



黑客派

# Solo 开源博客 - 修改源码构建自己 docker 镜像

作者: [T-Aoker](#)

原文链接: <https://hacpai.com/article/1577411866113>

来源网站: [黑客派](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

# 1. Solo 开源博客介绍

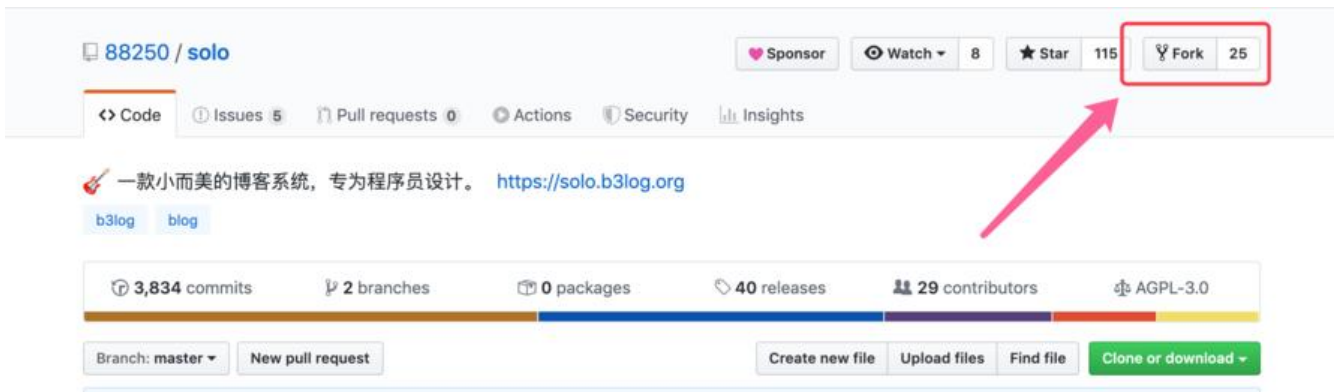
Solo 是一款小而美的开源博客系统，专为程序员设计。  
源码仓库：

- GitHub: <https://github.com/88250/solo>

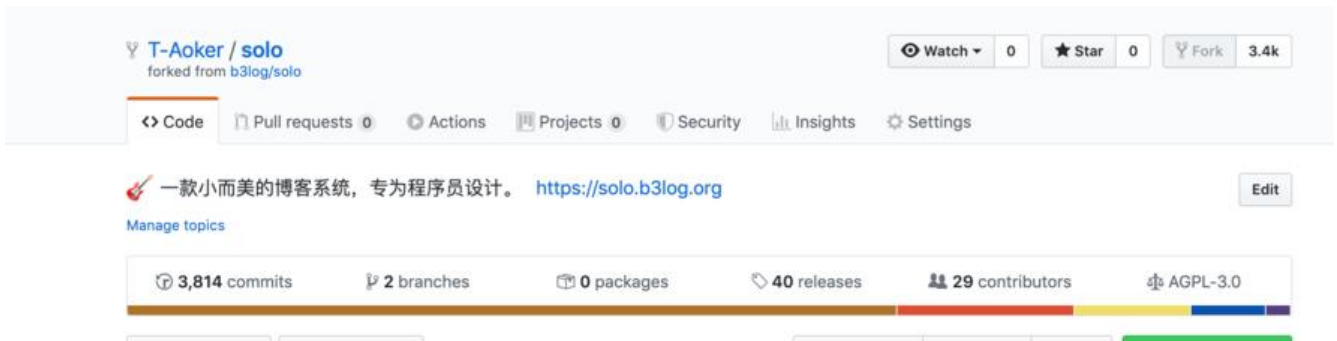
# 2. fork 源码到自己的 GitHub

- 打开 Solo 仓库, <https://github.com/88250/solo>

点击 fork 按钮, 拉取源码到自己的 GitHub 上



- 执行 fork 之后, 就会在自己的 GitHub 仓库看到克隆的项目



- 然后克隆自己的 solo 项目到本地, 即可开始快乐的修改 solo 源码了

# 3. 开始定制自己的 solo 之旅

如果我们想要修改 solo 源码, 第一步当然是要在本地把 solo 项目跑起来啦, 参考[搭建Solo开发环境](#)

