ALIBABA CLOUD

Alibaba Cloud

容器镜像服务

最佳实践

文档版本：20200928

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
`Courier字体`	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.构建容器DevOps

通过容器镜像服务可以便捷地构建基于容器的DevOps开发环境。

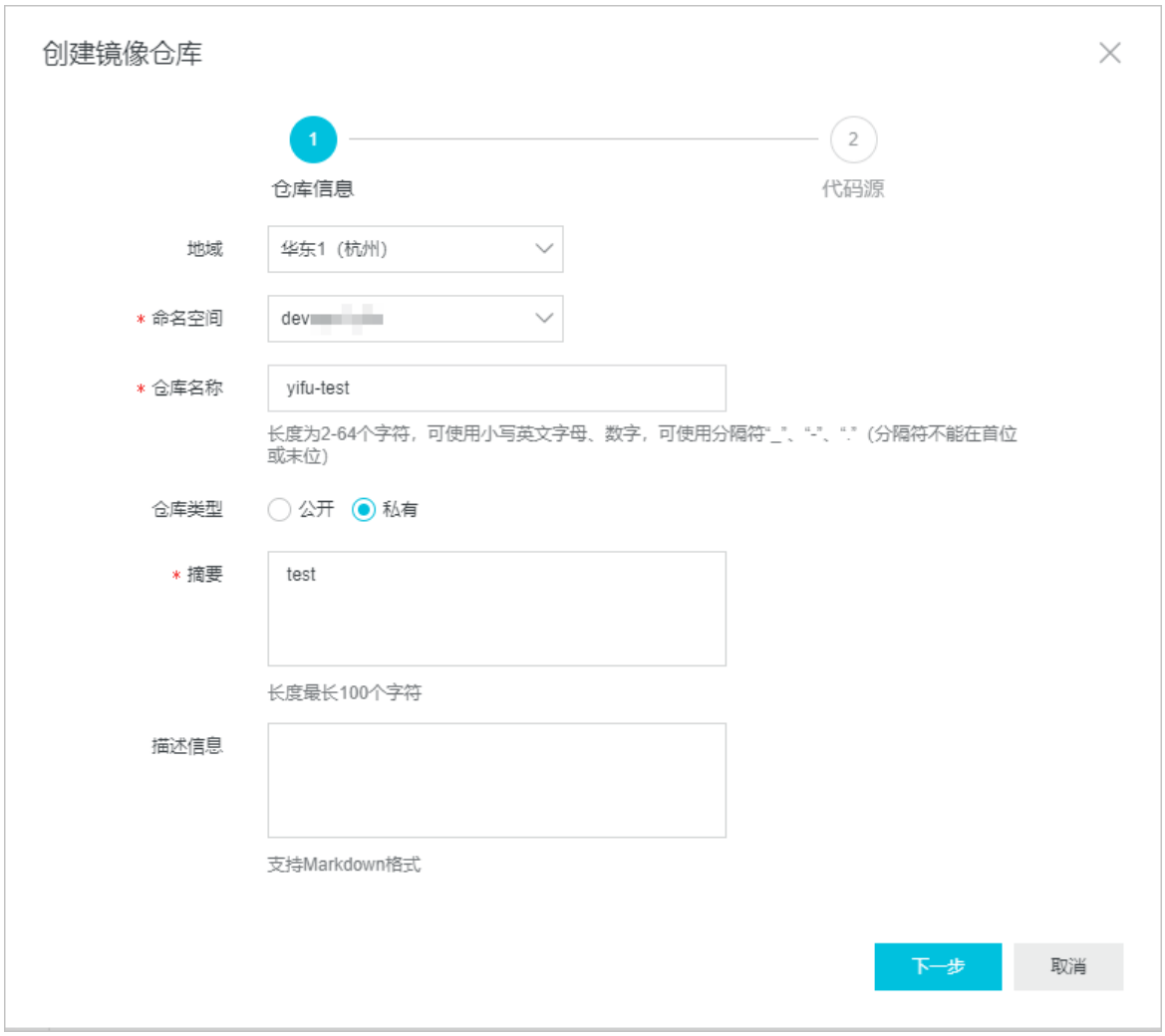
步骤一 创建仓库

1. 如果您还没有开通过阿里云Code，需要单击**绑定账号去开通**。

   > **说明** 默认情况下，如果您的容器镜像服务登录账户已经开通了阿里云Code，将会默认展示您在阿里云Code上的项目。

2. 基于阿里云Code上的项目，创建一个仓库。

   > **说明** 建议在构建设置上选择**代码变更时自动构建镜像**，这样当您在阿里云Code上进行代码修改时，将会触发仓库的自动构建，并将新的镜像推送至阿里云的Registry。



3. 将项目的master分支设置成latest的镜像版本。 当您希望使用这个仓库镜像时，可以直接使用*reigstry.aliyuncs.com/**/dockertest*，无需指定Tag为latest版本，默认使用稳定的master分支构建稳定的latest镜像版本。

