

# DAY19 微服务应用实战之服务治理

## 1 打卡任务

作业：

创建两个微服务应用：webapp和helloworld，webapp用来访问helloworld，然后再创建helloworld的灰度版本，进行灰度发布。本作业操作步骤较多，请按操作指导一步步执行。

打卡：

验证灰度发布成功，并截图。

## 2 准备工作

1、切换区域为“华东-上海二”。

2、已拥有DevCloud账号和密码，并导入下面两个工程到自己的DevCloud账号下：

<https://github.com/servicestage-demo/helloclient.git>

<https://github.com/servicestage-demo/helloworld.git>

步骤1：登录[DevCloud代码托管](#)，点击“新建仓库”，进入新建代码仓库页面。

步骤2：在新建代码仓库页面，点击“导入仓库”tab页，在“源仓库路径”中输入“https://github.com/servicestage-demo/helloclient.git”，然后点击“确定”按钮，导入成功。

步骤3：重复步骤1和2，导入<https://github.com/servicestage-demo/helloworld.git>工程。

3、已有CCE集群

## 3 创建 ServiceComb 微服务应用 helloworld

1、登录ServiceStage，选择 总览 > 快捷操作 > 创建ServiceComb应用。

2、运行环境选择Java8，修改应用显示名称为helloworld（**自动生成的应用名称不要修**

改)，应用来源选择“源码仓库”，代码源来源选择DevCloud，输入密码进行授权（如已授权则这一步可省略），仓库名称选择准备工作导入的helloworld项目，分支选择master，版本为1.0.0。



The screenshot shows a configuration form for an application. Key fields include:

- 应用类型 (Application Type):** ServiceComb
- 运行环境 (Runtime Environment):** Tomcat8, Java8 (selected)
- 应用名称 (Application Name):** servicestagedkdsi
- 应用显示名称 (Application Display Name):** helloworld
- 应用来源 (Application Source):** Jar包, 源码仓库 (selected), 模板
- 代码源来源 (Code Source):** DevCloud, GitHub, Gitee, Bitbucket, GitLab
- 已绑定账户 (Bound Account):** [Redacted]
- 命名空间 (Namespace):** Demo
- 仓库名称 (Repository Name):** helloworld
- 分支 (Branch):** master
- 版本 (Version):** 1.0.0

3、下一步，部署系统选择CCE，并选择已有集群（如果没有可以在更换配置，选择新创建）



The screenshot shows a deployment configuration form. Key fields include:

- 配置方式 (Configuration Method):** 使用已有 (selected), 新创建
- 部署集群 (Deployment Cluster):** testoneday
- 集群命名空间 (Cluster Namespace):** default

4、选择微服务引擎，默认用Cloud Service Engine，实例数量选择1个（节省资源），应用端口8081（与代码监听一致）

部署系统

云容器引擎CCE

Beta!

云容器实例CCI

集群

已有Kubernetes集群 **testoneday** [更换配置](#)

实例数量

1

10

1

个

资源配额

Micro: 5G Storage, 0.5 CPU, 1G Memory

应用端口

8081

微服务引擎

Cloud Service Engine

[购买微服务引擎](#)

5、下一步，确认提交，等待应用部署成功

应用helloworld正在创建中...

[返回应用管理](#)
[查看应用详情](#)

就绪

2019/02/16 09:46:07 GMT+08:00 查询 DCS 实例信息 完成  
2019/02/16 09:46:07 GMT+08:00 查询 RDS 实例信息 完成