## 创建 solo 数据库

```
create database solo DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_general_ci;
```

## 运行启动类

在 IDE 中直接运行 `org.b3log.solo.Server` 类。

启动成功后, 就可以通过 `localhost:8080` 访问了

## 4. 通过本地 solo 项目构建 docker 镜像

docker 镜像的构建，会依照项目根部的 Dockerfile 进行构建

如 solo 的 Dockerfile 文件如下

```
FROM maven:3-jdk-8-alpine as MVN_BUILD

WORKDIR /opt/solo/
ADD ./tmp
RUN cd /tmp && mvn package -DskipTests -Pci -q && mv target/solo/* /opt/solo/ \
&& cp -f /tmp/src/main/resources/docker/* /opt/solo/

FROM openjdk:8-alpine
LABEL maintainer="Liang Ding<d@b3log.org>"

WORKDIR /opt/solo/
COPY --from=MVN_BUILD /opt/solo/ /opt/solo/
RUN apk add --no-cache ca-certificates tzdata

ENV TZ=Asia/Shanghai
EXPOSE 8080

ENTRYPOINT [ "java", "-cp", "lib/*:", "org.b3log.solo.Server" ]
```

- 第一步，先进入本地克隆的 solo 项目根目录，与 Dockerfile 文件同级
- 第二步，直接执行构建 docker 镜像的命令，构建自己的 solo docker 镜像

```
docker build -t aoker/mysolo:dev ,
```

- 第三步，执行 docker run 命令，运行自己的镜像就 OK 了

```
docker run -d -p 8080:8080 --name solo \
--env RUNTIME_DB="MYSQL" \
--env JDBC_USERNAME="root" \
--env JDBC_PASSWORD="123456" \
--env JDBC_DRIVER="com.mysql.cj.jdbc.Driver" \
--env JDBC_URL="jdbc:mysql://docker.for.mac.host.internal:3306/solo?useUnicode=yes&characterEncoding=UTF-8&useSSL=false&serverTimezone=UTC" \
aoker/mysolo --listen_port=8080 --server_scheme=http --server_host=localhost --server_port=8080
```

```
--env JDBC_URL="jdbc:mysql://docker.for.mac.host.internal:3306/solo?useUnicode=yes&characterEncoding=UTF-8&useSSL=false&serverTimezone=UTC" \
```

可以看到我 MySQL 的配置，使用的是 docker.for.mac.host.internal，而非使用的 localhost，这是什么呢？

这是因为我使用的 Mac 系统安装的 docker for Mac，docker for Mac 在 Mac 系统上的运行机制导的

docker for Mac 比较特殊，docker 本身在 Mac 也是启动的虚拟机，并非像在 Linux 上面运行在宿机的一个进程，所以 Mac 的 docker host 模式，容器共用的端口是与 docker 虚拟机共用的，并非主机

