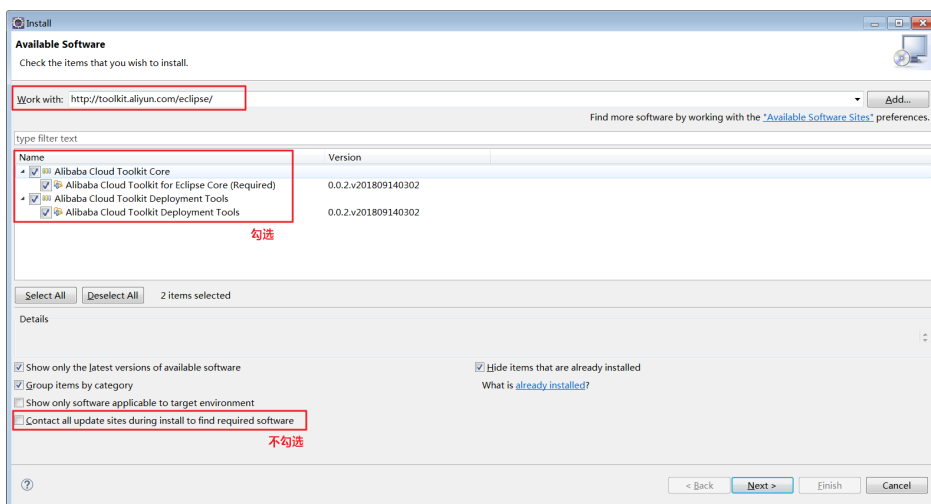
步骤一：安装 Cloud Toolkit

启动 Eclipse。

在菜单栏中选择 **Help > Install New Software**。

在 **Available Software** 对话框的 **Work with** 文本框中输入 Cloud Toolkit for Eclipse 的 URL <http://toolkit.aliyun.com/eclipse/>。

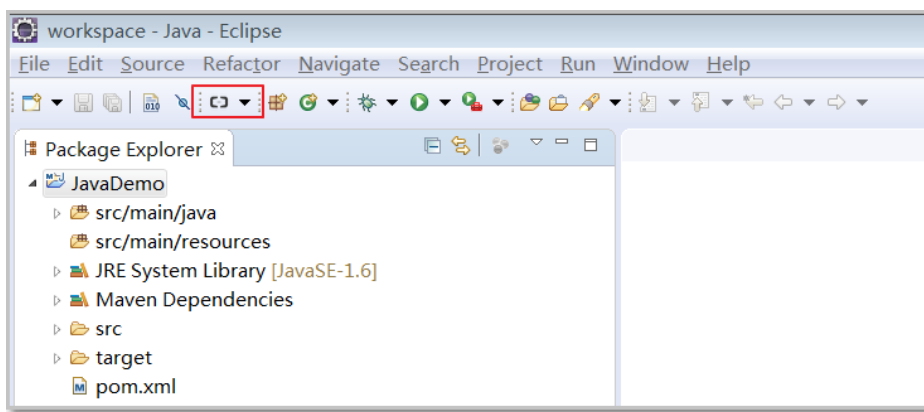
在下面的列表区域中勾选需要的组件 **Alibaba Cloud Toolkit Core** 和 **Alibaba Cloud Toolkit Deployment Tools**，并在下方 **Details** 区域中不勾选 **Connect all update sites during install to find required software**。完成组件选择之后，单击**Next**。



按照 Eclipse 安装页面的提示，完成后续安装步骤。

注意：安装过程中可能会提示没有数字签名，选择 **Install anyway** 即可。

Cloud Toolkit 插件安装完成后，重启 Eclipse，您可以在工具栏看到 Alibaba Cloud Toolkit 的图标。



步骤二：配置 Cloud Toolkit 账号

您需使用 Access Key ID 和 Access Key Secret 来配置 Cloud Toolkit 的账号。

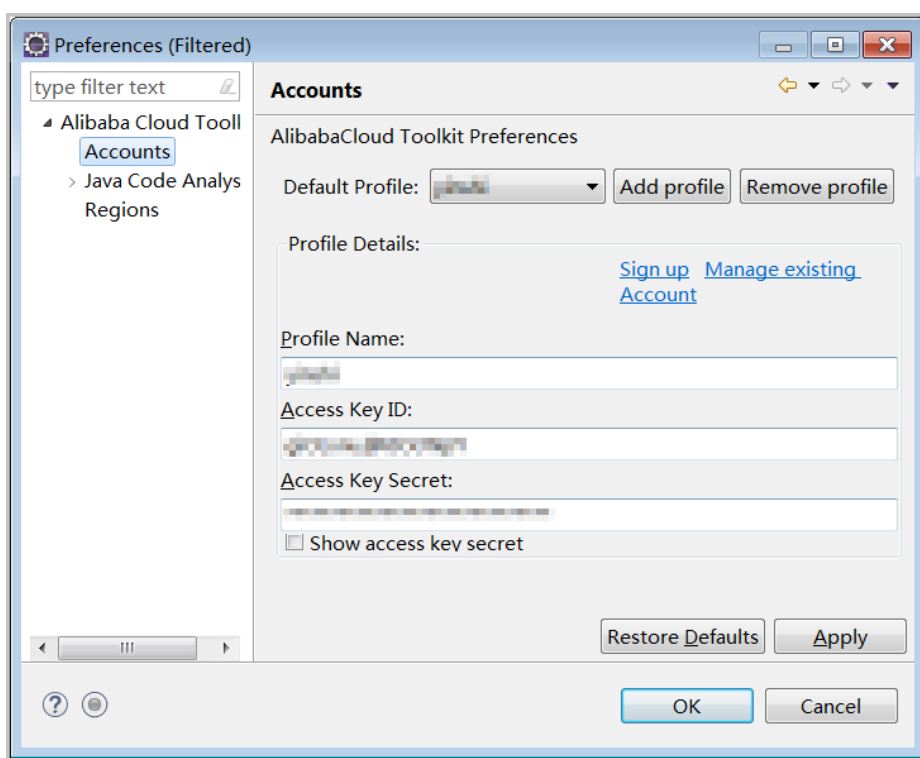
启动 Eclipse。

在工具栏单击 Alibaba Cloud Toolkit 图标右侧的下拉按钮，在下拉菜单中单击 **Alibaba Cloud Preference...**。

在 **Preference (Filtered)** 对话框的左侧导航栏中单击 **Accounts**。

在 **Accounts** 界面中设置 **Access Key ID** 和 **Access Key Secret**，然后单击 **OK**。

注意：如果您使用子账号的 Access Key ID 和 Access Key Secret，请确认该子账号至少拥有**部署应用**的权限，具体操作方式请参见 RAM 账号授权。



如果您已经注册过阿里云账号，在 **Accounts** 界面中单击 **Manage existing Account**，进入阿里云登录页面。用已有账号登录后，跳转至安全管理页面，获取 **Access Key ID** 和 **Access Key Secret**。

如果您还没有阿里云账号，在 **Accounts** 界面中单击 **Sign up**，进入阿里云账号注册页面，注册账号。注册完成后按照上述方式获取 **Access Key ID** 和 **Access Key Secret**。

说明：如果使用 EDAS 专有云企业版，还需要按以下步骤在 Cloud Toolkit 中配置 Endpoint。其中，Endpoint 请联系 EDAS 技术支持获取。

在 **Preference (Filtered)** 对话框的左侧导航栏中选择 **Appearance & Behavior**Endpoint。

在 **Endpoint** 界面中设置 Endpoint，配置完成后，单击 **Apply and Close**。

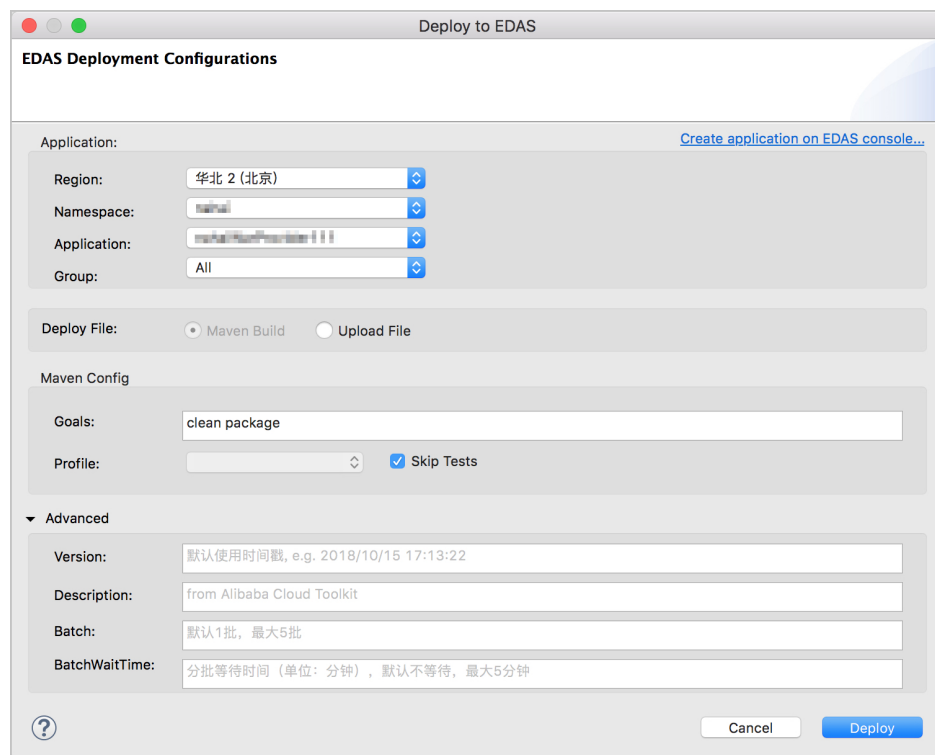
步骤三：将应用部署到 EDAS

目前支持使用 Cloud Toolkit 插件将应用通过 WAR 包 或 JAR 包部署到 EDAS。

在 Eclipse 界面左侧的 **Package Explorer** 中右键单击您的应用工程名，在弹出的下拉菜单中选择 **Alibaba Cloud > Deploy to EDAS....**

在 **Deploy to EDAS** 的运行配置页面，配置应用部署参数，然后单击 **Deploy**。

说明：如果您还没有在 EDAS 上创建应用，在对话框右上角单击 **Create application on EDAS console...**，跳转到 EDAS 控制台创建应用。



The screenshot shows the 'Deploy to EDAS' configuration window. It has a title bar with standard window controls. The main content area is titled 'EDAS Deployment Configurations'. Below this, there's a section for 'Application' with a link 'Create application on EDAS console...'. The 'Application' section includes dropdowns for 'Region' (华北 2 (北京)), 'Namespace' (default), 'Application' (default), and 'Group' (All). Below this is the 'Deploy File' section with radio buttons for 'Maven Build' (selected) and 'Upload File'. The 'Maven Config' section includes a text field for 'Goals' (clean package), a dropdown for 'Profile', and a checkbox for 'Skip Tests' (checked). The 'Advanced' section is expanded, showing fields for 'Version' (默认使用时间戳, e.g. 2018/10/15 17:13:22), 'Description' (from Alibaba Cloud Toolkit), 'Batch' (默认1批, 最大5批), and 'BatchWaitTime' (分批等待时间 (单位: 分钟), 默认不等待, 最大5分钟). At the bottom right, there are 'Cancel' and 'Deploy' buttons.

在配置页面中根据您的实际需求选择应用的 **Region**、**Namespace**、**Application** 和 **Group**。

- i. **Region**：应用所在地域。
- ii. **Namespace**：应用所在命名空间。
- iii. **Application**：应用名称。
- iv. **Group**：应用分组。

注意：如果在应用列表中获取不到应用，请参见应用列表获取不到应用进行操作排查。

设置构建方式。

- i. **Maven Build**：选择 Maven Build 方式来构建应用时，系统会默认添加一个 Maven 任务来构建部署包。
- ii. **Upload File**：选择 Upload File 方式来构建应用时，选择上传您的 WAR 包或者 JAR 包，然后进行部署。

设置应用的版本描述信息和分批部署信息。

- i. **Version**：部署版本。
- ii. **Description**：部署信息描述。

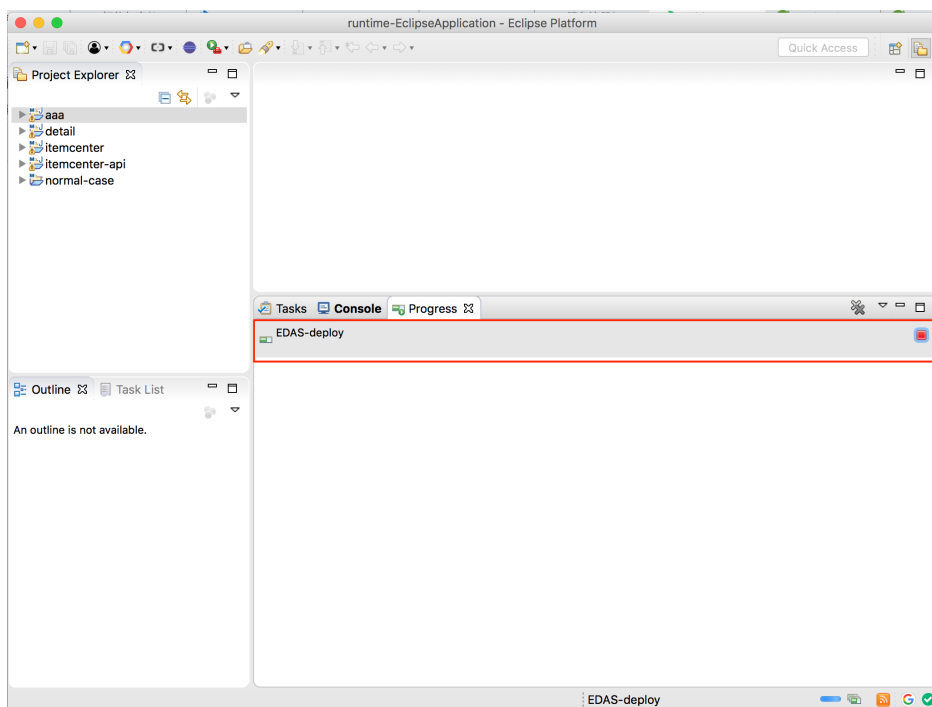
- iii. **Batch**：分批数。如果您的应用有多个分组，并且在部署时选择部署全部分组，那么将会自动按照分组粒度来分批，**Batch** 值不用设置。
- iv. **BatchWaitTime**：分批部署等待时间，单位为分钟。

注意：如果您的插件界面没有分批分批部署设置模块，请将您的插件升级至最新版本。

部署开始后，Eclipse 的 Console 区域会打印部署日志。您可以根据日志信息检查部署结果。

步骤四：终止 Cloud Toolkit 插件运行

在插件运行过程中，如果想停止插件运行，可以在 **Progress** 页面终止 **EDAS-deploy** 进程。



常见问题

应用列表获取不到应用

通常出现这种情况为使用子账号来部署应用，且子账号没有同步到 EDAS 系统或者没有进行正确授权，从而导致在应用列表下拉框中看不到应用。您可以通过 RAM 授权或 EDAS 子账号授权来确保子账号已经同步到 EDAS 并且得到授权。

RAM 授权

该授权方式可使子账号访问 EDAS 的所有资源。

1. 在 RAM 控制台左侧导航栏中选择**人员管理** > **用户**。

2. 在**用户**页面上找到需要授权的子用户，单击**操作**列中的**添加权限**。
3. 在**添加权限**面板的**选择权限**区域中，搜索 **AliyunEDASFullAccess** 权限，单击权限策略将其添加至右侧的**已选择**列表中，然后单击**确定**。
4. 在**添加权限**的**授权结果**页面上，查看授权信息摘要，并单击**完成**。
5. 使用主账号登录 EDAS 控制台，在左侧导航栏选择**系统管理** > **子账号**，单击**子账号**页面右上角的**同步子账号**。

EDAS 子账号授权

该授权方式可使子账号细粒度授权访问 EDAS 的资源。

1. 使用主账号登录 EDAS 控制台。
2. 在左侧导航栏选择**系统管理** > **角色**，单击**角色**页面右上角的**创建角色**。
3. 输入一个角色名称，在**可选权限**区域框中，选择**应用管理** > **应用列表** > **基本信息** > **部署应用**，单击**添加**将部署应用角色添加到**已选权限**，然后单击**确定**。
4. 在左侧导航栏选择**系统管理** > **子账号**，单击**子账号**页面右上角的**同步子账号**。
5. 选择需进行授权的子账号，在**操作**列单击**管理角色**，在左侧穿梭框中搜索并选择上面创建的角色，将该角色添加到右侧已选角色列表中，然后单击**确定**。
6. 选择需进行授权的子账号，在**操作**列单击**授权应用**，选择应用添加到右侧列表进行授权，然后单击**确定**。

问题反馈

如果您在使用 Cloud Toolkit 过程中有任何疑问，欢迎您扫描下面的二维码加入钉钉群进行反馈。



使用 IntelliJ IDEA 插件快速部署应用到 ECS/Swarm 集群

您在本地完成应用的开发、调试和测试后，可以在 IntelliJ IDEA 中安装 Cloud Toolkit 插件将应用快速部署到 EDAS 中。本文档将介绍如何在 IntelliJ IDEA 中安装 Cloud Toolkit，并快速部署应用到 EDAS 的 ECS 集群或 Swarm 集群。

前提条件

下载并安装 JDK 1.8 或更高版本。

下载并安装 IntelliJ IDEA (2018.3 或更高版本)。

说明：因 JetBrains 插件市场官方服务器在海外，如遇访问缓慢无法下载安装的，请加入文末交流群

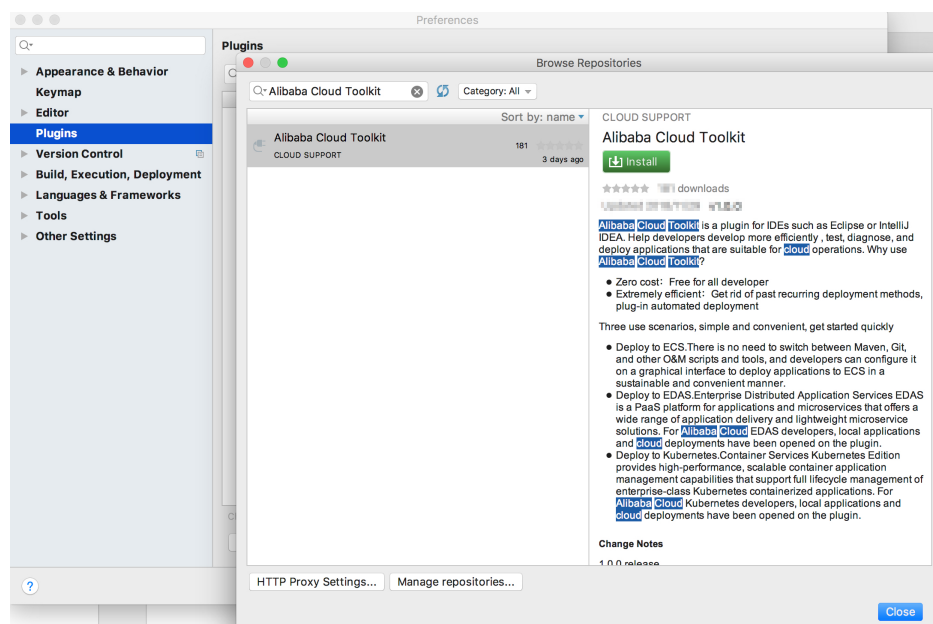
- , 向 Cloud Toolkit 产品运营获取离线包安装。
- 已在 EDAS 控制台创建了应用并完成了首次部署：
 - ECS 集群中创建应用请参见在 ECS 集群创建应用。
 - 在 Swarm 集群中创建应用请参见在 Swarm 集群创建应用。