构建

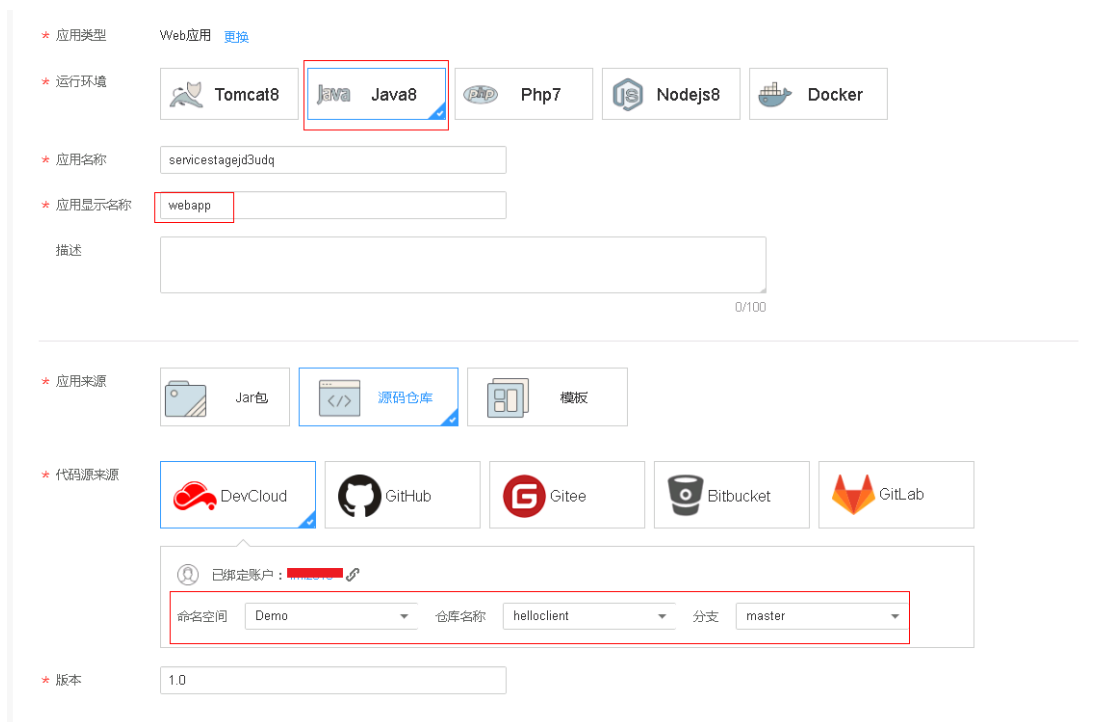
2019/02/16 09:46:07 GMT+08:00 构建应用 执行中  
2019/02/16 09:46:07 GMT+08:00 创建构建任务 完成  
2019/02/16 09:46:07 GMT+08:00 执行构建任务 完成  
2019/02/16 09:46:07 GMT+08:00 查询构建状态 执行中  
2019/02/16 09:46:07 GMT+08:00 查询构建状态 完成  
2019/02/16 09:46:07 GMT+08:00 等待构建完成 执行中

## 4 创建 Web 应用 webapp 用于访问刚才的 helloworld

- 1、在ServiceStage菜单点击 应用管理 > 创建应用，选择 Web应用 > 创建Web应用。



- 2、运行环境选择Java8，修改应用显示名称为webapp（自动生成的应用名称不要修改），应用来源选择“源码仓库”，代码源来源选择DevCloud，选择准备工作导入的helloclient项目，分支选择master。



- 3、下一步，部署系统：CCE，集群选择和上面相同的集群，实例数改为1，端口8080

（与代码监听保持一致）

**\* 部署系统**

云容器引擎CCE

Beta! 云容器实例CCI

**\* 集群**

已有Kubernetes集群 **testoneday** 更换配置

**\* 实例数量**

1 10

**\* 资源配额** ?

Micro: 5G Storage, 0.5 CPU, 1G Memory

**\* 应用端口**

8080

- 4、选择负载均衡，如果已存在则直接跳过；如果没有负载均衡，则点击“去配置”，然后选择“新创建”，弹性公网IP选择“使用已有”，如果没有可以通过查看弹性公网IP去进行解绑，解绑后返回刷新选择后点击“确定”即可。