步骤二 构建仓库

在仓库的详情页，单击**立即构建**，仓库将使用新添加的两条构建规则进行构建。

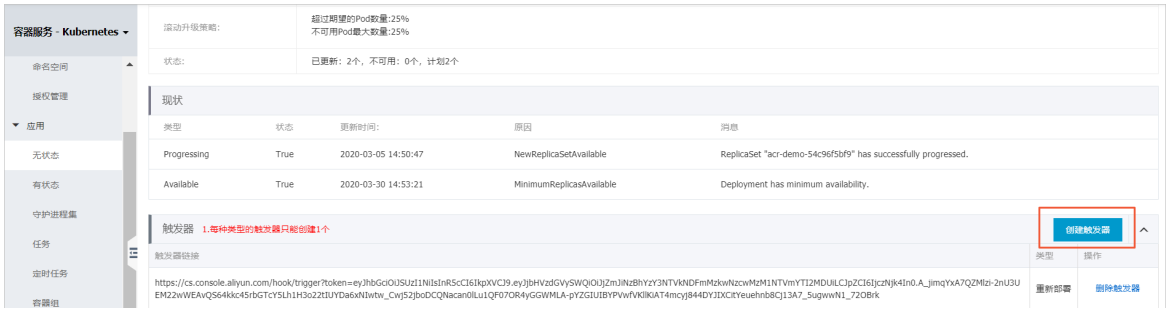
当您在阿里云Code项目的test分支中修改并提交代码后，将触发仓库的第二条构建规则进行自动构建，产生新版本的镜像。

步骤三 绑定仓库触发器

仓库的触发器可以订阅新版本镜像产生的事件，建议可以先用 `http://requestb.in/` 生成一个request URL，绑定在仓库触发器上。当产生新的镜像后，您会看到触发器的访问记录，包括请求的时间、请求的参数以及请求得到的结果。其中请求的参数提供了当前仓库的相关信息。

步骤四 绑定容器服务触发器

1. 登录容器服务管理控制台，单击左侧导航栏中的应用。
2. 选择应用的集群，单击先前创建的应用进入应用详情页。
3. 单击创建触发器，创建一个重新部署类型的触发器，并拷贝触发器URL。



4. 返回镜像信息页面，单击左侧操作栏中的触发器，新建一个触发器，并填入触发器的名称、URL、触发方式。

当阿里云Code上的代码被修改后，容器镜像将会自动构建，并自动触发容器服务上应用的重新部署。

2.使用免密组件拉取容器镜像

aliyun-acr-credential-helper是一个可以在ACK集群中免密拉取ACR默认版或企业版私有镜像的组件。该组件会默认安装在所有ACK集群中。本文列举四个场景介绍如何使用免密组件拉取私有镜像。