步骤一：安装 Cloud Toolkit

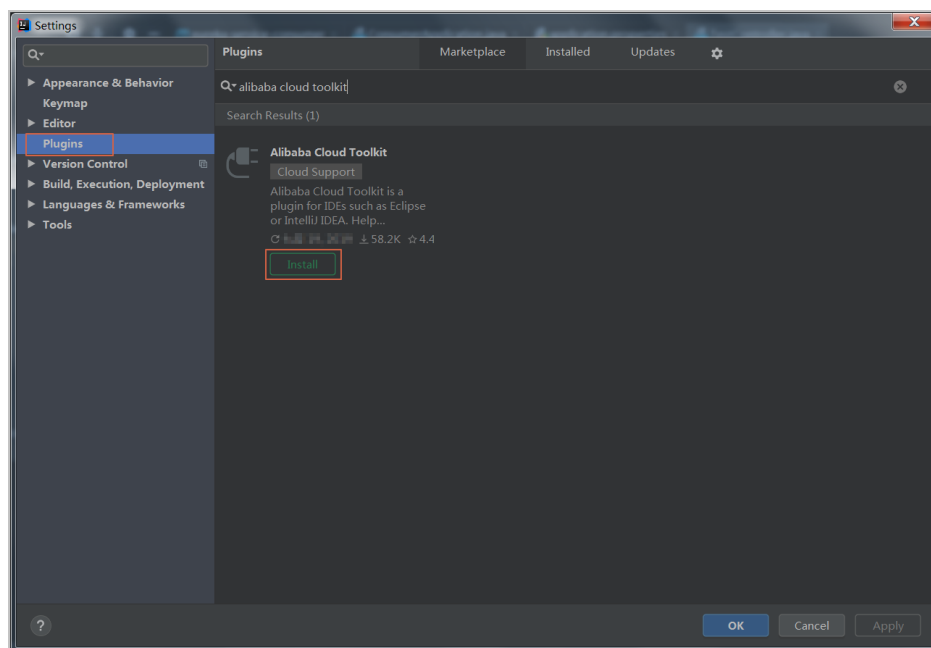
启动 IntelliJ IDEA。

在 IntelliJ IDEA 中安装最新版本的插件。如您之前已安装过旧版安装包，请升级至最新版本。

Mac 系统：进入 **Preference** 配置页面，选择左边的 **Plugins**，在右边的搜索框里输入 **Alibaba Cloud Toolkit**，并单击 **Install** 安装。



Windows 系统：进入 **Plugins** 选项，搜索 **Alibaba Cloud Toolkit**，并单击 **Install** 安装。



在 IntelliJ IDEA 中插件安装成功后，重启 IntelliJ IDEA，您可以在工具栏看到 Alibaba Cloud Toolkit 的图标 ()。

步骤二：配置 Cloud Toolkit 账号

在安装完 Alibaba Cloud Toolkit 后，您需使用 Access Key ID 和 Access Key Secret 来配置 Cloud Toolkit 的账号。

启动 IntelliJ IDEA。

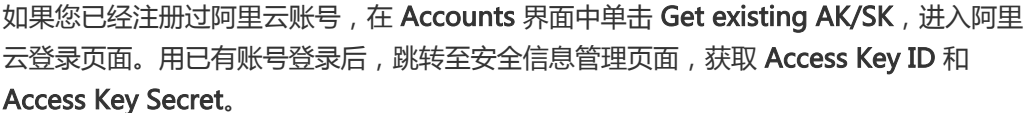
单击 Alibaba Cloud Toolkit 的图标 ()，在下拉列表中单击 **Preference...**，进入设置页面，在左侧导航栏选择 **Alibaba Cloud Toolkit > Accounts**。

在 **Accounts** 界面中设置 **Access Key ID** 和 **Access Key Secret**，然后单击 **OK**。

注意：

如果您使用**子账号**的 Access Key ID 和 Access Key Secret，请确认该子账号至少拥有**部署应用**的权限，具体操作方法请参见应用列表获取不到应用。

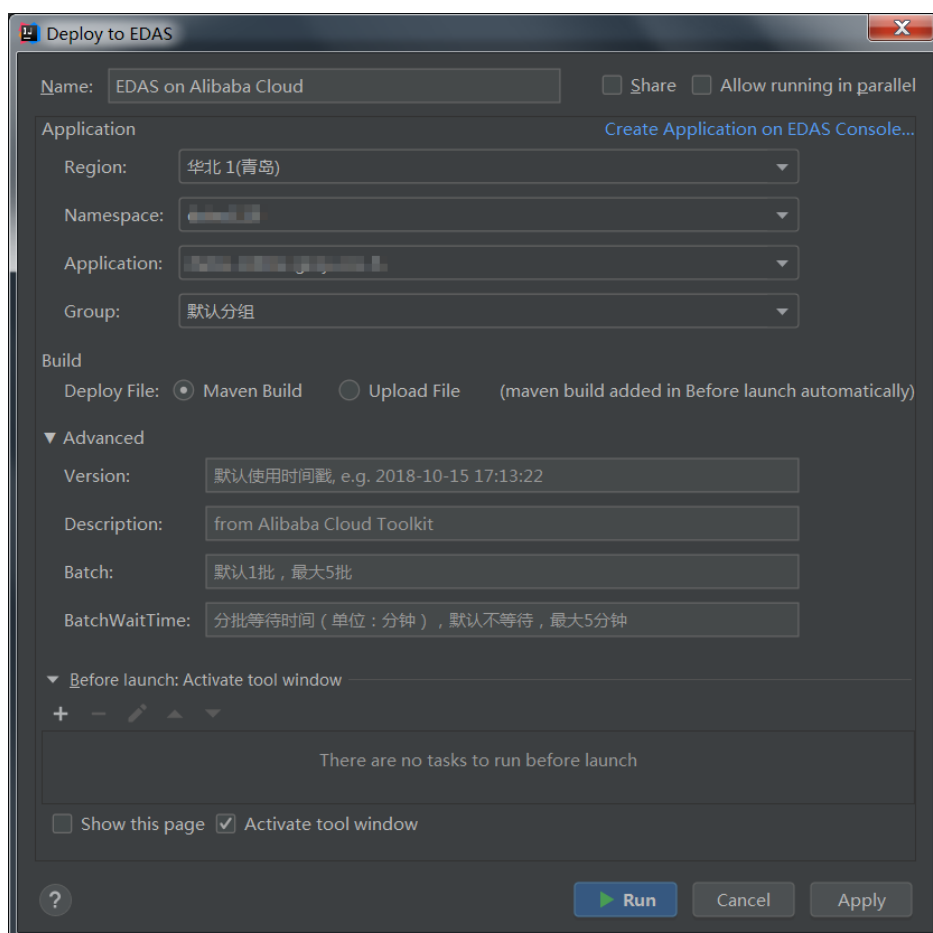
如果您是 **EDAS 专有云企业版用户**，还需配置 Endpoint 才能正常使用 Cloud Toolkit 功能。



步骤三：部署应用到 ECS 集群或 Swarm 集群

或在 IntelliJ IDEA 上打开 **Run/Debug configurations** 对话框，展开 **Templates** 选项，选择 **EDAS on Alibaba Cloud -> EDAS configuration factory**。

说明：如果您还没有在 EDAS 上创建应用，在对话框右上角单击 **Create application on EDAS console...**，跳转到 EDAS 控制台创建应用。



在配置页面中根据您的实际需求选择应用的 **Region**、**Namespace**、**Application** 和 **Group**。

- i. **Region**：应用所在地域。
- ii. **Namespace**：应用所在命名空间。
- iii. **Application**：应用名称。
- iv. **Group**：应用分组。

注意：如果在应用列表中获取不到应用，请参见应用列表获取不到应用进行操作排查。

设置构建方式。

- i. **Maven Build**：选择 Maven Build 方式来构建应用时，系统会默认添加一个 Maven 任务来构建部署包。
- ii. **Upload File**：选择 Upload File 方式来构建应用时，选择上传您的 WAR 包或者 JAR 包，然后进行部署。

设置应用的版本描述信息和分批部署信息。

- i. **Version**：部署版本。
- ii. **Description**：部署信息描述。

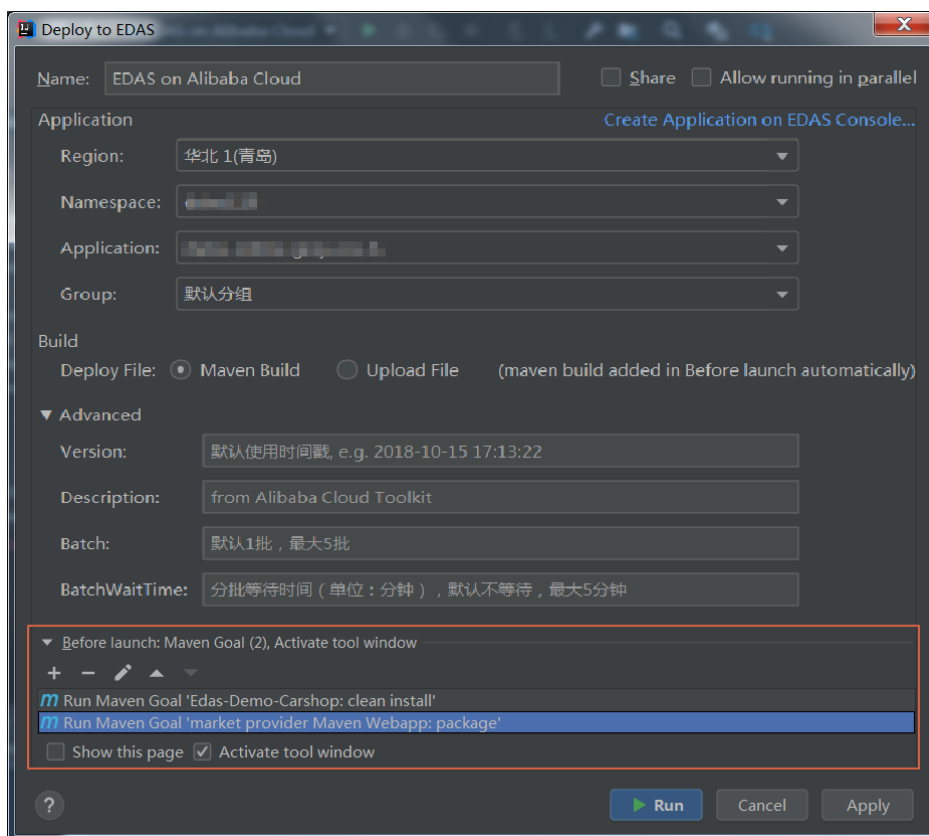
- iii. **Batch**：分批数。如果您的应用有多个分组，并且在部署时选择部署全部分组，那么将会自动按照分组粒度来分批，**Batch** 值不用设置。
- iv. **BatchWaitTime**：分批部署等待时间，单位为分钟。

注意：如果您的插件界面没有分批分批部署设置模块，请将您的插件升级至最新版本。

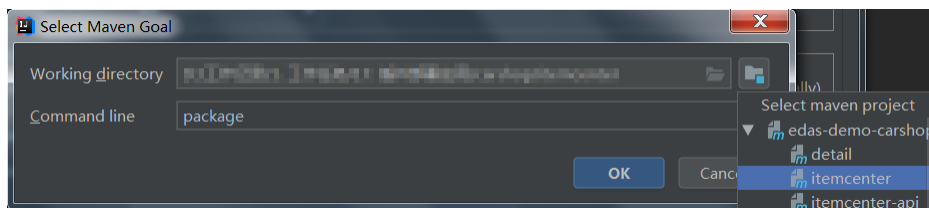
单击 **Run** 执行上面步骤的运行配置，IntelliJ IDEA 的 Console 区域会打印部署日志。您可以根据日志信息检查部署结果。

步骤四：管理 Maven 构建任务

在 IntelliJ IDEA 中安装的 Cloud Toolkit 内可以部署 Maven 的构建任务。您也可以在 Deploy to EDAS 的配置页面的 **Before launch** 区域来添加、删除、修改和移动 Maven 构建任务。



在添加 Maven 构建任务编辑框中，您可以单击右侧的文件夹按钮选择当前工程的所有可用模块，并在 **Command line** 中编辑构建命令。



部署多模块工程

实际工作中碰到的大部分 Maven 工程都是多模块的，各个项目模块可以独立开发，其中某些模块又可能会使用到其他的一些模块的功能，这样的项目工程就是多模块工程。

如果您的工程项目为 Maven 多模块工程并且想部署工程中的某子模块，那么需要保证 **EDAS Deployment Configurations** 页面中的 **Before launch** 中的 Maven 构建任务中最后一个任务为该子模块的构建任务。管理 Maven 构建任务的具体操作请参见[管理 Maven 构建任务](#)。

例如：一个 CarShop 工程存在以下示例的子模块：

- carshop
 - itemcenter-api
 - itemcenter
 - detail

其中 itemcenter 和 detail 为子模块，且都依赖于 itemcenter-api 模块，现在想部署 itemcenter 模块，应该怎么做？只需要在配置页面中的 **Before launch** 中增加如下两个 Maven 任务即可：

1. 增加一个在父工程 carshop 中执行 mvn clean install 的 Maven 任务；
2. 增加一个在子模块 itemcenter 中执行 mvn clean package Maven 任务。

常见问题

应用列表获取不到应用

通常出现这种情况为使用子账号来部署应用，且子账号没有同步到 EDAS 系统或者没有进行正确授权，从而导致在应用列表下拉框中看不到应用。您可以通过 RAM 授权或 EDAS 子账号授权来确保子账号已经同步到 EDAS 并且得到授权。

RAM 授权

该授权方式可使子账号访问 EDAS 的所有资源。

1. 在 RAM 控制台左侧导航栏中选择**人员管理** > **用户**。
2. 在**用户**页面上找到需要授权的子用户，单击**操作**列中的**添加权限**。
3. 在**添加权限**面板的**选择权限**区域中，搜索 **AliyunEDASFullAccess** 权限，单击权限策略将其添加至右侧的**已选择**列表中，然后单击**确定**。
4. 在**添加权限**的**授权结果**页面上，查看授权信息摘要，并单击**完成**。
5. 使用主账号登录 EDAS 控制台，在左侧导航栏选择**系统管理** > **子账号**，单击**子账号**页面右上角的**同步子账号**。

EDAS 子账号授权

该授权方式可使子账号细粒度授权访问 EDAS 的资源。

1. 使用主账号登录 EDAS 控制台。
2. 在左侧导航栏选择**系统管理** > **角色**，单击**角色**页面右上角的**创建角色**。
3. 输入一个角色名称，在**可选权限**区域框中，选择**应用管理** > **应用列表** > **基本信息** > **部署应用**，单击**添加**将部署应用角色添加到**已选权限**，然后单击**确定**。
4. 在左侧导航栏选择**系统管理** > **子账号**，单击**子账号**页面右上角的**同步子账号**。
5. 选择需进行授权的子账号，在**操作**列单击**管理角色**，在左侧穿梭框中搜索并选择上面创建的角色，将该角色添加到右侧已选角色列表中，然后单击**确定**。
6. 选择需进行授权的子账号，在**操作**列单击**授权应用**，选择应用添加到右侧列表进行授权，然后单击**确定**。

配置 Endpoint

说明：如果使用 EDAS 专有云企业版，还需要按以下步骤在 Cloud Toolkit 中配置 Endpoint。Endpoint 请联系 EDAS 技术支持获取。

在 **Preference (Filtered)** 对话框的左侧导航栏中选择 **Appearance & Behavior**Endpoint。

在 **Endpoint** 界面中设置 Endpoint，配置完成后，单击 **Apply and Close**。

问题反馈

如果您在使用 Cloud Toolkit 过程中有任何疑问，欢迎您扫描下面的二维码加入钉钉群进行反馈。



使用 Jenkins 创建持续集成

使用 Jenkins 可以构建 EDAS 应用的持续集成方案。该方案涉及下面的计算机语言或开发工具，阅读本文需要对下述的语言或工具有一定的理解。

工具	说明
Maven	Maven 是一个项目管理和构建的自动化工具。
Jenkins	Jenkins 是一个可扩展的持续集成引擎。
GitLab	GitLab 是一个利用 Ruby on Rails 开发的开源应用程序，实现一个自托管的 Git 项目仓库，可通过 Web 界面进行访问公开的或者私人项目。它拥有与 GitHub 类似的功能，能够浏览源代码，管理缺陷和注释。

注意：目前仅支持 ECS 集群和 Swarm 集群环境的应用部署。

准备工作

在开始持续集成之前，需要完成下述的准备工作。

获取阿里云的 Access Key ID 和 Access Key Secret。

使用已经开通了 EDAS 服务的主账号登录阿里云官网。

进入 Access Key 控制台，创建 Access Key ID 和 Access Key Secret。

在使用 Jenkins 自动部署应用之前，需要先在 EDAS 控制台中创建一个可以部署的应用。

登录 EDAS 控制台。

参考部署应用，部署应用。

如果已经创建了应用，请忽略此步。

在左侧导航栏中单击**应用管理**。找到您在上一步中创建的应用并单击进入详情页面，获取应用 ID 的字段内容。

使用 GitLab 托管您的代码。您可以自行搭建 Gitlab 或者使用阿里云 Code。

本文使用通过自行搭建的 GitLab 做演示，关于 Gitlab 的更多信息请参考 GitLab。

了解并使用 Jenkins。关于 Jenkins 的更多详细信息请参考 Jenkins 官网。

配置项目

参考通过 toolkit-maven-plugin 插件自动化部署应用修改项目配置，添加 toolkit-maven-plugin 及部署信息。您在修改完项目配置后，建议在本地使用 Maven 构建验证配置是否正确。