此时需要在容器中配置宿主机服务的 ip 由 localhost 改为

Mac 下 Docker 容器访问宿主机端口

Mac 下的 docker 其实是虚拟机，所以无法直接访问宿主机

Docker for Mac v 17.12 to v 18.02

直接使用: `docker.for.mac.host.internal`

Docker for Mac v 17.06 to v 17.11

直接使用: `docker.for.mac.localhost`

Docker for Mac 17.05 and below

设置主机主址别名: `sudo ifconfig lo0 alias 123.123.123.123/24`，然后使用: `123.123.123.123`

## 5. 通过腾讯（阿里）云镜像仓库管理自己的镜像

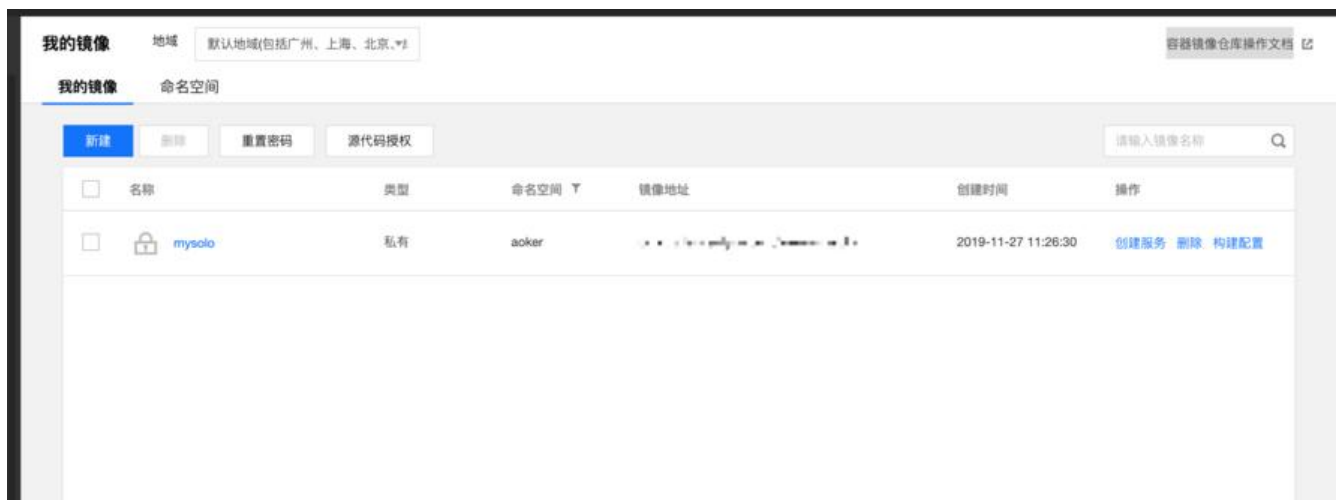
我们修改源码后，最终会构建镜像，部署到云服务器上，这时我们可以通过云服务商提供的镜像仓库能，管理自己的镜像，并在服务器上进行部署更新。

- 这次我们以腾讯云为例，我们可以通过腾讯云的镜像仓库，新建自己的私有仓库，来管理自己的镜像。

腾讯云镜像仓库地址: [腾讯云镜像仓库](#)

腾讯云镜像仓库使用文档: [腾讯云镜像仓库文档](#)

进入镜像仓库界面如图:



- 接下来就是进行镜像管理的操作

## 操作步骤

### 开通镜像仓库

说明:

首次使用镜像仓库的用户，需要先开通镜像仓库。

1. 登录容器服务控制台，选择左侧导航栏中的【镜像仓库】>【[我的镜像](#)】。

2. 根据以下提示填写相关信息，并单击【开通】进行初始化。如下图所示：

容器服务

我的镜像 默认地域(包括广州、上海...)

初次使用私有仓库(“我的镜像”), 需要先进行初始化。命名空间将用于分类您的容器镜像。密码用于通过docker login来登陆腾讯云容器服务。

用户名

密码

确认密码

开通

- **用户名**：默认是当前用户的账号，是您登录到腾讯云 docker 镜像仓库的身份。
- **密码**：是您登录到腾讯云 docker 镜像仓库的凭证。

## 创建命名空间

1. 选择左侧导航栏中的【镜像仓库】>【 [我的镜像](#)】，进入“我的镜像”页面。
2. 在“我的镜像”页面中，选择【命名空间】页签并单击【新建】。如下图所示：

我的镜像 默认地域(包括广州、上海...)

命名空间

新建

命名空间	仓库数目	创建时间	操作
命名空间列表为空			

共 0 项 每页显示行 20 1 / 1 页

3. 在弹出的“新建命名空间”窗口中，输入命名空间名并单击【提交】。如下图所示：

说明：

命名空间名称全局唯一，若您希望使用的命名空间名称已被其他用户使用，请尝试其他适用的命名空间名称。

## 新建命名空间



名称

最长30个字符，只能包含小写字母、数字及分隔符("、"\_"、"-")，且不能以分隔符开头、结尾或连续

提交

取消

## 创建镜像

1. 选择左侧导航栏中的【镜像仓库】>【[我的镜像](#)】，进入“我的镜像”页面。
2. 在“我的镜像”页面，单击镜像列表页上方的【新建】。如下图所示：

我的镜像

默认地域(包括广州、上海...)