前提条件

创建Kubernetes托管版集群

> 注意
>
> - 在Kubernetes资源（例如无状态应用Deployment）模板中配置拉取凭证（imagePullSecret）会导致免密组件失效，如果需使用免密组件，请避免手工配置拉取凭证（imagePullSecret）。
> - 如果部署的Kubernetes资源（例如无状态应用Deployment）使用了自定义的ServiceAccount，需先调整免密组件配置文件中Service-Account字段，使其作用于自定义的ServiceAccount，再进行部署资源操作。
> - 确认Kubernetes集群所属地域与要拉取的镜像所在的地域是否一致，默认配置只可以拉取本地域的镜像。如果需要跨地域拉取镜像，请参见下面的场景三。
> - 使用自定义RAM角色的AccessKey ID和AccessKey Secret进行镜像拉取的情况下，需要把访问密钥写入configMap，这样存在一定的密钥泄露风险。请确定AccessKey ID和AccessKey Secret所属的RAM角色只拥有拉取容器镜像的相关权限。

背景信息

免密组件通过读取ACK集群内的kube-system命名空间中的acr-configuration的配置，进行私有镜像拉取。当前支持针对私有镜像仓库使用以下三种权限之一的策略进行配置：

- 使用默认的ACK Worker RAM角色权限进行拉取（默认策略）。
- 使用自定义RAM角色的AccessKey ID和AccessKey Secret的权限进行拉取。
- 通过配置RAM AssumeRole权限， 使用其他用户的权限进行拉取。

使用免密组件涉及的镜像及集群限制如下：

- 镜像
  - 支持拉取容器镜像服务企业版实例和默认实例中的私有镜像。
  - 支持拉取集群当前用户容器镜像服务中的私有镜像，通过跨账号授权或AccessKey ID和AccessKey Secret配置可以拉取其他用户的私有镜像。
  - 支持跨地域拉取容器镜像服务中的私有镜像。
- 集群
  - 支持集群多命名空间免密拉取。
  - 支持的集群类型：
    - 专有版Kubernetes集群。
    - 托管版Kubernetes集群。

- 支持的集群版本：
  - 专有版Kubernetes集群：高于或等于1.11.2的版本默认支持免密拉取镜像。低于1.11.2版本请您手动升级，请参见升级集群。
  - 托管版Kubernetes集群：所有版本。

升级组件并对组件进行配置

在使用免密组件拉取镜像前，您可能需要升级组件并对组件进行配置，操作步骤如下。

1. 升级组件。
   i. 登录容器服务管理控制台。
   ii. 在控制台左侧导航栏中，单击**集群**。
   iii. 在**集群列表**页面，单击目标集群操作列下的**更多 > 系统组件管理**。
   iv. 在组件列表区域中找到**aliyun-acr-credential-helper**，单击**升级**。
2. 配置组件。
   i. 登录容器服务管理控制台。
   ii. 在控制台左侧导航栏中，单击**集群**。
   iii. 在**集群列表**页面，单击目标集群下的**详情**。
   iv. 在**集群信息**页面左侧导航栏，选择**应用配置 > 配置项**。
   v. 在**配置项**页面，单击目标**配置项名称**，单击**编辑**，修改**配置项**名称。

   如果您没有该配置项，请参见创建配置项。更新配置项，请参见修改配置项。

   以下是一个组件的YAML配置示例。

   - 企业版实例：

```
acr-registry-info: |
    - instanceId: cri-xxx
      regionId: cn-hangzhou
      domains: xxx.com,yyy.com
  watch-namespace: "all"
  service-account: "default"
  expiring-threshold: "15m"
```

   - 默认版实例：

```
acr-registry-info:
  watch-namespace: "all"
  service-account: "default"
  expiring-threshold: "15m"
```