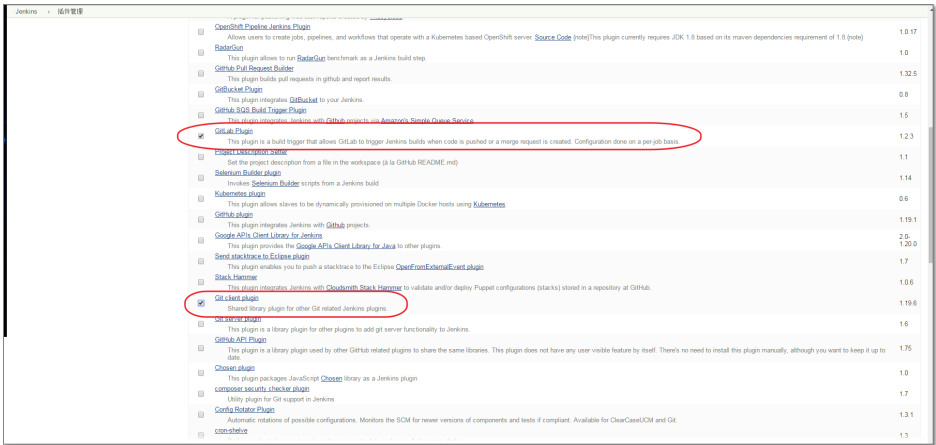
安装和配置 Jenkins

进入 Jenkins 官网下载安装 Jenkins（如已安装则请忽略此步）。

在 Jenkins 控制台的菜单栏中选择**系统管理** > **插件管理**，安装 Git 和 GitLab 插件。

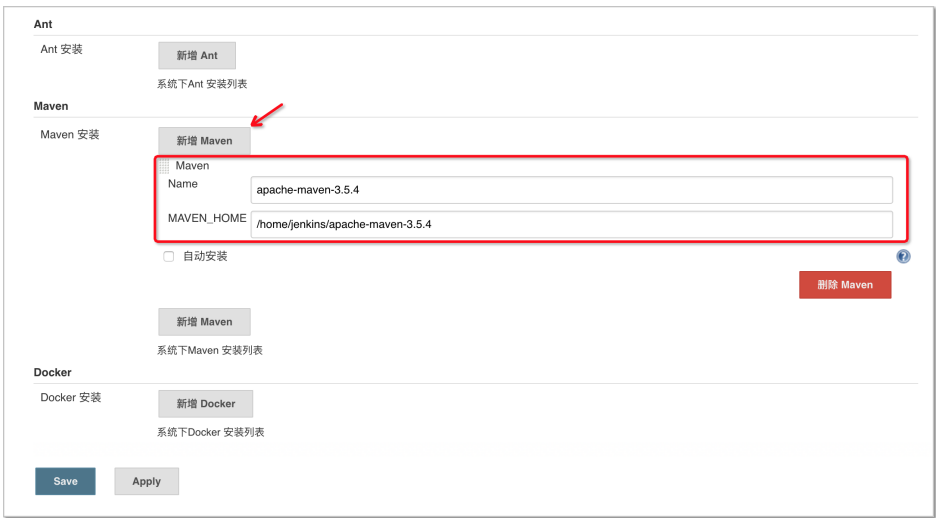
安装 GIT Client Plugin 和 GIT Plugin 插件可以帮助 Jenkins 拉取 Git 仓库中的代码。

安装 Gitlab Hook Plugin 插件可以帮助 Jenkins 在收到 Gitlab 发来的 Hook 后触发一次构建。



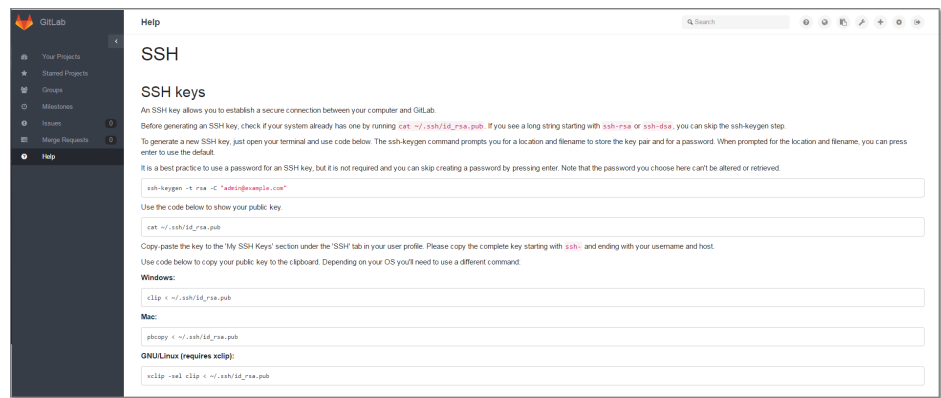
安装 Maven 构建工具，请参见 Maven 官网（如已安装请忽略）。

在 Jenkins 控制台的菜单栏中选择系统管理 > 全局工具配置，选择 Maven 版本名称并配置路径。

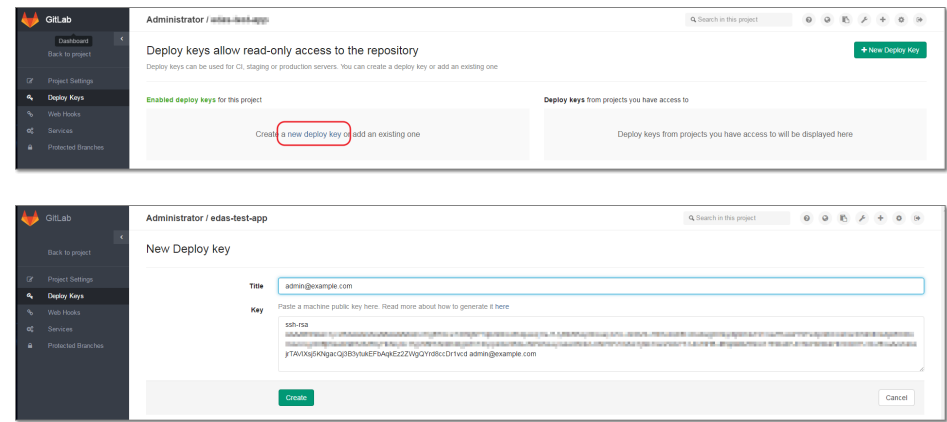


在 Jenkins 服务器上生成 SSH RSA 密钥对，并将公匙导入 GitLab，实现 Jenkins 拉取 GitLab 代码时自动认证。

参考 GitLab 文档，在 Jenkins 服务器运行 Jenkins 软件的用户下，生成 SSH RSA 密钥对。

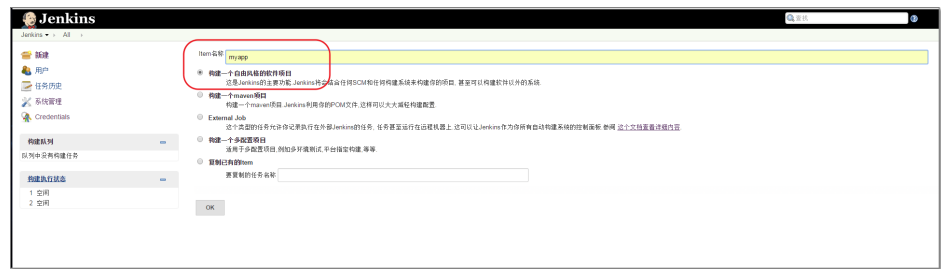


进入您的 GitLab 首页，在菜单栏选择 **Settings > Deploy Keys**。然后单击 **new deploy key** 添加 key，导入在 Jenkins 服务器上创建的 SSH RSA 公匙。



创建 Jenkins 任务。

在 Jenkins 首页左侧导航栏中单击**新建**，创建 Jenkins 任务，并选择**构建一个自由风格的软件项目**。

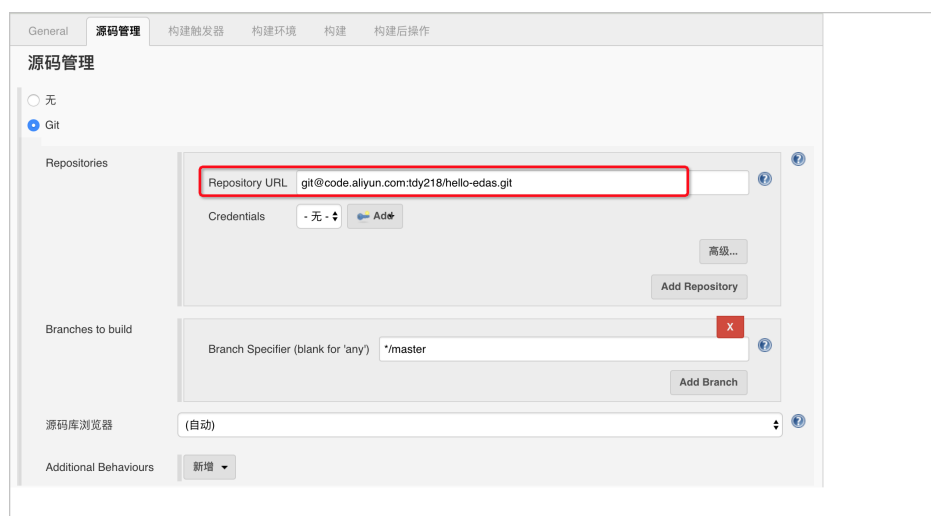


在 **源码管理** 页面中选择 **Git**，并设置相关参数。

Repository URL : 您的项目的 Git 协议地址。

Credentials : 安全凭证，选择**无**即可（前提是运行 Jenkins 软件的用户拥有 SSH

RSA 公匙已添加到该 Git 项目所在的 GitLab 中，否则这里会报错）。



单击**构建触发器**页签，勾选**轮询 SCM**。

单击**构建环境**页签，勾选 **Add timestamps to the Console Output**（为控制台输出的信息添加时间戳）。

单击**构建**页签，然后单击“**增加构建步骤**”。

在**调用顶层 Maven 目标**区域设置 **Maven 版本**和**目标**。如果您想部署多模块工程，请参见创建多模块工程的 Jenkins 任务。

Maven Version：单击该选项后面的下拉框，选择在**全局工具配置**里配置的 **Maven 版本名称**。

Goals：填入 clean package toolkit:deploy（如有其它参数，请根据实际情况填入）

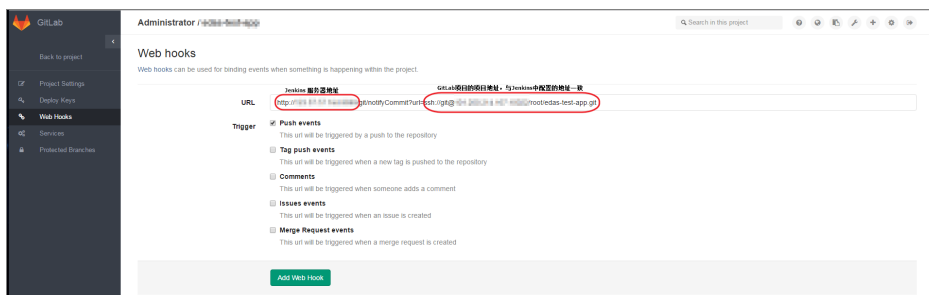


配置 Gitlab 的 Web Hook，实现自动构建

右键单击 GitLab 工程，然后选择 **Setting > Web Hooks**。

在 **Web Hooks** 页面的在 **URL 文本框**中输入`http://jenkins服务器地址:jenkins服务器监听端口/git/notifyCommit?url=本项目的git协议地址`

例如：`http://123.57.57.164:8080/git/notifyCommit?url=git@code.aliyun.com:tdy218/hello-edas.git`



图中表示的 Jenkins 服务器地址为您的 Jenkins 服务器的 Web 访问地址如 `http://123.57.57.164:8080`。

配置完成后，单击 **Test Hook**，进行测试。



配置正确后，提交变更到 GitLab

如果上述步骤配置正确，这次提交会触发一次 GitLab Hook。Jenkins 在接受到这个 Hook 后会构建您的 Maven 项目，并在构建结束时调用 EDAS POP API 脚本触发部署。

提交部署成功输出的日志信息（Build Number > 控制台输出）：

```
15:58:51 [INFO] Deploy application successfully!
15:58:51 [INFO] -----
15:58:51 [INFO] BUILD SUCCESS
15:58:51 [INFO] -----
15:58:51 [INFO] Total time: 24.330 s
15:58:51 [INFO] Finished at: 2018-12-25T15:58:51+08:00
15:58:51 [INFO] Final Memory: 23M/443M
15:58:51 [INFO] -----
15:58:51 Finished: SUCCESS
```

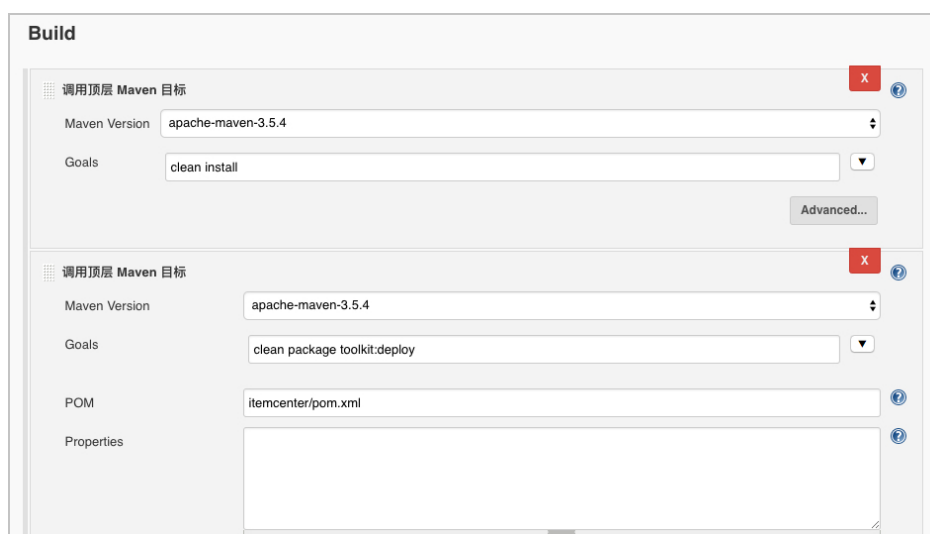
如果部署失败，可以登录 EDAS 控制台，在左侧导航栏中单击**应用管理 > 应用列表**，在应用列表页面单击具体应用名称，进入应用详情页面。在左侧导航栏单击**变更记录**来查看此次部署任务的执行过程。

创建多模块工程的 Jenkins 任务

创建多模块工程的 Jenkins 任务和安装和配置 Jenkins 第 5 步基本相同，只需要调整下调用顶层 Maven 目标。如果工程为多模块工程，想在 Jenkins 中部署子模块的话，那么需要在父模块中调用 mvn clean install 命令，然后在子模块中调用 mvn clean package toolkit:deploy 命令。以 Demo 工程为例，工程结构如下：

```
sh-3.2# tree -L 1 carshop
carshop
├── detail
├── itemcenter
├── itemcenter-api
└── pom.xml
```

其中，detail、itemcenter、itemcenter-api 为子模块，现在想部署 itemcenter 模块的话，那么需要在父工程中设置一个 clean install 构建目标，然后在 itemcenter 模块中设置 clean package toolkit:deploy 构建目标：



使用云效进行持续集成和发布

阿里云云效（下文简称云效）是阿里云一站式企业协同研发平台，源于阿里巴巴多年先进的管理理念和工程实践，提供从需求->开发->测试->发布->运维->运营端到端的协同服务和研发工具支撑。

云效支持多种开发模式，拥有部署回滚等功能。EDAS 提供了应用创建、部署、运维和日志监控等服务。云效对 EDAS 进行了集成，可以把在云效上打包的 War 包或者 Jar 包部署到 EDAS 中。云效与 EDAS 结合，可以很好的提供一站式持续交付体验。

您只需注册或使用一个已有的阿里云账号，即可通过云效进行应用程序的开发、调试及测试后，并通过云效轻松将应用程序部署到阿里云。

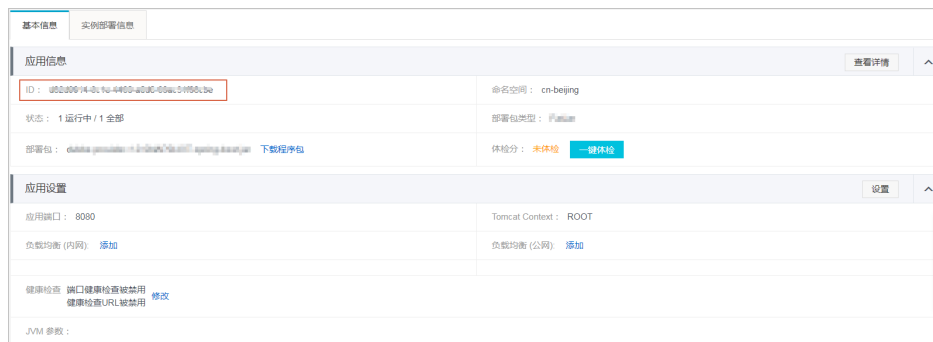
云效目前只支持基于 War 包和 Jar 包的部署，不支持镜像部署，故使用云效只能部署在 EDAS 的 ECS 集群和 Swarm 集群中的应用。本文档将向您介绍如何使用云效持续集成将代码快速生成部署包，并部署到 EDAS。

前提准备

在使用云效在 EDAS 上部署应用前，您需要先在 EDAS 上创建一个应用并获取应用的 ID 信息。

1. 参照创建 ECS 集群应用或创建 Swarm 集群应用创建一个应用。

登录EDAS 控制台，在左侧导航栏单击**应用管理**，选择您所创建的应用，单击**应用名称**进入**应用基本信息**页面，查看并复制应用的 ID。



使用云效配置持续集成

1. 使用阿里云账号，登录阿里云云效，进入**应用功能**菜单。

在**我的应用**页面的左上角单击**注册应用**，进入**基本信息**页签内设置应用的基本信息。完成设置后单击**下一步**。



- **应用名**：应用名称可以由小写字母、数字和中划线（-）组成，只能以小写字母开头。
- **所属项目**：选择应用归属的项目，您也可以单击右侧的**点击新建项目**创建一个公开或者私密的项目。
- **开发模式**：选择自由模式或分支模式。

在**配置代码库**页签内配置代码库，设置完成后单击**下一步**。

一站式研发解决方案

1 基本信息 2 配置代码库 3 应用模板 4 构建配置 5 应用信息预览 6 完成

代码源: Aliyun

代码库: ☒ 新建 ☐ 关联已有

Git组/库: /

您只能选择属于本企业且您是Master或Owner的Git组。 [点击新建Git组](#)

- 代码源：您可选择 Aliyun 或码云托管您的代码。
- 代码库：选择新建或关联已有。
 - 新建：您需新建或者选择 Git 组/库。
 - 关联已有：您需输入已有的 Git 库地址的 URL。