[容器镜像仓库操作文档](#)

我的镜像

命名空间

新建

删除

重置密码

源代码授权

请输入镜像名称



<input type="checkbox"/>	名称	类型	命名空间	镜像地址	创建时间	操作
--------------------------	----	----	------	------	------	----

您的镜像仓库列表为空，您可以[立即创建](#)

共 0 项

每页显示行 20

1 / 1 页

3. 输入镜像名称和描述，然后【提交】。

说明：

命名空间将用于分类容器镜像，也是您创建的私人镜像地址的前缀，本文以 `tkefiletest` 为例。

## 新建镜像仓库



名称

test

最长为200个字符，只能包含小写字母、数字及分隔符("、"\_"、"-")，且不能以分隔符开头或结尾

类型

私有

命名空间

tkefiletest

描述

最长为1000个字符

提交

取消

## 推送镜像到镜像仓库

### 登录到腾讯云 registry

1. 在终端替换以下命令中的相关信息并执行，登录腾讯云 registry。

```
$ sudo docker login --username=[username] ccr.ccs.tencentyun.com
```

**username:** 腾讯云账号，开通时已注册。

2. 输入密码后即登录完成。

### 上传镜像

根据以下提示替换命令中的相关信息并执行，上传镜像。

```
$ sudo docker tag [ImageId] ccr.ccs.tencentyun.com/[namespace]/[ImageName]:[镜像版本号]
$ sudo docker push ccr.ccs.tencentyun.com/[namespace]/[ImageName]:[镜像版本号]
```

- **ImageId 和镜像版本号:** 根据已有镜像信息补充。
- **namespace:** 开通镜像仓库时填写的命名空间。
- **ImageName:** 在控制台创建的镜像名称。

### 下载镜像

1. 执行以下命令登录到镜像仓库，需输入在 [开通镜像仓库](#) 中已设置的密码。

```
$ sudo docker login --username=[username] ccr.ccs.tencentyun.com
```

2. 替换命令中的相关信息并执行，下载镜像。

```
$ sudo docker pull ccr.ccs.tencentyun.com/[namespace]/[ImageName]:[镜像版本号]
```

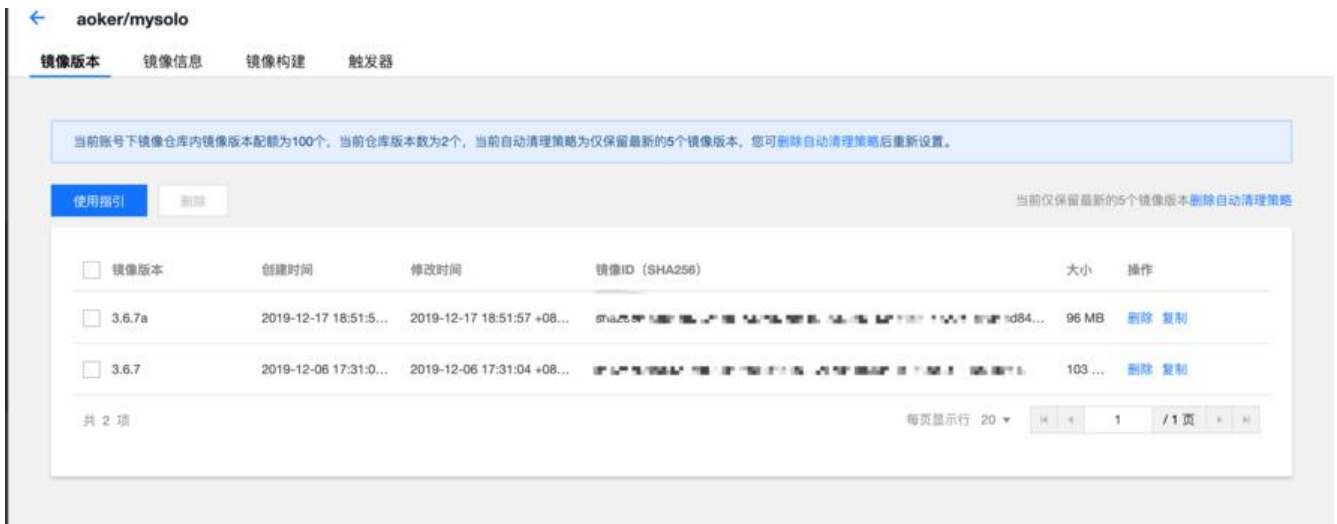
## 删除镜像

1. 选择左侧导航栏中的【镜像仓库】>【[我的镜像](#)】，进入“我的镜像”页面。
2. 在“我的镜像”页面，选择需删除镜像所在行右侧【删除】。
3. 在弹出的“删除镜像仓库”窗口中，单击【确定】即可 **删除该镜像所有版本**。如下图所示：