配置项	配置项说明	默认值

配置项	配置项说明	默认值
service-account	使免密组件作用于指定的服务账号。	Default。 说明 如果要配置多个请以逗号分隔，如果设置为“*”，表示支持Namespace下的所有ServiceAccount。
acr-registry-info	容器镜像的实例信息数组，YAML多行字符串格式，每个实例以三元组方式配置。 说明 实例信息三元组： ■ instanceId：实例ID，企业版实例必须配置此项。 ■ regionId：可选，默认为本地地域。 ■ domains：可选，默认为相应实例的所有域名。若要指定个别域名，多个以逗号分隔。	空，表示免密拉取本地地域的默认容器镜像实例仓库。
watch-namespace	期望能免密拉取镜像的Namespace。	Default。 说明 当取值为ALL时，表示期望所有Namespace都能免密拉取。如果需要配置多个Namespace时，以逗号分隔。
expiring-threshold	本地Cache Token过期阈值。	15 m（建议使用15 m）。

场景一：配置跨账号拉取权限

跨账号拉取权限分为：

- **使用角色扮演进行跨账号拉取**：A用户扮演B用户的角色拉取B用户的私有镜像。
- **使用自定义RAM角色的访问密钥进行跨账号私有镜像拉取**：需要使用指定的用户的权限拉取特定的私有镜像。

使用角色扮演进行跨账号拉取

> **说明**
>
> 配置原则如下：
>
> 1. B用户（RAM角色）可以拉取指定私有仓库下的私有镜像（B用户RAM角色有cr.*相关的权限）。
> 2. B用户（RAM角色）允许让A用户下的ACK Worker RAM角色扮演（信任策略）。
> 3. A用户下的ACK集群的Worker RAM角色有扮演B用户（RAM角色）的权限（AliyunAssumeRoleAccess）。
> 4. 设置A用户下的Worker RAM角色扮演B用户（configMap中的assumeRoleARN）。

1. **创建B用户的受信实体为阿里云账号类型的RAM角色，并确保该RAM角色拥有拉取B用户私有镜像的权限。**
   i. **创建RAM角色。具体操作请参见创建可信实体为阿里云账号的RAM角色。**

ii. 自定义RAM角色权限策略内容，确保该RAM角色拥有拉取B用户私有镜像的权限。具体操作请参见修改自定义策略内容。

> 注意 请确保该RAM角色有cr.*的相关权限。

```
{
"Action": [
 "cr:GetAuthorizationToken",
 "cr:ListInstanceEndpoint",
 "cr:PullRepository"
],
"Resource": [
"*"
],
"Effect": "Allow"
}
```

添加字段位置如下图所示。

2. 在创建的B用户的RAM角色上配置信任策略，允许A用户（需要拉取B用户下的私有镜像）的ACK集群的Worker RAM角色来扮演B用户。在策略内容Principal处填写A用户的ACK集群的Worker RAM角色的ARN。

   i. 查看A用户的ACK集群的Worker RAM角色的ARN信息。查看ARN信息具体步骤，请参见如何查看RAM角色的ARN?

ii. 配置信任策略。

a. 在RAM控制台左侧导航栏，单击RAM角色管理，找到目标RAM角色并单击RAM角色名称。

b. 在RAM角色基本信息页，单击信任策略管理页签，然后使用A用户的Worker RAM角色的ARN修改信任策略。

3. 确认A用户的Worker RAM角色拥有AssumeRole权限。

具体操作步骤，请参见查看权限策略基本信息。

4. 调整组件配置，新增assumeRoleARN配置。

配置内容为B用户Worker RAM角色的ARN。查看ARN信息具体步骤，请参见如何查看RAM角色的ARN? YAML配置示例如下。配置步骤，请参见上述配置组件。

```
data:
  service-account: "default"
  watch-namespace: "all"
  expiring-threshold: "15m"
  notify-email: "cs@aliyuncs.com"
  acr-registry-info: |
  - instanceId: ""
    regionId: cn-beijing
    domains: registry.cn-beijing.aliyuncs.com
    assumeRoleARN: acs:ram::.*:role/kubernetesworkerrole-test
```