在应用模版页签中，您可以先勾选编程语言和部署选项筛选应用模板，在筛选结果中选择基于 Java 开发的应用部署到 EDAS，然后单击下一步。

一站式研发解决方案

1 基本信息 2 配置代码库 3 应用模板 4 构建配置 5 应用信息预览 6 完成

选择应用模板

编程语言

☒ Java

☐ NodeJS

☐ PHP

☐ Python

☐ Go

☐ 其它语言

部署选项

☐ RDC脚本部署

☒ EDAS部署

☐ 阿里云容器服务

☐ Kubernetes部署

spring mvc

代码模板

基于spring mvc的java应用
部署到EDAS

spring-boot

代码模板

基于spring-boot的java应用
部署到EDAS

dingding-isv

代码模板

基于dingding-isv的java应用
部署到EDAS

miniapp-cloud-isv

代码模板

基于miniapp-cloud-isv的java应用
部署到EDAS

java

代码模板

基于java开发的应用
部署到EDAS

共5条 1/1

在构建配置页签内设置生成的部署包的信息，设置完成后单击下一步。

一站式研发解决方案

1 基本信息 2 配置代码库 3 应用模板 4 构建配置 5 应用信息预览 6 完成

语言版本: 1.7

产物路径: target/test.war

Docker构建: ☐ 根据源码库中的Dockerfile自动构建镜像并推送到容器镜像仓库

自定义构建镜像: ☐ 用户可以使用自定义镜像作为项目的构建时环境

构建配置: 预览doc-test.release文件,其中定义了构建的规则,详情见文档

```
# 请参考 https://help.aliyun.com/document_detail/59283.html 了解更多关于Release文件的编写方式
# 构建源语言类型
code.language=oracle-jdk1.7
# 构建打包使用的打包文件
build.output=target/test.war
# 该配置项会避免云效对build.output中的产物再进行压缩（EDAS部署需要此配置项）
build.output.notgz=true
```

- **语言版本**:选择 JAVA 的语言版本，您可以选择 1.7，1.8 或 1.9。
- **产物路径**：从代码库根目录算起，需打成 tgz 压缩包的内容的相对路径，可以是文件（比如 target/xxx.war）、目录下全部文件(如 target/*，此时解压后无该目录名）或目录(target)；已选择初始化模板的情况下为默认值，如需修改可下载代码库后修改构建配置文件设置部署包的格式和路径。
- **Docker 构建**：在 EDAS 中暂时还不支持镜像部署，无需勾选该选项。
- **自定义构建镜像**：在 EDAS 中暂时还不支持镜像部署，无需勾选该选项。

在**应用信息预览**页签内查看应用部署信息后单击**确定创建应用**。

一站式研发解决方案

1 基本信息 2 配置代码库 3 应用模板 4 构建配置 5 应用信息预览 6 完成

基础信息

应用名称: 应用名称 所属项目: 所属项目

开发模式: 主干模式

代码库配置

代码源: 阿里云 代码库地址: 代码库地址

应用模板

模板名字: 基于java开发的应用

构建配置

构建配置:
预览doc-test.release文件,其中定义了构建的规则,详情见文档

```
# 请参考 https://help.aliyun.com/document_detail/59293.html 了解更多关于release文件的编写方式
# 构建源语言类型
code.language=oracle-jdk1.8
# 构建打包使用的打包文件
build.output=target/test.war
# 该配置项会避免云效对build.output中的产物再进行压缩（EDAS部署需要此配置项）
build.output.notgz=true
```

上一步 确定创建应用

在**完成**页签内可以查看应用的创建进度，然后单击**进入应用列表**，您将看到云效创建的所有应用列表。

说明：如果您前面选择了新建代码库，该页面还会先显示您创建代码库的进度。

一站式研发解决方案

1 基本信息 2 配置代码库 3 应用模板 4 构建配置 5 应用信息预览 6 完成

创建代码库: 创建代码库

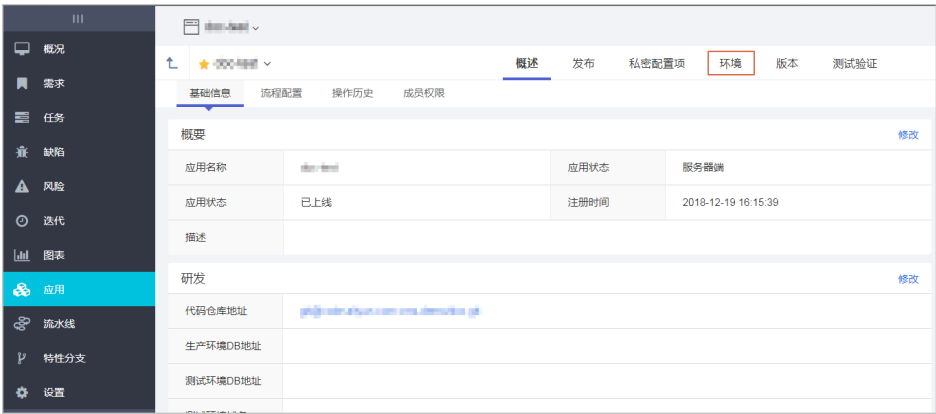
创建应用: 创建应用 45%

上一步 进入应用列表

云效上关联 EDAS 应用并配置部署策略

1. 使用阿里云账号，登录阿里云云效，进入**我的应用**功列表页面。选择您所创建的应用，并单击**应用名称**进入应用基础信息页面。

在应用上方菜单栏单击**环境**选项。



选择一个部署环境，然后在该环境下单击**部署配置**。



在**部署配置**页面配置应用信息。在**修改部署方式**选择框下面单击**切换至关联已有应用**。然后在**EDAS 应用 ID**设置框内输入您要部署的 EDAS 的应用 ID，完成输入后单击**保存**。

说明：应用 ID 的获取方式参考前提准备的步骤 2。



等待关联过程完成，会在**部署配置**出现关联应用的设置信息和**部署策略设置**，您可以配置应用的部署策略。设置**分组、批次、分批方式和分批等待时间**后单击**保存部署配置**。

环境 / 日常环境 / 部署配置

主干部署

部署历史

部署策略

部署配置

云服务

修改部署方式

EDAS部署

*区域

China North 1 (Beijing)

?

*命名空间

开发组

?

*应用名

demo-123456789

?

*应用类型

微服务应用

解除关联

部署策略配置

*分组

分组: 默认分组

?

*批次

2批

?

*分批方式

自动

?

*分批等待时间

不等待

?

保存部署配置

单击左侧导航栏的**应用**进入应用列表，选择您所创建的应用，在操作列单击**发布**，进入到**流水线**页面。

应用列表

ID	应用名称	应用类型	状态	Owner	AppOps	操作
demo-123456789	demo-123456789	微服务应用	已上线	demo-123456789	demo-123456789	发布 应用删除

在应用的**构建**流程内，单击**运行流水线**按钮，触发应用的构建发布流程。应用发布时会在流水线页面显示发布过程和相关日志，发布成功后您可以登录 EDAS 控制台查看应用的部署情况。

注意：如果应用发布失败，您可以参考 EDAS 操作常见问题进行问题排查。

使用 IntelliJ IDEA 插件快速部署应用到容器服务 K8s 集群

您在本地完成应用的开发、调试和测试后，可以在 IntelliJ IDEA 中安装 Cloud Toolkit 插件将应用快速部署到 EDAS 中。本文档将介绍如何在 IntelliJ IDEA 中安装 Cloud Toolkit，并快速部署应用到 EDAS。

前提条件

下载并安装 JDK 1.8 或更高版本。

下载并安装 IntelliJ IDEA (2018.3 或更高版本)。

说明：因 JetBrains 插件市场官方服务器在海外，如遇访问缓慢无法下载安装的，请加入文末交流群，向 Cloud Toolkit 产品运营获取离线包安装。

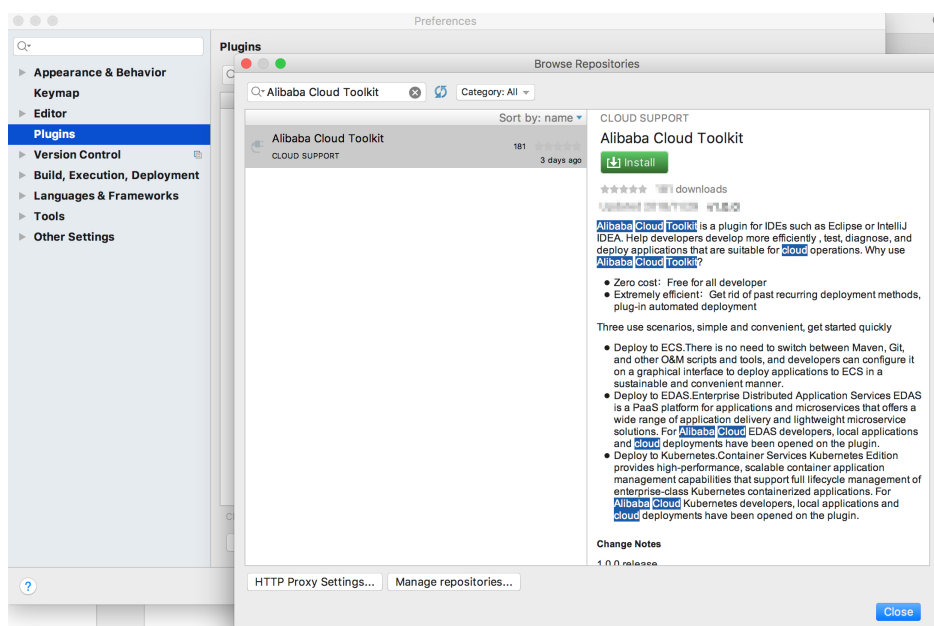
- 已在 EDAS 控制台创建了应用并完成了首次部署：
 - ECS 集群中创建应用请参见在 ECS 集群创建应用。
 - 在容器服务 K8S 集群中创建应用请参见在容器服务集群中创建应用。
 - 在 Swarm 集群中创建应用请参见在 Swarm 集群创建应用。

步骤一：安装 Cloud Toolkit

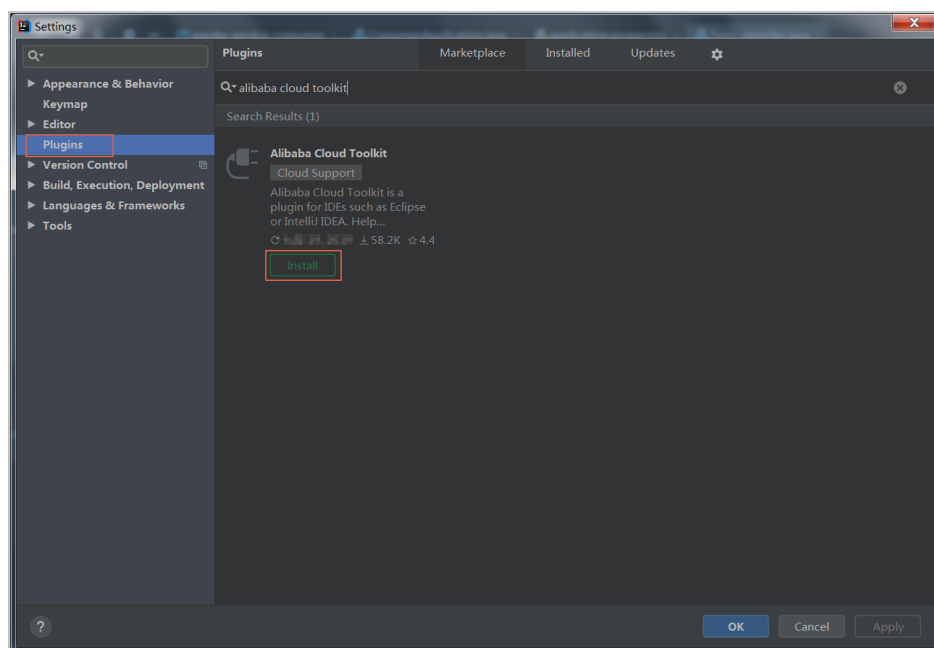
启动 IntelliJ IDEA。

在 IntelliJ IDEA 中安装最新版本的插件。如您之前已安装过旧版安装包，请升级至最新版本。

Mac 系统：进入 **Preference** 配置页面，选择左边的 **Plugins**，在右边的搜索框里输入 **Alibaba Cloud Toolkit**，并单击 **Install** 安装。



Windows 系统：进入 Plugins 选项，搜索 Alibaba Cloud Toolkit，并单击 Install 安装。



在 IntelliJ IDEA 中插件安装成功后，重启 IntelliJ IDEA，您可以在工具栏看到 Alibaba Cloud Toolkit 的图标 ()。

步骤二：配置 Cloud Toolkit 账号

在安装完 Alibaba Cloud Toolkit 后，您需使用 Access Key ID 和 Access Key Secret 来配置 Cloud Toolkit 的账号。

启动 IntelliJ IDEA。

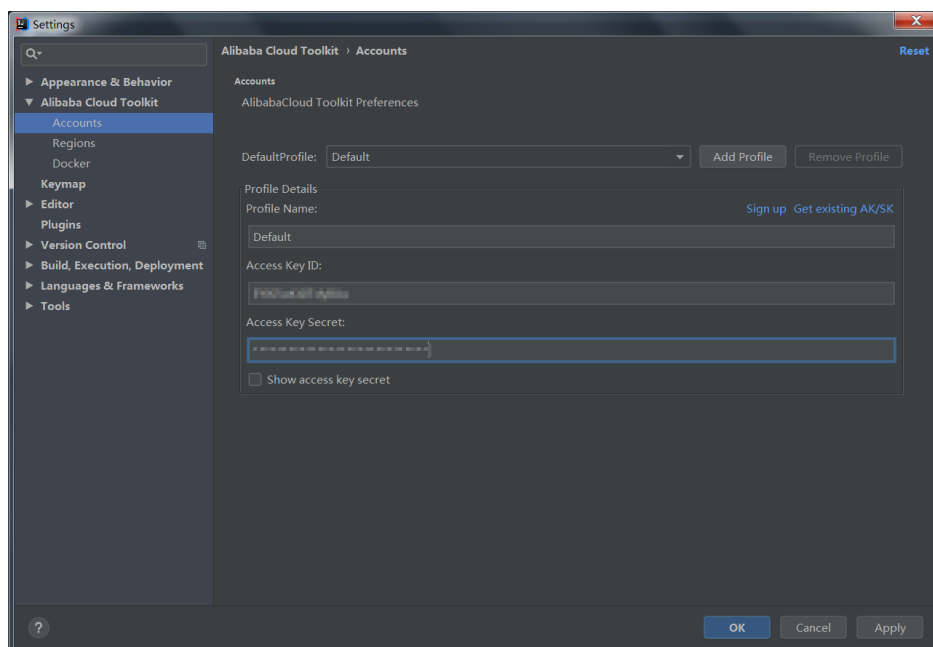
单击 Alibaba Cloud Toolkit 的图标 ()，在下拉列表中单击 **Preference...**，进入设置页面，在左侧导航栏选择 **Alibaba Cloud Toolkit > Accounts**。

在 **Accounts** 界面中设置 **Access Key ID** 和 **Access Key Secret**，然后单击 **OK**。

注意：

如果您使用**子账号**的 Access Key ID 和 Access Key Secret，请确认该子账号至少拥有**部署应用**的权限，具体操作方法请参见应用列表获取不到应用。

如果您是 **EDAS 专有云企业版用户**，还需配置 **Endpoint** 才能正常使用 Cloud Toolkit 功能。



如果您已经注册过阿里云账号，在 **Accounts** 界面中单击 **Get existing AK/SK**，进入阿里云登录页面。用已有账号登录后，跳转至安全信息管理页面，获取 **Access Key ID** 和 **Access Key Secret**。

如果您还没有阿里云账号，在 **Accounts** 界面中单击 **Sign up**，进入阿里云账号注册页面，注册账号。注册完成后按照上述方式获取 **Access Key ID** 和 **Access Key Secret**。

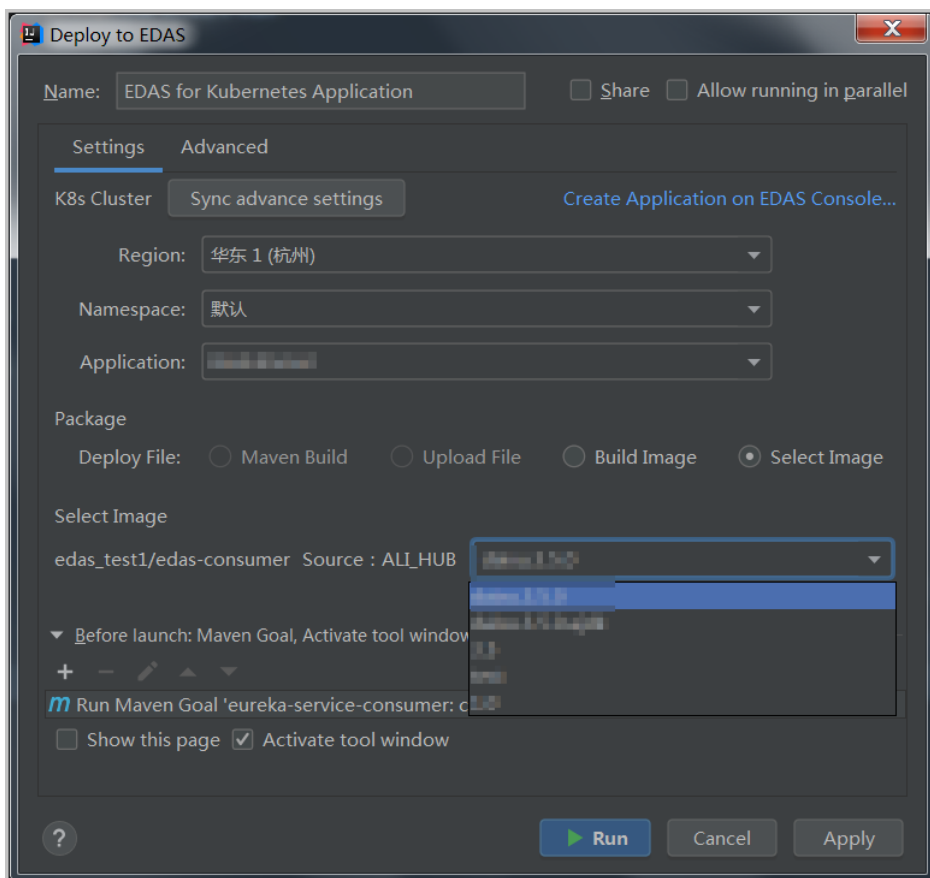
步骤三：部署应用到容器服务 K8s 集群

在 IntelliJ IDEA 上单击 Cloud Toolkit 的图标 ()，在下拉列表中选择 **Deploy to EDAS -> EDAS for Kubernetes Application**。

或在 IntelliJ IDEA 上打开 **Run/Debug configurations** 对话框，展开 **Templates** 选项，选择 **EDAS on Alibaba Cloud -> EDAS for Kubernetes Application**。

在 **Deploy to EDAS** 对话框配置应用部署参数。

说明：如果您还没有在 EDAS 上创建应用，在对话框右上角单击 **Create application on EDAS console...**，跳转到 EDAS 控制台创建应用。