## 6. 利用腾讯云在线进行构建镜像

1. 进入我们刚刚新建的镜像仓库，如图所示，会列出我们之前已经构建好的镜像版本



2. 我们进入镜像构建-构建配置, 进行设置镜像构建, 关联我们的源码仓库, 授权腾讯云访问我们的github




镜像地址 `ccr.ccs.tencentyun.com/undefined`

代码源



Organization

T-Aoker 

Repository

<input type="radio"/> color-scheme	公有仓库
<input type="radio"/> CS-Notes	公有仓库
<input type="radio"/> darktable	公有仓库
<input type="radio"/> hexo-theme-matery	公有仓库
<input type="radio"/> ik-analyzer-solr6	公有仓库
<input type="radio"/> jeecg-boot	公有仓库
<input type="radio"/> Learning-Linux	公有仓库
<input checked="" type="radio"/> my-solo	私有仓库
<input type="radio"/> solo	公有仓库
<input type="radio"/> solo-1	公有仓库

触发方式

满足以下任意条件即触发构建镜像

3. 我们配置好后就可以看到配置好的配置

构建配置  

代码源 <https://www.github.com/T-Aoker/my-solo>

触发规则 创建新Tag时触发

Dockerfile路径 Dockerfile

立即构建

构建记录

开始时间	状态	Git commit	生成的镜像版本	来源	耗时	操作
2019-12-17 18:33:23	构建成功 	9249fb315849b063421645b32f1c885bf5940207	3.6.7a	源码构建	18分26秒	-
2019-12-06 17:13:38	构建成功 	d630142365ddc30ae9029238c00cd0ab4d788389	3.6.7	源码构建	17分19秒	-

原文链接: [Solo 开源博客 - 修改源码构建自己 docker 镜像](#)

4. 点击立即构建，选择构建版本，并可以在线进行构建镜像，构建成功后，并会在下面生成构建记录并创建对应版本镜像



● 当然除了我们手动构建镜像，还可以设置触发器，进行自动构建镜像，这样就可以在相应触发条件成时，自动生成对应镜像

通过触发器，可以在每次生成新的Tag（镜像版本）时，自行执行动作，如：自动更新使用该镜像仓库的朋

触发器名称

英文字母开头，2-64个字符以内，支持a-z、A-Z、0-9、-、\_

触发条件

全部触发

镜像仓库内，有新的Tag生成，或Tag发生更新时，触发动作

指定Tag触发

镜像仓库内，有指定Tag生成或更新时，触发动作

正则触发

镜像仓库内，有符合正则表达式的Tag生成或更新时，触发动作

触发动作

更新容器的镜像

需指定更新的服务,以及该服务下的指定容器镜像

选择服务/容器

请选择容器 ▾

保存

取消

[添加触发器](#)

5. 构建好的镜像如何部署到自己的服务器呢？

只需要执行如下命令即可，拉取镜像到服务器上

```
$ sudo docker login --username=[username] ccr.ccs.tencentyun.com
```

替换命令中的相关信息并执行，下载镜像。

```
$ sudo docker pull ccr.ccs.tencentyun.com/[namespace]/[ImageName]:[镜像版本号]
```

然后运行镜像，并可run起来了

```
docker run -d --name mysolo --network=host \  
--env RUNTIME_DB="MYSQL" \  
--env JDBC_USERNAME="root" \  
--env JDBC_PASSWORD="123456" \  
--env JDBC_DRIVER="com.mysql.cj.jdbc.Driver" \  
--env JDBC_URL="jdbc:mysql://127.0.0.1:3306/solo?useUnicode=yes&characterEncoding=UT-8&useSSL=false&serverTimezone=UTC" \  
-v /dockerData/solo/skins:/opt/solo/skins \  
ccr.ccs.tencentyun.com/aoker/mysolo:3.6.7 --listen_port=8080 --server_scheme=http --server host=服务器IP --server_port=8080
```