使用RAM用户的访问密钥拉取跨账号私有镜像

1. 创建一个RAM用户并为该用户添加cr镜像拉取权限。有关配置cr镜像拉取权限的具体操作步骤，请参见使用角色扮演进行跨账号拉取步骤1。
2. 配置kube-system命名空间中的配置项acr-configuration，并填入创建的RAM用户的访问密钥AccessKey ID和AccessKey Secret。此种方式可以使免密组件直接使用创建的RAM用户拉取私有镜像。查看访问密钥的具体步骤，请参见查看访问密钥基本信息。

   配置示例如下。配置步骤请参见上述配置组件。

```
data:
  service-account: "default"
  watch-namespace: "all"
  expiring-threshold: "15m"
  notify-email: "cs@aliyuncs.com"
  acr-registry-info: |
  - instanceId: ""
    customAccessKey: "xxxxx" // 此处填写刚创建的RAM用户的AccessKey ID。
    customAccessKeySecret: "xxxxxx" // 此处填写刚创建的RAM用户的AccessKey Secret。
```

场景二：配置当前账号拉取权限

您需要检查当前账号ACR拉取的权限。

1. 登录容器服务管理控制台。
2. 在控制台左侧导航栏中，单击集群。
3. 在集群列表页面中，单击目标集群名称或者目标集群右侧操作列下的详情。
4. 在集群信息页面，单击集群资源页签，单击Worker RAM角色右侧链接。
5. 在RAM角色基本信息的权限管理页签，单击目标权限策略名称。
6. 单击修改策略内容。
7. 在策略内容区域增加以下字段后，单击确定。

```
{
  "Action": [
    "cr:Get*",
    "cr:List*",
    "cr:PullRepository"
  ],
  "Resource": "*",
  "Effect": "Allow"
}
```

添加字段位置如下图所示。

说明 您还需要检查Pod中镜像字段的值是否存在于ACR仓库里面。

- 如果存在，但不在当前账号下，您可能需要使用跨账号镜像拉取（参考场景一）。
- 如果存在，且在当前账号下，您可以提交工单。
- 如果不存在，您需要确认该镜像是否存在于阿里云容器镜像服务上。如果不存在，则需要同步外部镜像到阿里云容器镜像服务商，或者变更改镜像为公网可拉取状态。

场景三：配置跨地域拉取镜像权限

如果需要拉取的镜像与当前ACK集群不属于同一地域的时候，需要修改配置项acr-configuration中的configMap。

例如，默认仓库同时拉取北京地域与杭州地域的镜像，配置如下。配置步骤请参见上述配置组件。

```
data:
  service-account: "default"
  watch-namespace: "all"
  expiring-threshold: "15m"
  notify-email: "cs@aliyuncs.com"
  acr-registry-info: |
  - instanceId: ""
    regionId: cn-beijing
  - instanceId: ""
    regionId: cn-hangzhou
```

场景四：同时拉取默认实例和企业实例的私有镜像

如需同时拉取默认实例和企业版实例的私有镜像，请按照以下方式修改配置项acr-configuration中的configMap。配置步骤请参见上述配置组件。

```
data:
  service-account: "default"
  watch-namespace: "all"
  expiring-threshold: "15m"
  notify-email: "cs@aliyuncs.com"
  acr-registry-info: |
  - instanceId: ""
  - instanceId: "cri-xxxx"
```

3.在Dockerfile中使用多阶段构建打包Java应用

多阶段构建指在Dockerfile中使用多个FROM语句，每个FROM指令都可以使用不同的基础镜像，并且是一个独立的子构建阶段。使用多阶段构建打包Java应用具有构建安全、构建速度快、镜像文件体积小等优点，本文以Github上的Java Maven项目为例，结合阿里云容器镜像服务（ACR）的镜像构建服务，介绍如何进行多阶段构建。