在配置页面中根据您的实际需求选择应用的 **Region**、**Namespace** 和 **Application**。

- i. **Region**：应用所在地域。
- ii. **Namespace**：应用所在命名空间。
- iii. **Application**：应用名称。

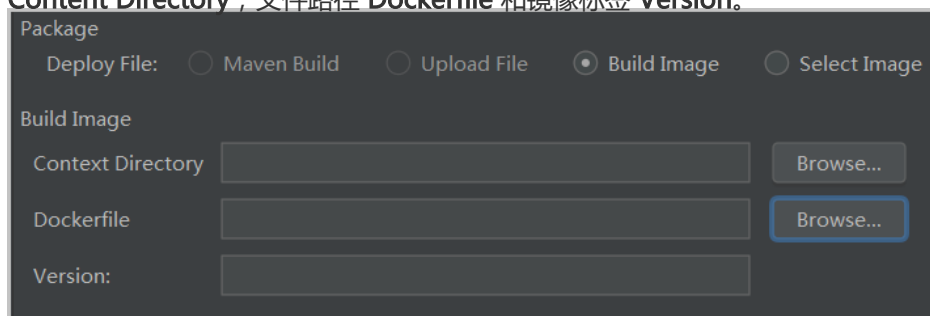
注意：如果在应用列表中获取不到应用，请参见应用列表获取不到应用进行操作排查。

设置部署方式，可选的部署方式由您在控制台第一次部署的方式决定。

- i. **Maven Build**：选择 Maven Build 方式来构建应用时，系统会默认添加一个

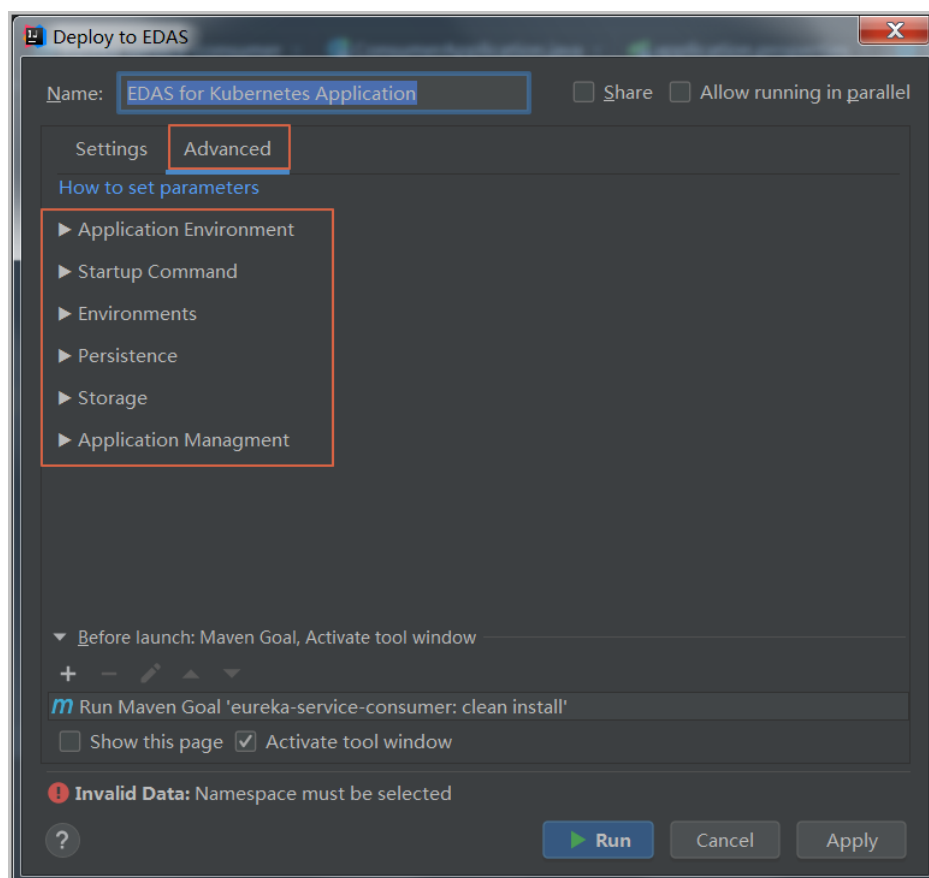
Maven 任务来构建部署包。

- ii. **Upload File** : 选择 Upload File 方式来构建应用时, 选择上传您的 WAR 包或者 JAR 包, 然后进行部署。
- iii. **Build Image** : 选择 Build Image 方式来构建应用时, 需要依次输入工作目录 **Content Directory**, 文件路径 **Dockerfile** 和镜像标签 **Version**。



- iv. **Select Image** : 选择 Build Image 方式来构建应用时, 可在 **Select Image** 的下拉列表选择需要部署的镜像。

单击 **Advanced** 进入高级部署参数设置页签, 并配置 Application Environment、Startup Command、Environments、Persistence、Storage 和 Appcalition Management 等参数。详细的高级部署参数请参见 `DeployK8sApplicationRequest`。



- i. **Application Environment** : 设置应用运行环境信息, 多次重复部署时需要重新填写 PackageVersion。

▼ Application Environment

EdasContainerVersion	3.5.3
JdkVersion	Open JDK 8
PackageVersion	Using timestamp default

ii. **Startup Command**：设置启动命令信息，详细的部署参数请参见 `DeployK8sApplicationRequest`。

iii. **Environments**：设置部署环境变量信息，详细的部署参数请参见 `DeployK8sApplicationRequest` 中的 **Envs** 部分。

iv. **Persistence**：设置持久化存储。

▼ Persistence

Storage Type	NAS Storage
Storage Service Type	SSD Performance
NAS	1561e49808
Mount Json	[{"nasPath":"/K8s", "mountPath":"/mnt"}]

v. **Storage**：设置本地存储，详细的部署参数请参见 `DeployK8sApplicationRequest`。

Application Management：设置应用生命周期管理。

Poststart 脚本：一种容器钩子。该钩子在容器被创建后立刻触发，通知容器它已经被创建。该钩子不需要向其所对应的 hook handler 传入任何参数。如果该钩子对应的 hook handler 执行失败，则该容器会被杀死，并根据该容器的重启策略决定是否要重启该容器。参考 [Container Lifecycle Hooks](#)

PreStop 脚本：一种容器钩子。该钩子在容器被删除前触发，其所对应的 hook handler 必须在删除该容器的请求发送给 Docker daemon 之前完成。在该钩子对应的 hook handler 完成后不论执行的结果如何，Docker daemon 会发送一个 SIGTERM 信号量给 Docker daemon 来删除该容器。参考 [Container Lifecycle Hooks](#)

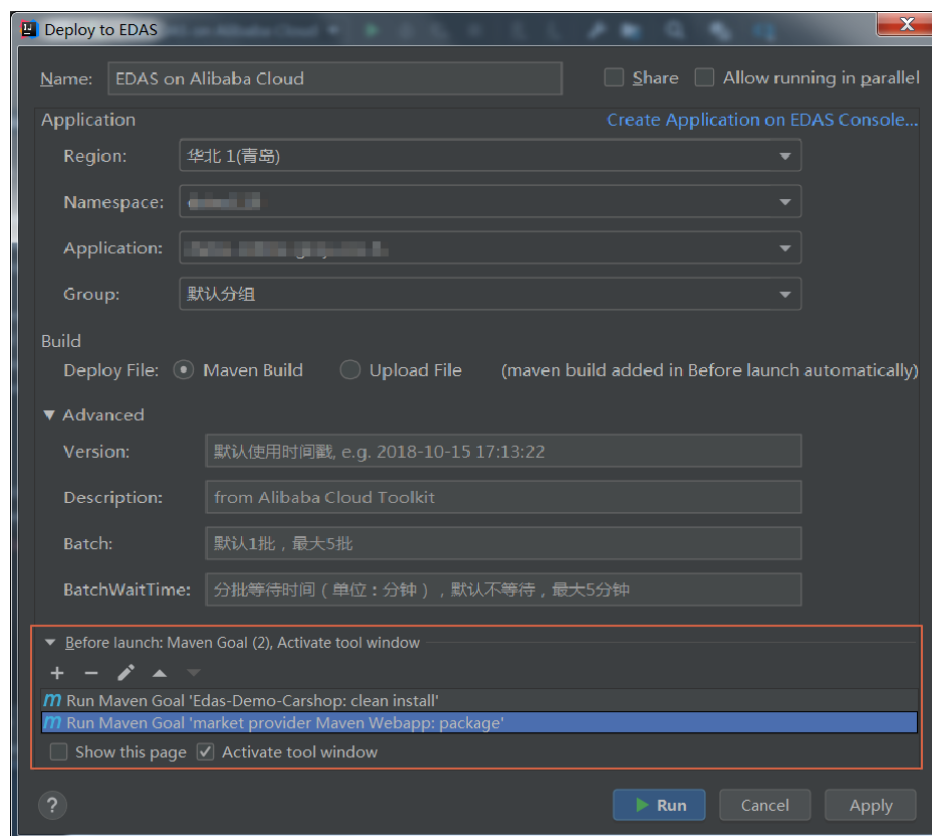
Liveness 脚本：一种探测容器状态的探针，探测应用是否处于健康状态。如果不健康，则删除重建容器。参考 [Pod Lifecycle](#)

Readiness 脚本：一种探测容器状态的探针，探测应用是否启动完成并且处于正常服务状态。如果不正常，则更新容器的状态。参考 [Pod Lifecycle](#)

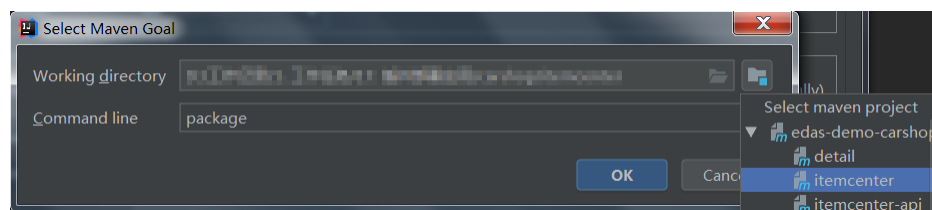
单击 **Run**，IntelliJ IDEA 的 Console 区域会打印部署日志。您可以根据日志信息检查部署结果。

步骤四：管理 Maven 构建任务

在 IntelliJ IDEA 中安装的 Cloud Toolkit 内可以部署 Maven 的构建任务。您也可以在 Deploy to EDAS 的配置页面的 **Before launch** 区域来添加、删除、修改和移动 Maven 构建任务。



在添加 Maven 构建任务编辑框中，您可以单击右侧的文件夹按钮选择当前工程的所有可用模块，并在 **Command line** 中编辑构建命令。



部署多模块工程

实际工作中碰到的大部分 Maven 工程都是多模块的，各个项目模块可以独立开发，其中某些模块又可能会使用到其他的一些模块的功能，这样的项目工程就是多模块工程。

如果您的工程项目为 Maven 多模块工程并且想部署工程中的某子模块，那么需要保证 **EDAS Deployment Configurations** 页面中的 **Before launch** 中的 Maven 构建任务中最后一个任务为该子模块的构建任务。管理

Maven 构建任务的具体操作请参见[管理 Maven 构建任务](#)。

例如：一个 CarShop 工程存在以下示例的子模块：

- carshop
 - itemcenter-api
 - itemcenter
 - detail

其中 itemcenter 和 detail 为子模块，且都依赖于 itemcenter-api 模块，现在想部署 itemcenter 模块，应该怎么做？只需要在配置页面中的 **Before launch** 中增加如下两个 Maven 任务即可：

1. 增加一个在父工程 carshop 中执行 mvn clean install 的 Maven 任务；
2. 增加一个在子模块 itemcenter 中执行 mvn clean package Maven 任务。

常见问题

应用列表获取不到应用

通常出现这种情况为使用子账号来部署应用，且子账号没有同步到 EDAS 系统或者没有进行正确授权，从而导致在应用列表下拉框中看不到应用。您可以通过 RAM 授权或 EDAS 子账号授权来确保子账号已经同步到 EDAS 并且得到授权。

RAM 授权

该授权方式可使子账号访问 EDAS 的所有资源。

1. 在 RAM 控制台左侧导航栏中选择**人员管理 > 用户**。
2. 在**用户**页面上找到需要授权的子用户，单击**操作**列中的**添加权限**。
3. 在**添加权限**面板的**选择权限**区域中，搜索 **AliyunEDASFullAccess** 权限，单击权限策略将其添加至右侧的**已选择**列表中，然后单击**确定**。
4. 在**添加权限**的**授权结果**页面上，查看授权信息摘要，并单击**完成**。
5. 使用主账号登录 EDAS 控制台，在左侧导航栏选择**系统管理 > 子账号**，单击**子账号**页面右上角的**同步子账号**。

EDAS 子账号授权

该授权方式可使子账号细粒度授权访问 EDAS 的资源。

1. 使用主账号登录 EDAS 控制台。
2. 在左侧导航栏选择**系统管理 > 角色**，单击**角色**页面右上角的**创建角色**。
3. 输入一个角色名称，在**可选权限**区域框中，选择**应用管理 > 应用列表 > 基本信息 > 部署应用**，单击**添加**将部署应用角色添加到**已选权限**，然后单击**确定**。
4. 在左侧导航栏选择**系统管理 > 子账号**，单击**子账号**页面右上角的**同步子账号**。
5. 选择需进行授权的子账号，在**操作**列单击**管理角色**，在左侧穿梭框中搜索并选择上面创建的角色，将该角色添加到右侧**已选角色**列表中，然后单击**确定**。

6. 选择需进行授权的子账号，在操作列单击**授权应用**，选择应用添加到右侧列表进行授权，然后单击**确定**。

配置 Endpoint

说明：如果使用 EDAS 专有云企业版，还需要按以下步骤在 Cloud Toolkit 中配置 Endpoint。Endpoint 请联系 EDAS 技术支持获取。

在 **Preference (Filtered)** 对话框的左侧导航栏中选择 **Appearance & Behavior**Endpoint。

在 **Endpoint** 界面中设置 Endpoint，配置完成后，单击 **Apply and Close**。

问题反馈

如果您在使用 Cloud Toolkit 过程中有任何疑问，欢迎您扫描下面的二维码加入钉钉群进行反馈。



灰度发布

灰度发布简介

灰度发布是指在应用的新、旧版本间平滑过渡的一种发布方式。

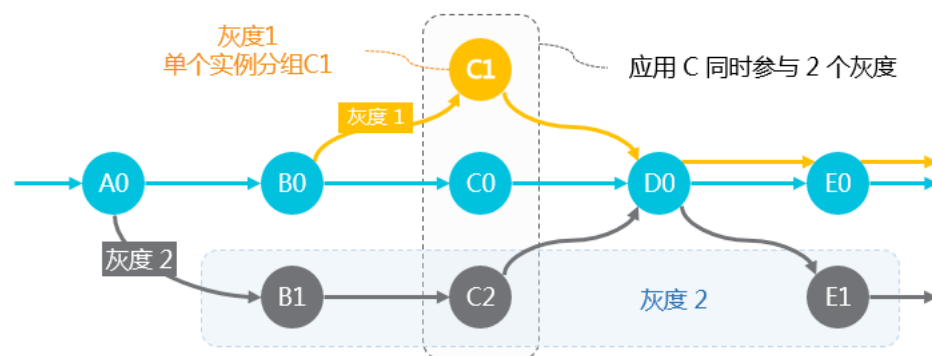
使用场景

在应用发布与产品迭代中，通常会采用灰度发布来控制上线风险。在发布前先进行小规模的生产环境验证，让新版本的应用实例来承接、处理限定规模或范围的业务流量。通过收集使用体验的数据，对应用新版本的功能、性能、稳定性等指标进行评判，进而决定扩大新版本部署范围直至全量升级或回滚至老版本。

除了这种场景外，基于服务保障或其它考虑，还会要求业务系统的部分应用另外准备一组应用实例，用来处理来自特定渠道或用户的业务流量。

实现原理

灰度发布在实现上就是对流量按指定的规则进行识别，并引导到下游应用对应的实例分组。



如上图所示，某个业务系统包含 5 个应用 A~E。每个应用都有默认分组（非灰度）A0~E0。

- 应用 C 创建了实例分组 C1，并部署了新版本，通过单个应用的流量控制（灰度1）进行平滑升级，即单个应用的灰度发布。
- 应用 C 创建了实例分组 C2，和其它应用的灰度实例分组一同参与了另一个全链路流量控制（灰度2）来处理特定渠道或用户的业务流量，即全链路灰度发布。

EDAS 通过定义灰度环境来管理灰度发布。灰度环境包含灰度的识别规则，同时也是一个逻辑空间概念，容纳同处于该灰度环境的应用实例分组。因此，也可以将应用的某个实例分组（非默认分组）添加或移出某个灰度环境。

灵活特性

EDAS 全链路灰度方案能够通过控制台实现灰度发布和流量管控，具有以下灵活特性：

只需为要求灰度的部分应用准备实例资源，不用将业务系统整体再搭建一套。

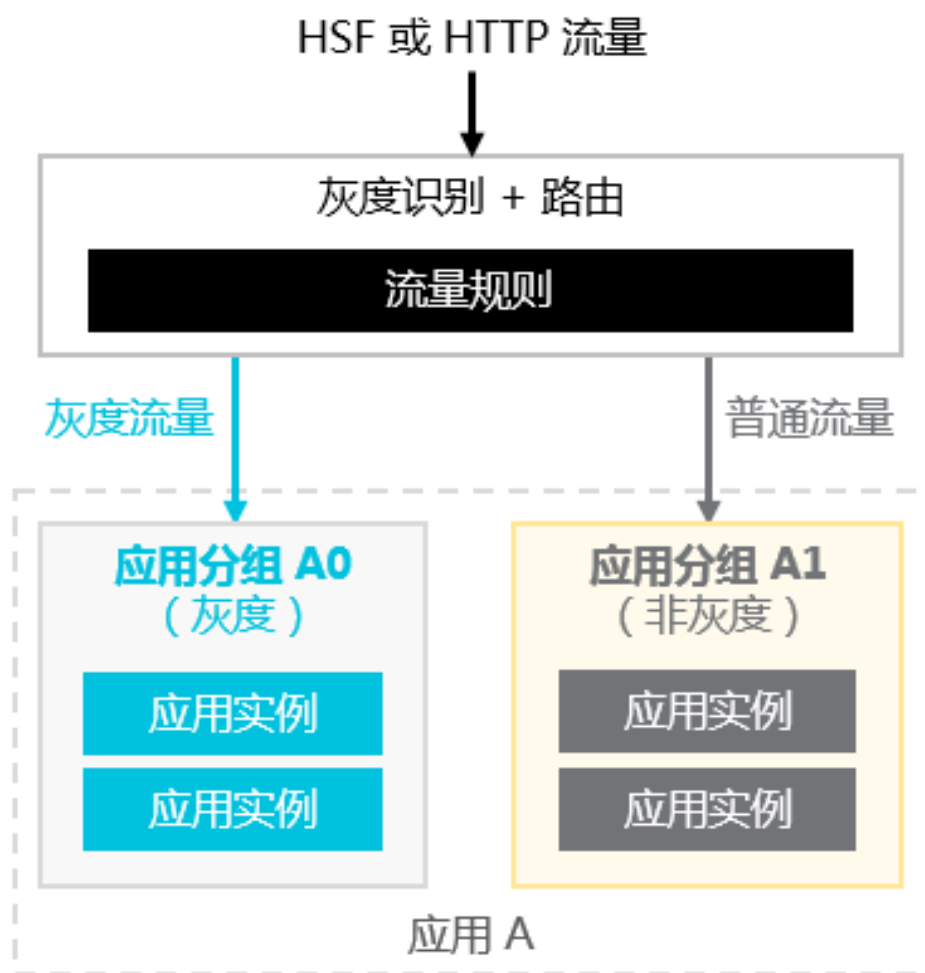
支持多重灰度，允许不同的应用有各自不同的灰度控制，甚至允许同一个应用同时参与多个灰度控制。