**\* 资源配额** ?

Micro: 5G Storage, 0.5 CPU, 1G Memory

**\* 应用端口**

8080

**\* 访问方式**

新建负载均衡 您有待完善的配置信息，去配置>>

**\* 域名**

自动生成

自动生成的域名仅有7天有效期，您可以选择绑定域名或应用创建后绑定域名。

**\* JVM参数**

比如-Xms256m -Xmx1024m，多个参数以空格分隔，不填则使用默认值

**更换访问方式配置**

**\* 配置方式**

使用已有 新创建

**\* 负载均衡**

elb-o84p 查看负载均衡

**\* 弹性公网IP**

新创建 使用已有

119.3.27.65 查看弹性公网IP

HTTPS

配置费用

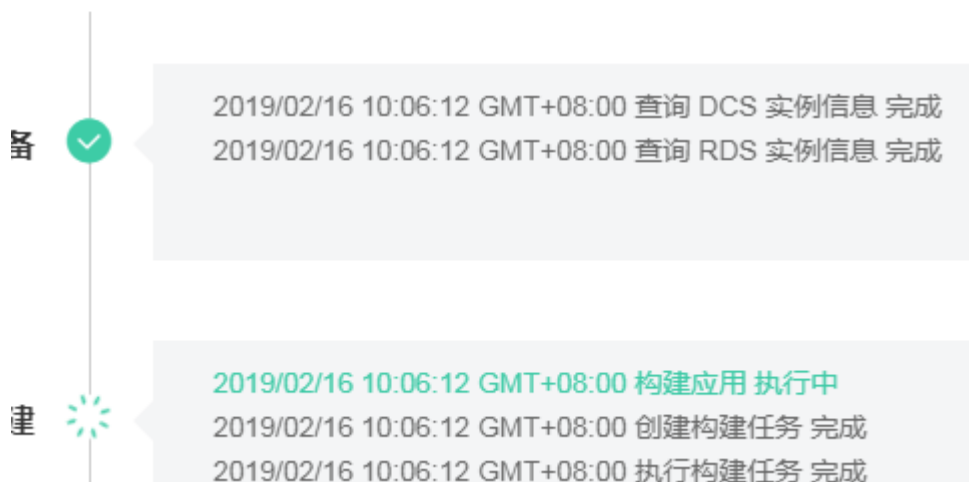
¥ 0.00/小时

- 5、下一步，确认提交，等待应用部署完成

应用webapp正在创建中...

返回应用管理

查看应用详情



6、部署完成后，点击查看应用详情，点击访问地址就可以访问刚才部署的服务

应用管理 > webapp

概览	实例列表	访问方式	更新升级	伸缩	事件	运行日志	阈值告警	运维配置	调度策略	基础设施
应用名称	webapp								类型	
状态	运行中								部署系统	
实例个数（正常/全部）	1/1								所在集群	
外部访问地址	86e220a7-2c10-4969-b724-cac5c6e6f648.hwstaff-100387732.cn-north-1.huaweicse.com								命名空间	
创建时间	2019/02/16 10:06:12 GMT+08:00								标签	

在外部访问地址后增加接口url: /cse/helloworld?name=friends，效果类似如下：

浏览器地址栏显示：  
不安全 | 86e220a7-2c10-4969-b724-cac5c6e6f648.hwstaff-100387732.cn-north-1.huaweicse.com/cse/helloworld?name=friends  
应用 | CICD | OLD | MicroService | 备忘 | K8S | golang | 原型 | oauth | FUXI | 环境 | ServiceStage · Co...  
"hi, friends"

## 5 创建 ServiceComb 微服务应用 helloworldgray

- 1、在ServiceStage菜单，点击 总览 > 快捷操作 > 创建ServiceComb应用。
- 2、运行环境选择Java8，修改应用显示名称为helloworldgray（自动生成的应用名称不

要修改），应用来源选择“源码仓库”，代码源来源选择DevCloud，输入密码进行授权（如已授权则这一步可省略），仓库名称选择准备工作导入的helloworld项目，分支选择gray，版本填写2.0.0

3、剩下操作与上面步骤3后的完全一样（注意端口号要改为8081），等待部署完成。

## 6 灰度发布

当前 helloworld 存在两个版本, master 分支的版本 1.0.0 返回的是 hi, xxx; gray 分支的 2.0.0 返回的是 hello, xxx。下面我们添加灰度策略

- 1、选择左边菜单 基础设施 -> 微服务引擎，进入Cloud Service Engine的控制台。在服务目录，点击进入helloworld微服务

微服务名称	所属应用
webapp	springmvc
helloworld	springmvc

- 2、 左边菜单 灰度发布。添加发布规则：对于参数name=world的导流到2.0.0版本

### 创建新规则

★ 发布规则

权重 自定义

★ 规则名称

作用域

★ 版本 ☐ 1.0.0 ☒ 2.0.0

☐ 是否添加自定义版本

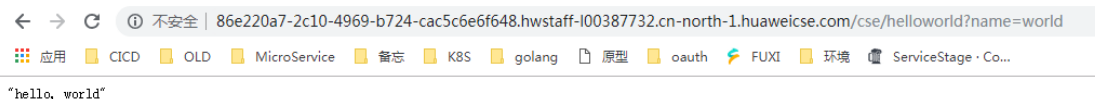
规则配置

★ 参数名

★ 规则

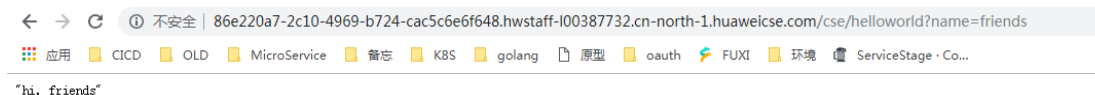
• 参数符合【name=world】条件的请求将会分配到【2.0.0】微服务版本中

- 3、 再访问接口`http://{服务域名}/cse/helloworld?name=friends`,可以看到返回一直都是“hi, friends”; 如果把name改为world, 结果返回一直是“hello, world”



## 7 打卡截图

- 1、 访问接口url, `http://{服务域名}/cse/helloworld?name=friends`, 返回 “hi, friends”



- 2、 选择左边菜单 基础设施 -> 微服务引擎, 进入Cloud Service Engine的控制台。查看服务治理





3、访问接口`http://{服务域名}/cse/helloworld?name=world`,可以看到返回一直都是“hello, world”