前提条件

- 开通阿里云容器镜像服务。
- 请准备一个托管在Github、Gitlab、Bitbucket或者阿里云Code平台上的Java源代码仓库。

> **说明** 您可以拷贝并托管位于Github上的一个简单的Java Maven项目来体验整个流程。

背景信息

镜像构建的通用问题

镜像构建服务使用Dockerfile来帮助用户构建最终镜像，但在具体实践中，存在一些问题：

- Dockerfile编写有门槛

  开发者（尤其是Java）习惯了语言框架的编译便利性，不知道如何使用Dockerfile构建应用镜像。

- 镜像容易臃肿

  构建镜像时，开发者会将项目的编译、测试、打包构建流程编写在一个Dockerfile中。每条Dockerfile指令都会为镜像添加一个新的图层，从而导致镜像层次深，镜像文件体积特别大。

- 存在源码泄露风险

  打包镜像时，源代码容易被打包到镜像中，从而产生源代码泄漏的风险。

多阶段构建优势

针对Java这类的编译型语言，使用Dockerfile多阶段构建，具有以下优势：

- 保证构建镜像的安全性

  当您使用Dockerfile多阶段构建镜像时，需要在第一阶段选择合适的编译时基础镜像，进行代码拷贝、项目依赖下载、编译、测试、打包流程。在第二阶段选择合适的运行时基础镜像，拷贝基础阶段生成的运行时依赖文件。最终构建的镜像将不包含任何源代码信息。

- 优化镜像的层数和体积

  构建的镜像仅包含基础镜像和编译制品，镜像层数少，镜像文件体积小。

- 提升构建速度

  使用构建工具（Docker、Buildkit 等），可以并发执行多个构建流程，缩短构建耗时。

步骤一：使用多阶段构建Dockerfile

以Java Maven项目为例，在Java Maven项目中新建Dockerfile文件，并在Dockerfile文件添加以下内容。

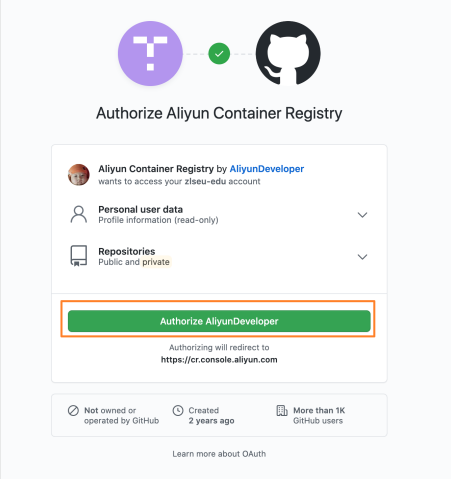
> **说明** 该Dockerfile文件使用了二阶段构建。
>
> - 第一阶段：选择Maven基础镜像（Gradle类型也可以选择相应Gradle基础镜像）完成项目编译，拷贝源代码到基础镜像并运行RUN命令，从而构建Jar包。
> - 第二阶段：拷贝第一阶段生成的Jar包到OpenJDK镜像中，设置CMD运行命令。

```
# First stage: complete build environment
FROM maven:3.5.0-jdk-8-alpine AS builder

# add pom.xml and source code
ADD ./pom.xml pom.xml
ADD ./src src/

# package jar
RUN mvn clean package

# Second stage: minimal runtime environment
From openjdk:8-jre-alpine

# copy jar from the first stage
COPY --from=builder target/my-app-1.0-SNAPSHOT.jar my-app-1.0-SNAPSHOT.jar

EXPOSE 8080

CMD ["java", "-jar", "my-app-1.0-SNAPSHOT.jar"]
```

步骤二：绑定源代码仓库

在容器镜像服务控制台，绑定您托管的代码仓库，以下内容以GitHub为例。

1. 登录容器镜像服务控制台。
2. 在顶部下拉菜单中选择所需地域，在左侧导航栏中选择**默认实例 > 代码源**。
3. 在GitHub栏单击**绑定账号**，在弹出的对话框中单击**点击前往源代码仓库登录**，跳转到GitHub。
4. 在授权界面，单击**Authorize AliyunDeveloper**。