支持链路灰度，允许多个应用同处于一个灰度控制，上游环节识别出来的灰度流量，经过非灰度的中间应用环节，在下游环节仍可路由到对应的灰度应用实例。

如何对单个应用进行灰度流量控制

单个应用的新版本发布过程中，灰度流量控制是最常见的场景。由于不涉及多个应用，可以直接在新版本应用实例分组上设置规则，按规则识别灰度流量并由当前应用分组承接。

单个应用灰度发布的示意图如下：



发布流程

前提条件：创建灰度实例分组。创建承接灰度流量的实例分组并添加 ECS 实例。

设置流控规则和部署新版本。

设置流控规则和部署新版本可以有多种策略。您可以根据具体情况评估选用，详情请参考灰度发布策略。

前提条件

已创建灰度实例分组。

如果您使用子账号进行灰度流量控制，还需要是使用主账号为该子账号添加所需权限。

登录 EDAS 控制台。

在左侧导航栏选择**系统管理** > **角色**。

在**角色**页面对应的子账号右侧单击**管理权限**。

在**管理权限**页面添加以下权限。

只读权限：**查看集群**、**查看 app 信息**、**查看服务列表**和**查看流量管理规则**。

写权限：**设置流量管理规则**。

设置并启用流控规则

单个应用流量控制目前支持 HTTP 和 HSF 两种类型的灰度流量规则。

HTTP 流控

支持指定 URL 访问路径，并用 Cookie 值、Header 值或者 URL 参数，根据按 100 取模后余数的范围或白名单作为条件判定流量。

注意，一个应用如果有多个分组都设置了 HTTP 流控，只有最后设置的生效。

进入应用详情页，在**基本信息**页面**应用设置**区域的**流量管理**右侧单击**开启**。

如果未开启流量管理，则无法设置 HTTP 流控规则。

在**实例部署信息**页面灰度实例分组右侧单击**流量控制**，在弹出的下拉菜单中选择**HTTP 流控**。

在**流量控制**页面勾选**启用下面的配置来控制进入当前应用分组的 HTTP 请求流量**。

在**流量控制**页面设置流控规则参数，然后单击**保存**。

流量控制

☐ 启用下面的配置来控制进入当前应用分组的HTTP请求流量

选择path：

HTTP 相对路径，例如 /a/b，注意是严格匹配，留空代表匹配任何路径

* 场景规则：

☒ 满足下列所有规则 ☐ 满足下列规则之一

* 条件列表：

参数类型	参数	条件	值
根据Cookie值	Cookie字段名，最长64个字符	请选择	

+添加规则条件

保存

放弃

流控规则参数说明：

参数	说明	
选择 path	输入访问资源的路径。	
场景规则	如果配置了多条规则，则要设置这些规则生效的条件。包括满足所有条件或满足条件之一。	
规则列表	参数类型	包括根据 Cookie 值、根据 HTTP Header 值和根据 URL 参数值。
	参数	输入参数，不能超过 64 个字符。书写规范和样例请参考流控规则参数说明。
	条件	选择条件，包括按 100 取模和白名单两种方式。
	值	输入参数取模或列表的值。

HSF 流控

支持指定服务和方法，并用方法的参数值，按取模（ mod 100 ）后余数的范围或列表（ list ）作为条件判定流量。

在**实例部署信息**页面灰度实例分组右侧单击**流量控制**，在弹出的下拉菜单中选择 **HSF 流控**。

在**流量控制**页面勾选**启用下面的配置来控制进入当前应用分组的 HSF 请求流量**。

在**流量控制**页面设置流控规则参数，然后单击**保存**。

流量控制

1

注意：要设置 HSF 流控，需要保证该应用的容器为 3.5.3 或更新的版本。

☐ 启用下面的配置来控制进入当前应用分组的HSF请求流量

* 选择服务：

com.alibaba.edas.demo.api.DempService:1.0.0(hsfgroup),括号内为服务分组

* 方法：

echoTime (java.lang.String, java.util.List<java.lang.Integer>)

* 场景规则：

☐ 满足下列所有规则 ☐ 满足下列规则之一

条件列表

参数	参数值获取表达式	条件	值
请选择	args	请选择	

+添加规则条件

保存

放弃

流控规则参数说明：

参数	说明	
选择服务	选择入口应用中的服务。	
选择方法	选择入口应用中的方法。	
场景规则	如果配置了多条规则，则要设置这些规则生效的条件。包括满足所有条件或满足条件之一。	
规则列表	参数类型	包括 根据 Cookie 值 、 根据 HTTP Header 值 和 根据 URL 参数值 。
	参数	输入参数，不能超给过 64 个字符。书写规范和样例请参考流控规则参数说明。
	条件	选择条件，包括 按 100 取模 和 白名单 两种方式。
	值	输入参数取模或列表的值。

部署新版本

您可以根据您的应用框架、部署包类型选择部署环境和部署方式。详情请参见部署应用。

如何进行全链路灰度发布

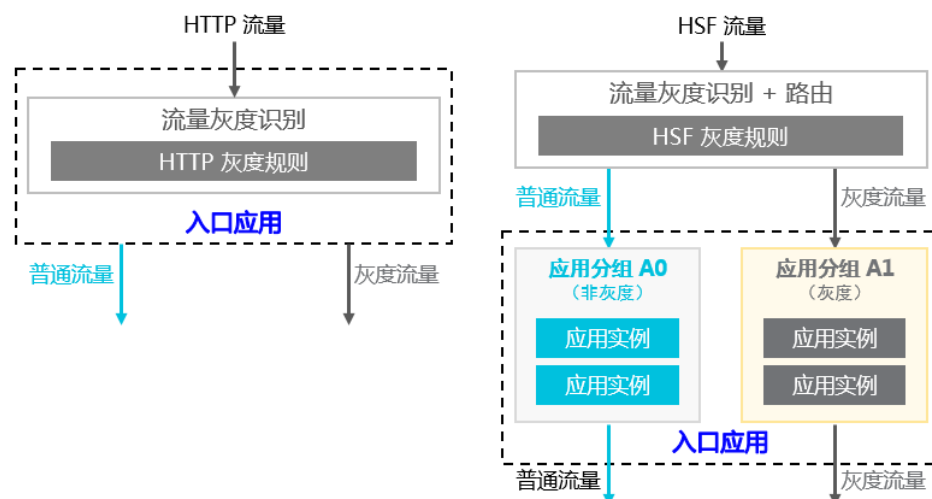
全链路灰度发布涉及多个应用的灰度流量控制场景，在 EDAS 中被定义为一个灰度环境。灰度环境包含流量灰

度判断（也称流量灰度打标）的规则，以及指定哪些应用的哪些分组属于这个灰度环境，承接灰度流量。关于灰度发布的原理，请参见灰度发布简介。

全链路灰度支持多重灰度，即在一个业务系统中可以定义多个灰度环境同时工作。要注意的是，一个应用可以有多个分组属于不同的灰度环境，但一个分组只能属于一个灰度环境。

入口应用与灰度规则

在全链路灰度中首先要指定使用该灰度规则的入口应用，并根据 HTTP 和 HSF 指定各自的灰度策略。入口应用和灰度规则作用如下图所示。



对于 HTTP 流量，在流量进入该入口应用之后才会执行灰度判断。如果符合规则条件，就会将其标记为对应的灰度。简单地说，就是只做灰度判断。

注意:

此处仅仅是对已收到的 HTTP 流量进行灰度判断，全链路灰度目前还不支持 HTTP 流量的灰度路由。实际是在入口应用处根据 HTTP 流量判断灰度，对之后各环节产生的所有 HSF 流量进行灰度路由。

和全链路灰度发布不同，单个应用灰度发布的 HTTP 流量控制规则是针对应用分组自身的，不只做灰度判断，也会进行灰度路由。

对于 HSF 流量，在流量进入该入口应用之前就会执行灰度判断，并且进行标准的灰度路由处理。如果入口应用有属于当前灰度的应用分组实例，灰度流量就会直接路由到该灰度分组，否则灰度流量就会降级处理，路由到该入口应用的非灰度默认分组。简单地说，就是既做灰度判断，又进行灰度路由。

灰度规则的设置和在单应用流量控制中是类似的，不同的是可以有多条规则。

- 每个规则都要指定流量协议类型，有 HTTP 和 HSF 两种类型。
- 每个规则都可以有多项规则条件，相互之间是“与”或者“或”的关系。

前提条件

已创建灰度实例分组。

如果您使用子账号进行全链路灰度发布，还需要使用主账号为该子账号添加所需的权限。步骤如下：

登录 EDAS 控制台。

在左侧导航栏选择**系统管理** > **角色**。

在**角色**页面对应的子账号右侧单击**管理权限**。

在**管理权限**页面添加以下权限：

只读权限：**查看集群**、**查看 app 信息**、**查看服务列表**和**查看灰度发布配置**。

写权限：**更新灰度发布配置**。

发布流程

全链路灰度发布中有多种策略，您可以根据您的实际需求和情况评估使用，没有固定的流程。详情请参见灰度发布策略。

创建灰度环境

创建灰度环境是灰度发布的核心步骤。您需要在创建灰度环境时选择入口应用、配置流量规则和添加应用实例分组。

设置灰度环境基本信息

在控制台左侧导航栏选择**应用管理** > **灰度发布**。

在**灰度发布**页面右上角单击**新建环境**。

在**基本信息**页面设置**命名空间**（包括地域和命名空间）**灰度环境名称**、**灰度标记**、和**灰度环境描述**，然后单击**下一步**。

基本信息

设置入口流量规则

选择灰度应用

创建完成

* 命名空间：

华北2

选择应用所在命名空间

* 灰度环境名称：

* 灰度标识：

灰度环境描述：

请输入描述信息

0/64

下一步

设置入口应用和流量规则

说明：HSF 应用需要使用 3.5.3 及更新的容器版本才能设置 HSF 流量规则。

在设置入口流量规则页面选择入口应用并配置流量规则，然后单击下一步。

基本信息

设置入口流量规则

选择灰度应用

创建完成

* 入口应用：

请选择

* 协议类型：

☒ HTTP

☐ HSF

选择path：

HTTP 相对路径，例如 /a/b，注意是严格匹配，空白代表匹配任何路径

* 场景规则：

☒ 满足下列所有规则

☐ 满足下列规则之一

* 条件列表：

参数类型	参数	条件	值
根据Cookie值	Cookie字符串名，最长64个字符。	请选择	

+添加规则条件

+创建新的入口流量规则

上一步 下一步

入口流量规则参数说明：

参数	说明	
入口应用	在下拉菜单中按规划选择该灰度环境的入口应用。	
协议类型	根据实际需求选择协议类型，HTTP 或 HSF。	
path（适用于 HTTP）	输入访问资源的路径。	
服务（适用于 HSF）	选择入口应用中的服务。	
方法（适用于 HSF）	选择入口应用中的方法。	
场景规则	如果配置了多条规则，则要设置这些规则生效的条件。包括满足所有或满足其中之一。	
条件列表	参数类型	包括根据 Cookie 值、Header 值和根据 URL 值。
	参数	输入参数，不能超过 64 个字符。输入规范和样例请参考/。
	条件	选择条件，包括按 100% 流量和按 10% 流量两种方式。

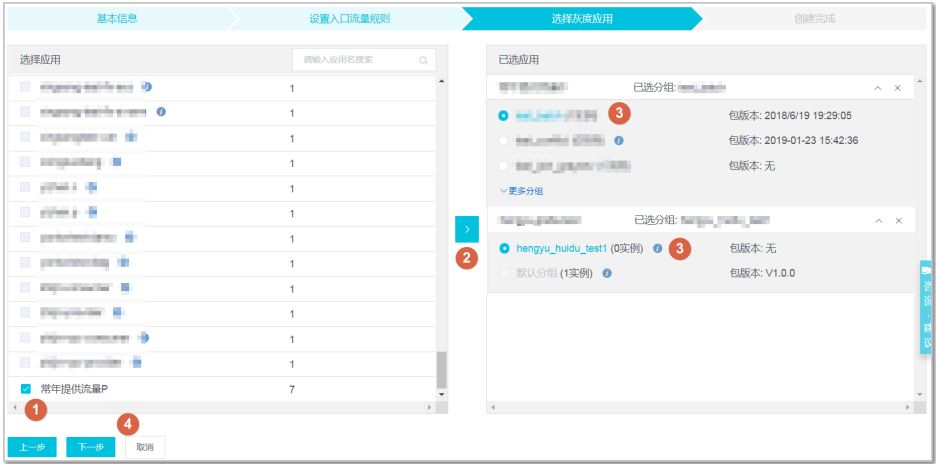
	值	输入参数取模或列表的
--	---	------------

说明：

- 如果需要设置多个规则，单击**添加规则条件**。
- 如果需要创建多个入口流量规则，单击**创建新的入口流量规则**。

添加应用实例分组

在**选择灰度应用**页面**选择应用**列表中按规划勾选应用，单击 > 添加到**已选应用**列表中，勾选应用实例分组，单击**下一步**。



说明：

- 选择应用列表中某些应用不可勾选，且应用名右侧有 **i** 标识，则表示该应用只有默认分组，而默认分组不能添加到灰度环境中。
- 已选应用列表中某些应用实例分组右侧同样有 **i** 标识，则表示该分组下没有实例，需要添加实例。

在**创建完成**页面检查灰度环境相关设置，确认无误后，单击**提交**。

启动流控规则

灰度环境创建完成后，开启入口流量即可启动灰度环境中的入口应用的流控规则。具体步骤如下：

返回**灰度发布**页面。

选择灰度环境所在的**地域**和**命名空间**。

找到创建完成的灰度环境，单击**入口流量**右侧的按钮。

部署新版本

您可以根据您的应用框架、部署包类型选择部署环境和部署方式。详情请参见部署应用。

结果验证

完成全链路灰度发布后，您可以通过监控灰度流量验证灰度流量是否分发到了对应的灰度环境或应用实例分组中，详情请参见监控灰度流量。

在容器服务 K8s 集群中灰度发布 Spring Cloud 应用

部署在容器服务 Kubernetes 集群中的 Spring Cloud 应用在迭代中需要升级，可以使用灰度发布进行小规模验证，验证通过后再全量升级。

使用限制

在容器服务 K8s 集群中灰度发布 Spring Cloud 应用有以下使用限制：

目前仅支持单应用在发布过程中的灰度流量控制。

入口应用（承接外部流量的第一个应用节点）暂不支持灰度。

入口应用直接承接外部流量，由于暂不支持对这种流量进行截获和路由，因此对入口应用暂不能提供流量灰度控制。一种理想的模式是搭建微服务网关来作为承接外部流量的“入口应用”，这样所有业务应用都在网关之后，就可以实现流量灰度控制了。

如果应用包含了以下原生功能或配置时，请勿使用灰度发布，否则发布后该原生功能将出现异常：

- HPA
- Rancher
- Istio
- 依赖 Deployment.Metadata.Name 或 Deployment.Metadata.Uid 的功能与配置

前提条件：在应用程序中添加支持灰度的依赖

Spring Cloud 应用不具备灰度路由功能，需要在**灰度应用**和**上游应用**的pom.xml文件中添加支持灰度特性的依赖。

```
<dependency>
<groupId>com.alibaba.edas</groupId>
<artifactId>edas-sc-gray-starter</artifactId>
<version>1.0.0</version>
</dependency>
```

添加依赖后，路由规则将由 EDAS 管控，客户端（应用）无法定制路由规则。当灰度发布结束后，如果应用需要定制路由规则，可以通过配置管理配置spring.cloud.edas.gray.enabled=false取消灰度特性。

这里的上游应用可能是您自己在微服务环境内搭建的微服务网关（API Gateway）类型应用，例如 Zuul 或者 Spring Cloud Gateway。为了帮助您更好的识别灰度应用和上游应用，下面以示例方式说明：

注意：目前依赖对 Spring Cloud Gateway 的支持还不完善，暂不能提供流量灰度控制。

微服务架构中不包含微服务网关（API Gateway）

服务拓扑关系为：microservice1 > microservice2 > microservice3

- 如果要灰度发布 microservice3：为 microservice3 和 microservice2 添加依赖。
- 如果要灰度发布 microservice2：为 microservice2 和 microservice1 添加依赖。
- 暂不支持灰度发布 microservice1。

微服务架构中包含微服务网关（API Gateway）

服务拓扑关系为：Zuul > microservice1 > microservice2

- 如果要灰度发布 microservice2：为 microservice2 和 microservice1 添加依赖。
- 如果要灰度发布 microservice1：为 microservice1 和 zuul 添加依赖。

服务拓扑关系为：

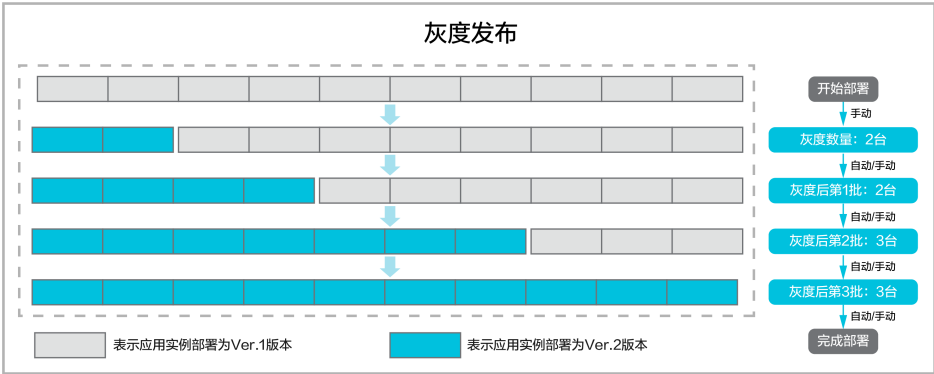
```
zuul
↓↓
microservice1 → microservice2
```

- 如果要灰度发布 microservice2：为 microservice2、microservice1 和 zuul 添加依赖
- 如果要灰度发布 microservice1：为 microservice1 和 zuul 添加依赖。