返回至代码源界面，GitHub栏中显示已绑定，表示绑定成功。

步骤三：新建镜像仓库

1. 在左侧导航栏中选择默认实例 > 镜像仓库，然后单击创建镜像仓库。
2. 设置镜像仓库信息。

配置项	描述
地域	镜像仓库所在区域，目前阿里云容器镜像服务已经全球23个区域开服。
命名空间	仓库所属命名空间。一个镜像仓库必须且仅属于一个命名空间。一个命名空间下可以包含多个镜像仓库。
仓库名称	输入仓库名称。
仓库类型	仓库类型分为公开和私有。无论公开还是私有类型仓库，推送镜像都需要先进行登录。 ○ 公开：拉取镜像时可以免登录，直接通过网络拉取。 ○ 私有：必须要通过Docker客户端进行登录，才能拉取镜像。
摘要	简要描述信息。

配置项	描述
描述信息	完整描述信息。

3. 单击**下一步**，设置代码源。
   - 代码源：将代码源设为**GitHub**，选择步骤二：绑定源代码仓库中绑定的源代码仓库。
   - 构建设置：
     a. **代码变更时自动构建镜像**：当分支有代码提交后会自动触发构建规则。
     b. **海外机器构建**：构建时会在海外机房构建，构建成功后推送到指定地域。
     c. **不使用缓存**：每次构建时会强制重新拉取基础依赖镜像，可能会增加构建时间。
4. 单击**创建镜像仓库**。
   选择**默认实例** > **镜像仓库**，在**镜像仓库**界面查找到创建的仓库，表示创建镜像仓库成功。

步骤四：执行构建

1. 在左侧导航栏中选择**默认实例** > **镜像仓库**，单击仓库名称或目标仓库**操作**列的**管理**，进入仓库详情页面。
2. 单击左侧导航栏中的**构建**，在**构建规则设置**区域的右侧单击**添加规则**。
3. 设置构建规则。

配置项	描述
类型	定义了推送代码到托管仓库，触发构建规则的事件。目前有Branch和Tag两种类型的推送。
Branch/Tag	设置构建的代码分支。
Dockerfile目录	设置Dockerfile文件所在的目录。这里的目录指的是相对目录，以代码分支的根目录为父目录。
Dockerfile文件名	设置Dockerfile文件名，默认为Dockerfile。
镜像版本	设置镜像版本。

4. 单击**确认**，返回至构建页面。

5. 在**构建规则设置**区域中，找到创建的规则，单击目标规则对应**操作**列的**立即构建**。
   在**构建日志**区域中找到构建记录，当构建状态显示**成功**，表示构建成功。

执行结果

- 查看构建的镜像

  在左侧导航栏中选择**默认实例 > 镜像仓库**，单击仓库名称或目标仓库**操作**列的**管理**，进入仓库详情页面。单击**镜像版本**，查看构建的镜像。

- 查看镜像层数

  使用Docker客户端登录镜像仓库，执行以下命令，拉取构建的镜像并检查镜像层数。可以发现仅包含了第一阶段编译Jar包，大小增加有限，镜像层数控制在四层。

```
docker pull registry.cn-hangzhou.aliyuncs.com/docker-builder/java-multi-stage:master


docker inspect registry.cn-hangzhou.aliyuncs.com/docker-builder/java-multi-stage:master | jq -s .[0]
[0].RootFS
{
 "Type": "layers",
 "Layers": [
  "sha256:f1b5933fe4b5f49bbe8258745cf396afe07e625bdab3168e364daf7c956b6b81",
  "sha256:9b9b7f3d56a01e3d9076874990c62e7a516cc4032f784f421574d06b18ef9aa4",
  "sha256:edd61588d12669e2d71a0de2aab96add3304bf565730e1e6144ec3c3fac339e4",
  "sha256:2be89a7ccd49878fb5f09d6dfa5811ce09fc1972f0a987bbb5a006992aa3dff3"
 ]
}
```

- 运行镜像，查看运行结果

  使用Docker客户端登录镜像仓库，执行以下命令，运行镜像，查看运行结果。

```
docker run -ti registry.cn-hangzhou.aliyuncs.com/docker-builder/java-multi-stage:master
Hello World!
```