示例场景说明

一个应用包含 10 个应用实例，部署了 Ver.1 版本，现在要将应用升级为 Ver.2 版本。在其中 2 个实例中进行

灰度发布，验证 Ver.2 版本，验证通过后，再将剩余 8 个实例分批部署 Ver.2 版本，完成整个应用的升级。



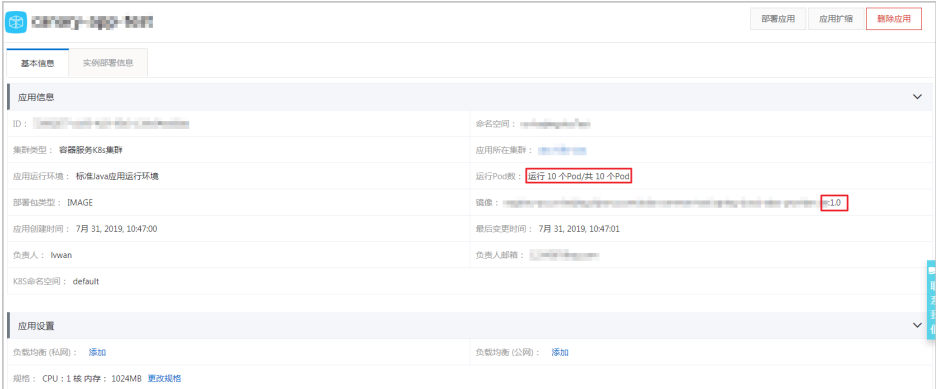
灰度发布流程

- 确认当前应用状态。
- 确认应用的实例数和版本。
- 灰度发布。
- 设置新版本、灰度策略、灰度规则，然后启动灰度发布。
- 说明：灰度路由只针对第 1 批发布（灰度发布）有效。如果在变更详情中单击了**开始下一批**或者**回滚**，则灰度规则会被清理，失效。
- 验证新版本。
- 分批发布剩余实例。

步骤一：确认当前应用状态

确认当前应用的实例数和版本，以决定灰度实例数和之后分批发布的批次。

- 登录 EDAS 控制台。
- 在左侧导航栏中选择**应用管理**->**应用列表**，在应用列表页面中单击需要灰度发布的应用名称。
- 在应用详情也查看应用现在的版本和实例数。



步骤二：灰度发布

设置新版本、灰度策略、灰度规则，然后启动灰度发布。

在应用详情页右上角单击**部署应用**。

在**部署模式选择**页面**灰度发布**区域单击**开始部署**。

注意：容器服务 Kubernetes 集群中灰度发布的流量控制暂时只对非入口应用的 Spring Cloud 应用生效。



设置灰度发布的新版本。

应用的部署方式由首次部署方式决定，根据首次部署方式设置对应的新版本。

Spring Cloud 应用通常为 JAR 包或镜像，所以本文以 JAR 包或镜像为例。

JAR 包部署：上传新版本 JAR 包或者设置新版本 JAR 包的地址，并选择 Java 环境和版本。

* 应用运行环境：

标准Java应用运行环境

* Java环境：

Open JDK 8

* 文件上传方式：

上传JAR包

* 上传JAR包：

选择文件

http://edas-bj-oss-cn-beijing-internal.aliyuncs.com/apps/K8S_APP_ID/94dc04fc-a604-475a-acb5-765d3e1f5cde/gray-provider-example-0.0.1-SNAPSHOT.jar

* 版本：

2.0

镜像：在镜像仓库中选择新版本的镜像。

配置镜像*



选择镜像仓库

选择镜像名称

选择镜像版本

类型：PRIVATE

来源：

镜像仓库

1.0

在发布策略区域配置发布策略参数。

右侧的发布策略配置信息会根据配置显示灰度发布流程。

发布策略

1

当您的应用使用了以下K8s原生功能或配置（包括HPA、Rancher、istio，以及依赖Deployment.Metadata.Name或 Deployment.Metadata.Uid的功能与配置）时，请勿使用灰度发布或分批发布。否则，应用部署之后，这些K8s原生功能或配置将出现异常。详情请参考[使用须知](#)。

灰度数量

2

(应用当前实例数10)

为了保证应用稳定性，灰度实例数不能超过应用实例总数的1/2

灰度后剩余批次

3

灰度发布后，剩余的应用实例将按照指定的批次完成分批发布

分批间处理方式

自动

分批时间间隔

5

min

批次内部署间隔

10

秒

收起高级选项

发布策略配置信息

1

开始部署

2

灰度数量：2台

3

(手动开始) 灰度后第1批：2台

批次间隔5min

4

(自动开始) 灰度后第2批：3台

批次间隔5min

5

(自动开始) 灰度后第3批：3台

6

部署结束

发布策略参数说明：

- **灰度数量**：灰度发布的应用实例数量。右侧会显示应用当前实例数，为了保证应用稳定性，灰度实例数不能超过应用实例总数的 1/2。如示例中的实例总数为 10，则灰度数量不能大于 5。
- **灰度后剩余批次**：灰度发布后，剩余的应用实例按照设定的批次完成分批发布。

分批间处理方式：**灰度后剩余批次**设置为 2 批或 2 批以上时需要设置，包括自动和手动两种方式。

- **自动**：根据分批时间间隔自动分批发布。
- **手动**：人为手动触发下一个批次的发布。

分批时间间隔：**分批间处理方式**设置为**自动**时需要设置，即剩余批次间的发布时间间隔，单位为分钟。

批次内部署间隔：每一批次内，如果应用实例数大于 1，应用实例间的部署时间间隔，单位为秒。

109

设置入口流量灰度规则。

在 Spring Cloud 灰度规则 区域单击创建新的入口流量规则

在规则区域设置参数。

path :

HTTP 相对路径，例如 /a/b，注意是严格匹配，留空代表匹配任何路径

* 场景规则 :

☒ 满足下列所有规则

☐ 满足下列规则之一

* 条件列表 :

参数类型	参数	条件	值
根据URL参数值	version	白名单	1

+ 添加规则条件

流量控制参数说明：

- i. 场景规则：选择满足下列所有规则。
- ii. 条件列表：包含 Cookie、HTTP Header 和 URL 参数 3 种方式，根据实际需求选择。
 - 参数类型：包含根据 URL 参数值、根据 HTTP Header 值和根据 URL 参数值，根据实际需求选择。
 - 参数：输入参数。
 - 条件：选择 白名单或按 100 取模。
 - 值：输入参数值。

根据实际需求，如果需要，可以重新配置启动命令、环境变量、Hosts 绑定设置和健康检查等高级选项。

设置完成后，单击**确定**，启动灰度发布。

灰度发布开始后，控制台会自动跳转到变更详情页面，可以查看灰度发布的实时状态。

灰度发布启动后，在**变更详情**页面可以查看**已部署实例数**（灰度实例数）和总实例数，单击**流量查看**。

在**流量监控**对话框基于**实例视角**和**服务视角**验证灰度流量是否分发到灰度实例中及监控数据。

详情请参见**监控灰度流量**。

步骤三：验证新版本

对灰度实例部署的新版本进行验证。

灰度发布成功后，开始验证新版本。新版本验证的时间可以自主控制。

说明：灰度发布是应用升级的中间态，在为完成整个应用的升级前不能对应用进行如扩容、缩容、删除等生命周期管理操作。

- 新版本验证通过，可以分批发布剩余实例。
- 新版本验证未通过，可以回退应用的灰度实例到原有版本。

步骤四：分批发布剩余实例

将剩余实例按发布策略完成分批发布。

在应用详情页左侧导航栏单击**变更记录**，然后在**变更记录**页面灰度发布记录的操作列单击**查看**。



在**变更详情**页面单击**开始下一批**，在弹出的确认对话框中单击**确定**。

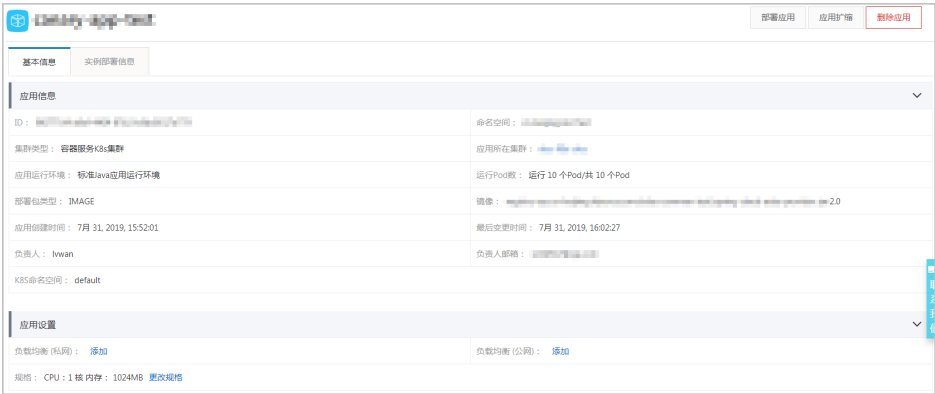
注意：灰度路由只针对第 1 批发布（灰度发布）有效。如果在变更详情中单击了**开始下一批**或者**回滚**，则灰度规则会被清理，失效。

本示例中剩余实例会按发布策略分为 3 批，每批间隔 5 分钟，批次内实例间间隔 10 秒自动发布。可以通过变更详情查看发布过程。

在分批发布过程中，如果发现新版本有问题，也可以在批次发布间隔回滚应用。

结果验证

灰度发布和分批发布后，进入**基本信息**页面，查看应用的版本和实例数。版本应该升级到 Ver.2 版本，运行实例数应该为 10。至此应用升级成功。



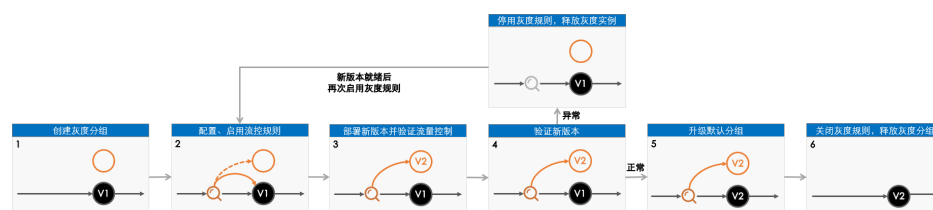
如何通过灰度发布升级单个应用

在应用的迭代过程中，可以通过灰度发布对新版本进行小规模验证，在验证通过后，再将应用全量升级到新版本。

示例场景说明

Web 应用 A，包含 2 个应用实例，通过 WAR 包在 ECS 集群中部署了 V1 版本。V2 版本开发完成后，需要先在 1 个实例进行验证，验证通过后，再将另一个实例从 V1 版本升级到 V2，从而完成应用 A 的升级。

灰度发布流程



创建灰度分组。

在灰度分组中配置并启用流量控制规则。

在灰度分组中部署新版本（V2），并验证指定流量是否分发到灰度分组的实例中。

通过分发到灰度分组中的流量验证新版本。

验证通过后，将默认分组中的应用版本升级为 V2。

如果验证发现问题，停用灰度分组的流控规则，将灰度分组中的实例更换回默认分组。待 V2 版本更新完成后，再次启用灰度分组的流控规则，并在灰度分组中进行部署、验证。

关闭灰度分组的流量控制规则，删除灰度分组。

步骤一：创建灰度分组

在 ECS 集群中，不同应用版本是基于实例分组部署的，而且流控规则也是基于实例分组配置的，所以需要先创建灰度分组。

登录 EDAS 控制台。

在左侧导航栏选择**应用管理** > **应用列表**，在**应用列表**页面单击具体应用名称。

在应用详情页面单击**实例部署信息**页签，然后在页面右上方单击**创建新分组**。

在**创建分组**对话框将**分组名**设置为**灰度分组**，然后单击**创建**。

分组成功创建后，页面上方会出现**创建分组成功**的提示。

步骤二：配置并启用流量控制规则

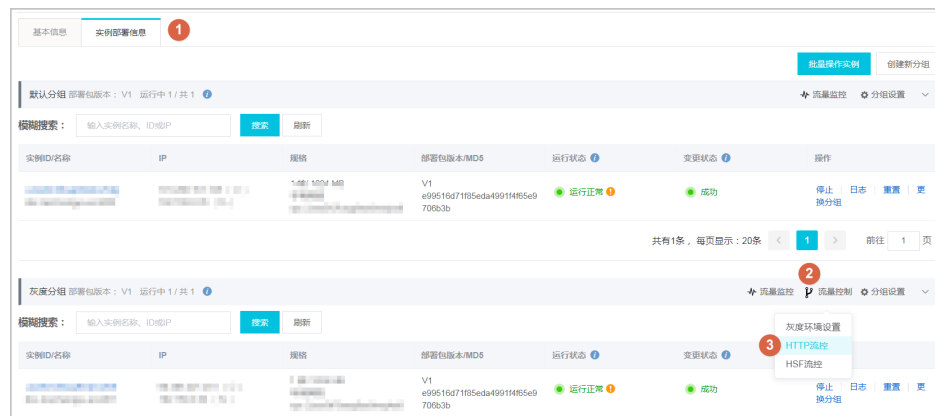
流量控制包含 HSF 和 HTTP 两种类型。

- 如果设置 HTTP 流控，需要在应用**基本信息**页面开启**流量管理**。
- 如果设置 HSF 流控，不需要开启**流量管理**，但需要保证应用的容器 EDAS Container 为 3.5.3 及以上版本。

下文以 HTTP 流控为例进行说明。

在**基本信息**页面**应用设置**区域**流量管理**右侧单击**开启**。

在**实例部署信息**页面灰度分组区域右上角单击**流量控制**，在下拉列表中选择 **HTTP 流控**。



在**流量控制**对话框中设置流控参数，勾选**启用**下面的配置来控制进入当前应用分组的 HTTP 请求流量，然后单击**保存**。

流量控制参数说明：

- 注意：**参数设置完成后，需要勾选启用下面的配置来控制进入当前应用分组的 HTTP 请求流量，流控规则才会生效。

灰度规则生效后，如果灰度分组没有实例来承接灰度流量，会自动由默认分组处理。

步骤三：部署新版本并验证流量控制

更换实例到灰度分组。

返回**实例部署信息**页面，在默认分组中选定实例的操作列单击**更换分组**。

在**更换分组**对话框中单击**目标分组**右侧的下拉框，在下拉列表中选择**灰度分组**，然后单击**确定**。

实例更换分组后，将会默认使用当前部署的版本 V1。

部署新版本

您可以通过控制台和工具等多种方式部署，本示例以控制台部署为例。如果您想使用其它部署方式，请参见[应用部署概述](#)。

在应用详情页右上角单击部署应用。

在**部署应用**对话框中设置部署参数，然后单击**部署**。

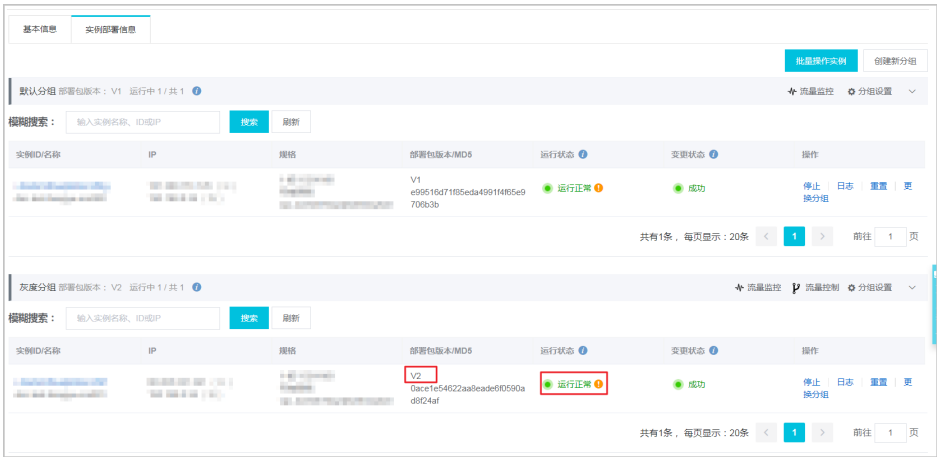
说明：本文的示例应用 V1 版本为 WAR 包部署，所以再次部署时，必须通过 WAR 包部署。如果您的应用此前使用了 JAR 包部署，选择 **JAR 包** 相关选项即可。



- i. **应用部署方式**：选择 **WAR 包部署**。
- ii. **文件上传方式**：选择**上传 WAR 包**，单击**选择文件**，在弹框中选择应用本地的 V2 版本 WAR 包。
- iii. **版本**：填写 **V2**。
- iv. **分组**：选择**灰度分组**。
- v. **批次**：选择 **1 批**。
- vi. **分批方式**：选择**自动**。

部署完成后，控制台会自动跳转到变更详情页面。您可以查看此次部署的进度。待**执行状态**显示为**执行成功**后，说明部署成功。如果部署失败，变更详情中会有相关的日志信息，您可以根据日志排查，详情请参见变更流程问题排查指南。

返回应用详情页面，单击**实例部署信息**页签，可以看到灰度分组中的**部署包版本**更新为 **V2**，且**运行状态**为**运行正常**。



验证流量控制

在浏览器的地址栏中输入http://<默认分组中实例的 IP>:<服务端口>并回车。

进入应用 V1 版本的 Web 页面。

在浏览器的地址栏中输入http://<默认分组中实例的 IP>:<服务端口>?version=1并回车。

进入应用 V2 版本的 Web 页面。

至此说明流量控制规则生效，制定的流量会被分发到灰度分组的实例中。

步骤四：验证新版本

您可以根据实际业务验证新版本。

如果验证发现问题，停用灰度分组的流控规则，将灰度分组中的实例更换回默认分组。待 V2 版本更新完成后，再次启用灰度分组的流控规则，并在灰度分组中进行部署、验证。

步骤五：升级默认分组的应用版本

在新版本 V2 验证通过后，将默认分组的应用升级到 V2 版本。

升级应用即再一次部署应用，详细操作步骤请参见部署新版本。

步骤六：禁用流控规则并释放灰度分组

在两个分组下的两个实例都升级到 V2 版本后，需要禁用流控规则，并释放灰度分组。

在**实例部署信息**页面灰度分组区域右上角单击**流量控制**，在下拉列表中单击 **HTTP 流控**。

在**流量控制**对话框中取消勾选**启用下面的配置来控制进入当前应用分组的 HTTP 请求流量**，然后单击**保存**。

返回**实例部署信息**页面，在灰度分组中实例的操作列单击**更换分组**。

在**更换分组**对话框中单击**目标分组**右侧的下拉框，在下拉列表中选择**默认分组**，然后单击**确定**。

返回**实例部署信息**页面，在灰度分组区域右上角单击**删除分组**。

灰度发布的约束限制

全链路灰度提供了灵活的灰度发布方式，同时也有一些限制以及约定。

应用实例分组的归属灰度唯一

如果一个应用实例分组同时属于多个灰度环境，会造成流量控制冲突。所以，在全链路灰度中，一个应用实例分组只能属于一个灰度环境。

说明：单个应用流量控制里对应用分组启用 HSF 流量规则后，实际上也会创建一个灰度环境来容纳这个应用分组。

流量灰度属性唯一

当一个流量已经被标记属于某个灰度以后，不会因为符合其它灰度规则的条件被覆盖标记成为其它灰度。

灰度规则生效优先级

同一个应用可能会作为多个灰度环境的入口应用，可能会有流量同时符合多个灰度规则的情况。根据上面流量灰度属性唯一的限制，需要设置多个灰度规则的生效优先策略。目前的约定是按规则的创建、变更时间来判定，后创建或者后变更的灰度规则优先。

单应用流量控制与全链路灰度发布的关联限制

一个应用可能同时使用单应用流量控制和全链路灰度发布。正如灰度发布简介中的例子所示，应用 C 的实例分

组 C1 定义了一个单应用流量控制，而实例分组 C2 则参与了涉及多个应用的全链路灰度。

单应用流量控制中设置 HSF 规则相当于创建了一个灰度，而同一个应用分组不能属于两个灰度。因此当一个应用分组（例如上图中的 C2）已经加入到某个全链路灰度环境中，就不允许再在单应用流量控制中单独设置 HSF 规则。同样，如果一个应用分组（例如上图中的 C1）已经设置了自己的 HSF 规则，就不允许加入另外一个全链路灰度。

单应用流量控制中，入口应用同一入口端点规则唯一

单应用流量控制中，对同一个应用的 HTTP 规则和 HSF 规则有唯一性限制，实际上一个“入口端点”只允许一个规则定义。关于入口端点的定义：

- 对于 HTTP 协议，一个端点指的是一个应用，一个应用只有一个分组能设置；
- 对于 HSF 协议，一个端点指的是一个应用接口里的一个方法。

在单应用流量控制里，一个应用不能制定多个 HTTP 规则，一个应用的一个接口方法，不能出现在多个 HSF 规则中。

灰度发布策略

灰度发布时，设置流控规则和部署新版本这两个动作的先后次序并没有限制，可以有多种策略。以下是几种可能的组合方式，您可以根据具体情况评估选用。

使用方式	优势	问题
部署新版本 > 设置流控规则	可以验证灰度分组承接流量的能力。 应用部署和灰度设置只需要一次衔接。	在灰度规则设置生效前，进入新版本应用实例的流量可能不是灰度的。
部署旧版本 > 设置流控规则 > 升级到新版本	可以验证灰度分组承接流量的能力。	最后一步需要注意应用实例的平滑升级，考虑对正在处理请求的影响。 两次应用部署动作。
设置流控规则 > 部署新版本	应用部署和灰度设置只需要一次衔接。	未部署前，灰度流量会降级路由到非灰度环境，但第一批新版本实例在发布后，有受到全部灰度流量冲击的风险。
设置无效流控规则 > 部署新版本 > 设置有效流控规则	可控性最好，新部署的应用实例直到新的有效灰度规则生效后才接收到灰度流量。	两次灰度设置动作。

说明：

- 推荐采用第三种方式。操作简单，可以保证非灰度的流量不会进入灰度环境，但是需要注意在开始部署时灰度应用实例较少，有受灰度流量冲击的风险，要做好流量保护。
- 全链路灰度和单个应用的组合方式相似，只是将设置流控规则改为创建灰度环境。

流控规则参数说明

灰度发布可以基于 HTTP 和 HSF 创建流控规则，它们的规则参数设置有所不同。

HTTP 可以根据 Cookie 值、HTTP Header 值 和 URL 参数值 3 种不同类型设置参数，按取模（mod 100）后余数的范围或列表（list）作为条件判定流量。相对通用和简单，不再详细介绍。需要说明的是，当参数值中包含非数字时将通过哈希（Hash）算法将非数字转化为数字。所以如果您的参数相对复杂，建议您使用列表方式进行流量判定。

本文档将重点对 HSF 流控规则中的参数进行说明。

HSF 参数表达式

全链路灰度支持用参数表达式来获取参数的某个属性，目前已支持下面的表达式：

表达式	说明	备注
args0	当前参数的值	-
args0.name	参数的 name 属性	会翻译成 Java 语句： <code>arg.getName()</code> 。
args0.isEnabled()	参数的 enabled 属性，是一个 bool 类型。	Java 规范里，bool 类型的 getter 格式是 <code>isXXX()</code> 。
args0[0]	arg 是一个数组，获取数组的第一个值。	-
args0.get(0)	arg 是一个 List，获取 List 的第一个值。	-
args0.get(“key”)	arg 是一个 Map，获取 key 对应的值。	-

假设选择的是第一个参数，EDAS在页面会自动生成 args0 前缀。

以上的表达式可以组合使用，比如如下的表达式：

`args0.persons[0].meta.get("name")`先取参数的 persons 数组的第 1 个。再取 person 的 meta 属性，是一个 Map。再取 Map 中 “name” 这个 key 的值。

支持的条件

- = : 支持字符串、数字、Boolean 和 Char 的比较。
- != : 支持字符串、数字、Boolean 和 Char 的比较。
- > : 支持数字的比较。
- >= : 支持数字的比较。
- < : 支持数字的比较。
- <= : 支持数字的比较。

支持的值表达式

HSF 参数匹配条件中的值表达式，代表了一个 Java 中的值。目前，仅支持 Java 中的基本类型，比如数字、字符、Boolean 和 Char 等。不支持复杂类型和用户自定义类型。

值表达式有以下几种：

Java 标准 String

使用 Java 标准 String 表达式来表示一个字符串，被半角双引号(`"`)包含的一段字符，即以 `"` 开始，并以 `"` 结束。比如：

- `"tom"` : 字符串 tom
- `"10"` : 字符串 10
- `"abc "` : 字符串 abc , 后面紧跟着一个空格
- `"a"` : 字符串 a
- `"\n"` : 换行
- `"\" abc\" "` : 字符串 `"abc"`
- `"a\\bc"` : 字符串 a\bc

该表达式可以表达任意的字符串，语法就是 Java 里标准的 String 表达式。

数字类型

如果要表示一个数字类型的值，只需要输入数字本身即可。比如：

- 100
- 1.23
- -3.14
- 1.23f

注意：1.23 在 Java 里默认是 double 类型的，如果要表示 float 类型的 1.23，需要用 1.23f。这是 Java 类型系统的精度决定的。

Boolean 类型

Boolean 类型只有两个值：true 和 false。

Char 类型

Char 类型表示一个字符，用半角单引号（ ' ）包含，即以' 开始，并以' 结束。如 ' a' 。

Null 类型

Null 类型表示 Java 里的 Null。直接输入 null 即可。

字符串字面量

字符串字面量用一种 “一字不差” 的方式来表达一个字符串。里面的字符不需要进行转义。比如：

字符串字面量	Java String
tom	"tom"
"	"\" "
\	"\" "
a\b	"a\b"

下面的表格列出了所有类型的值对应的值表达式：

值类型	值	值表达式（在界面中需要填入的内容）	备注
java.lang.String	"tom"	"tom" 或 tom	
java.lang.String	"true"	"true"	
java.lang.String	"10"	"10"	想表达String类型的10，必须要加上双引号。没有双引号时，会被解析为数字类型的10。
java.lang.String	换行	"\n"	
java.lang.String	'	"\" "	
java.lang.String	"	"\" "	
java.lang.String	\	"\" "	
java.lang.String	aa' bb	"aa' bb"	
int	10	10	
java.lang.Integer	10	10	
byte	10	10	
boolean	true	true	
java.lang.Boolean	true	true	
short	10	10	

long	100	100	
java.lang.Long	100	100	
float	1.23f	1.23f	当arg为float类型时，一定要在数字后加上f。1.23f == 1.23返回false
java.lang.Float	1.23f	1.23f	
double	1.23	1.23	
java.lang.Double	1.23	1.23	
char	'a'	'a'	
null	null	null	

示例

参数为 String

输入框内什么都不用填。表示参数本身。

参数0	args0		=	tom
-----	-------	--	---	-----

参数为数组

假设参数类型为 String 数组。

输入框输入[0]。表示取数组的第一个元素。

参数0	args0	[0]	=	tom
-----	-------	-----	---	-----

参数为 List

假设参数为List<String>。

输入框输入.get(0)。表示取List的第一个元素。注意不要忘记了点号(.)。

参数0	args0	.get(0)	=	tom
-----	-------	---------	---	-----

参数为复杂类型

假设某个方法的第一个参数是如下的类型：

```
public class Person {
```

```
private String name;  
private int age;  
private String[] array;  
private List<String> list;  
private Map<String,String> map;  
}
```

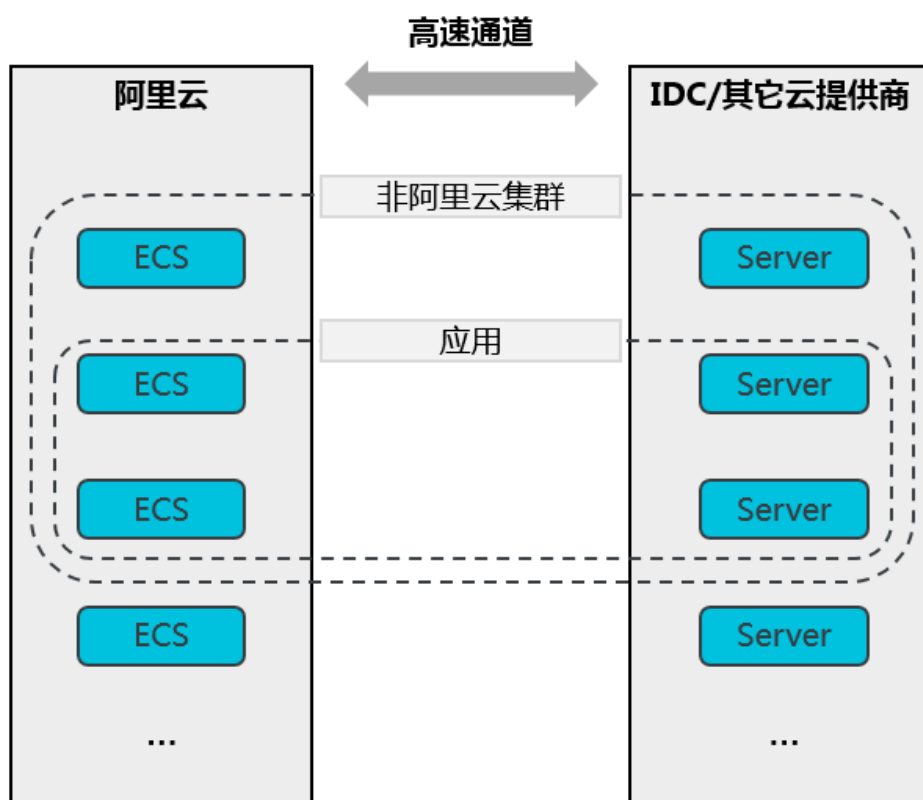
可以设置如下的规则：

参数0 ▾	args0	.name	= ▾	tom
参数0 ▾	args0	.age	> ▾	20
参数0 ▾	args0	.array[0]	= ▾	"tom"
参数0 ▾	args0	.list.get(0)	= ▾	tom
参数0 ▾	args0	.map.get("key")	= ▾	value

在混合云中部署应用

EDAS 支持混合云，且可以对混合云中的扩容、网络 and 统一管理等问题提供完整的解决方案。您可以将公共云、本地 IDC 或其它云服务提供商的机器通过专线连通，并添加到 EDAS 公共云的混合云 ECS 集群中。然后即可通过 EDAS 控制台统一部署及管理 HSF、Dubbo 和 Spring Cloud 等应用。对于阿里云公共云中的 ECS 实例，EDAS 提供弹性伸缩功能。

混合云中，应用的部署架构如下图所示。



- 通过高速通道连通阿里云和 IDC 或其它云。
- 创建非阿里云集群，并将阿里云的 ECS 实例和 IDC 或其它云提供商的机器添加到该集群中。
- 将应用部署在该集群的机器上。

使用场景

您的应用系统可能同时有以下需求及问题：

- 云端（公共云）流量有一定的波动性，尤其在某些时间段内会迎来流量高峰。流量可以预估，但难免有偏差，对于扩容的时间及 ECS 实例数量不好控制，且需要提前购买。
- 某些核心业务系统安全性要求较高，想要将这部分应用部署在自己的 IDC（Internet Data Center，互联网数据中心）内。但会面临公共云、IDC 及其它云服务提供商的机器间网络不通的问题，无法统一管理部署在不同环境下的应用。
- 根据实际的业务需求和可用性需求，希望将该应用系统同时部署在多个云服务提供商（多云）的机器上。但将无法实现统一管控，需要人工处理，容易导致误操作。

混合云中，EDAS 主要适用于以下场景：

公共云管理 IDC：在您本地 IDC 通过专线接入阿里云公共云的 VPC 之后，您可以使用公共云 EDAS 管理 IDC 内的应用。

云上应用扩容：EDAS 提供弹性伸缩功能，帮您自动完成阿里云公共云的机器购买和释放，您只需关联您的支付账号，无需提前购置机器。

部署和管理其它云服务提供商的机器：使用 EDAS，您能够将应用部署到除阿里云之外的云服务提供商的机器上，并进行统一管理。

本文档主要介绍公共云管理 IDC 场景。部署和管理其它云服务提供商的机器只需要您将所需机器通过专线接入 EDAS 公共云的 VPC，后续操作和公共云管理 IDC 的场景一致。云上应用扩容请参见弹性伸缩。

说明：目前仅**专业版**和**铂金版**用户可以在混合云中部署应用。

部署流程

1. 准备工作
2. 创建混合云集群
3. 部署应用

准备工作

- 在公共云上创建专有网络。

开通高速通道。

申请物理专线接入，将您本地的 IDC 接入阿里云 VPC。

请确保本地 IDC 中的机器满足如下要求：

- 操作系统：CentOS 7
- 物理机和虚拟机都可以，但不支持 Docker
- 硬件：CPU 和 内存等无特殊要求

创建混合云集群

创建混合云集群包含以下两步：

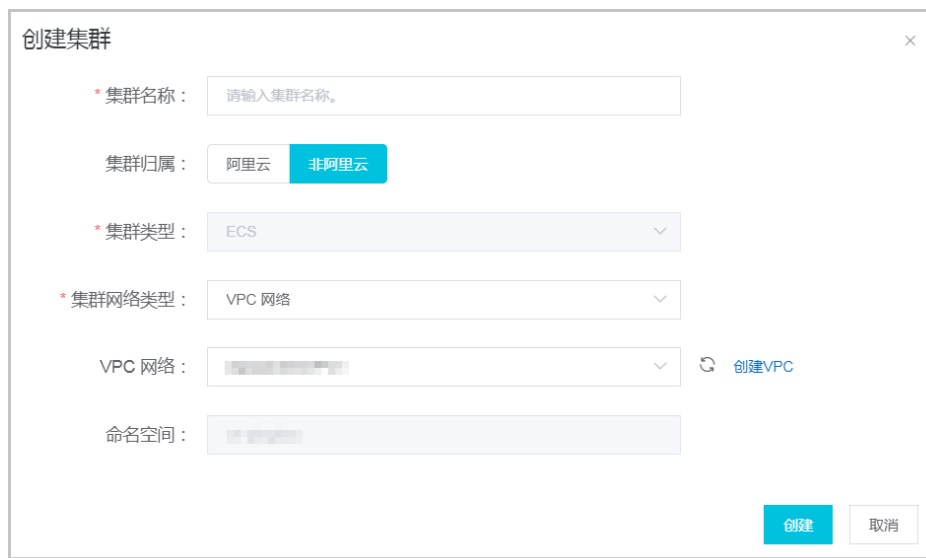
1. 创建集群
2. 添加实例

创建集群

在控制台左侧导航栏中，选择**资源管理** > **集群**，进入集群列表页面。

在集群列表页面上方选择**地域**和**命名空间**，然后在页面右上角单击**创建集群**。

在**创建集群**对话框中输入相关信息，然后单击**创建**。



创建集群对话框包含以下字段：

- * 集群名称：** 请输入集群名称。
- 集群归属：** 阿里云 **非阿里云**
- * 集群类型：** ECS
- * 集群网络类型：** VPC 网络
- VPC 网络：** [下拉菜单] [创建VPC](#)
- 命名空间：** [下拉菜单]

底部有 **创建** 和 **取消** 按钮。

集群字段说明：

- **集群名称：** 输入集群名称。名字仅支持字母、数字、下划线和点且长度不能超过64个字符。
- **集群归属：** 选择**非阿里云**。
- **集群类型：** 默认为 ECS，不可更改。
- **集群网络类型：** 默认为 VPC 网络，不可更改。
- **VPC 网络：** 在下拉菜单中选择集群所属的 VPC。
- **命名空间：** 显示在集群列表页面选择的混合云命名空间，不可配置。

集群创建成功后当前页面右上角会出现**已成功创建集群**的提示，同时新创建的集群会在集群列表中显示。

注意：集群创建成功后，仅仅是一个空集群。只有添加实例后，才能满足应用使用的需求。

添加实例

您需要添加阿里云公共云 ECS 实例和 IDC/其它云提供商的机器。

注意：在混合云集群中仅支持通过命令行手动安装的方式添加主机。

在集群列表页面单击刚刚创建的集群名称。

在集群详情页面中右上角单击**添加 ECS 实例**。

在**添加 ECS 实例**对话框中复制 EDAS Agent 的安装命令。

使用 **root** 账号登录您的**阿里云 ECS 实例或本地 IDC 中的机器**。

复制、执行 EDAS Agent 安装命令。

安装成功后，该实例或机器就添加到集群中了。

开通所需端口

为保证您在混合云中的应用能正常使用 EDAS 提供的相关功能，所以在添加实例后您需要开通以下端口。

- 8182：用于抓取基础监控、调用链监控日志
- 12200~12300：用于完成 RPC 调用
- 65000~65535：Web 端口

应用部署的实例类型不同，需要分别处理。

- 阿里云 ECS 实例：参考添加安全组规则开通相关端口。
- IDC/其它云提供商的机器：请参考对应方案开通相关端口。

结果验证

返回**集群**页面，在集群列表中查看刚创建的**集群状态及实例数**。

集群状态为**正常**则说明集群创建成功。**实例数**和实际添加的主机数量一致，则说明主机添加成功。

单击集群名称，进入集群详情页面。在集群部署信息区域查看添加的**实例名称及集群状态**状态。

集群状态状态为**运行中**说明主机状态正常。

部署应用

混合云集群的类型目前仅支持 ECS 集群，所以在混合云中目前仅支持在 ECS 集群中部署应用。

在混合云集群中部署应用的方式与在 ECS 集群的部署步骤一致，请参考部署 Java 应用到 ECS 集群中的操作步骤。

后续操作

在混合云部署应用后，可以通过 EDAS 对该应用进行应用生命周期管理、监控、诊断、弹性伸缩、限流降级等功能。

说明：在混合云中部署的应用暂时不支持流量管理。