应用平台

开发指南

文档版本	06
发布日期	2025-02-17

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目 录

1 使用 Spring Cloud 框架实现应用开发

1.1 Spring Cloud 概述

Spring Cloud为开发人员提供了一些工具来快速构建分布式系统中的一些常见模式（例如配置管理，服务发现，断路器，智能路由，微代理，控制总线，短期微服务和契约测试）。分布式系统的协调导致了样板模式，使用Spring Cloud，开发人员可以快速构建实现这些模式的服务和应用程序。它们可以在任何分布式环境中工作，包括开发人员自己的笔记本电脑、裸机数据中心和Cloud Foundry等托管平台。

约束与注意事项

AppStage提供的SDK是基于Java1.8版本开发的，如果Spring Cloud项目使用Java11及以上版本，则不支持使用AppStage提供的SDK进行应用开发。

1.2 准备工作

开发技能要求

- 熟悉Java语言，能够编写Java语言代码。
- 掌握IaC开发技术，熟悉YAML语言。
- 了解Spring Cloud框架。

环境准备

- 已下载并安装Maven，根据以下步骤配置Maven。
  a. 在<localRepository>标签内添加自己的本地仓库位置路径，这个本地仓库位置是自己创建的。

     D:\apache-maven-3.8.6-bin\repository

     ```
     <localRepository>D:\apache-maven-3.8.6-bin\repository</localRepository>
     ```

  b. 修改Maven默认的JDK版本。

     在<profiles>标签下添加一个<profile>标签，修改Maven默认的JDK版本。

     ```
     <profile>
       <id>JDK-1.8</id>
       <activation>
     ```



```
        <activeByDefault>true</activeByDefault>
        <jdk>1.8</jdk>
    </activation>
    <properties>
        <maven.compiler.source>1.8</maven.compiler.source>
        <maven.compiler.target>1.8</maven.compiler.target>
        <maven.compiler.compilerVersion>1.8</maven.compiler.compilerVersion>
    </properties>
</profile>
```

- 安装并配置IntelliJ IDEA开发工具。
  a. 在IntelliJ IDEA中选择File > Settings > Bulid,Execution,Deployment > Bulid Tools > Maven。
  b. 在User settings file中配置setting.xml。
  c. 在Local repository中配置自定义的Maven仓库地址。
- JAVA开发环境的配置。

  AppStage提供的SDK是基于Java1.8版本开发的，如果Spring Cloud项目使用Java11及以上版本，则不支持使用AppStage提供的SDK进行应用开发。以下步骤以win7环境配置JDK8 64位为例，如果已经下载JDK并配置好环境请跳过本步骤。

  a. 下载**JDK文件**。
  b. 下载完成后按照提示安装，位置自选，比如安装到本地C:\Program Files\Java\jdk1.8.0_131。
  c. 配置Java环境变量：右键“计算机>属性>高级系统设置>环境变量”，进行如下操作。
     i. 新建系统变量JAVA_HOME，变量值为实际JDK安装位置。
     ii. 在Path中添加%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin （注意用英文分号分隔）。
     iii. 新建系统变量CLASSPATH，变量值为%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar。
     iv. 打开命令行窗口，输入“java -version”，显示如**图1**表示配置成功。

     图 1-1 配置成功示例

     ```
     C:\>java -version
     java version "1.8.0_131"
     Java(TM) SE Runtime Environment (build 1.8.0_131-b11)
     Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed mode)
     ```

SDK 下载与安装

1. 获取SDK并进行完整性校验。
   - SDK：**nuwa-open-sdk-1.1.0-20240204093135.zip**
   - 完整性校验：**nuwa-open-sdk-1.1.0-20240204093135.zip.sha256**
2. 打开本地Spring Cloud项目。
3. 手动导入jar包。
   a. 在项目目录下新建一个lib目录，存放jar包。
   b. 将本地的jar包复制粘贴至lib目录下。
   c. 将jar包导入到项目中。
      i. 选择“File > Project Structure > Project Settings > Module”。

ii. 单击“+”，选择“JARs or Directories...”。
iii. 选中jar包，单击“apply”。导包完成。

下载 Demo

下载Spring Cloud项目的Demo，参考本文档对Demo源码进行理解，您可以基于Demo进行二次开发，节省开发成本。

Demo下载链接：**huaweicloud-appstage-demo-java-codeHub**。

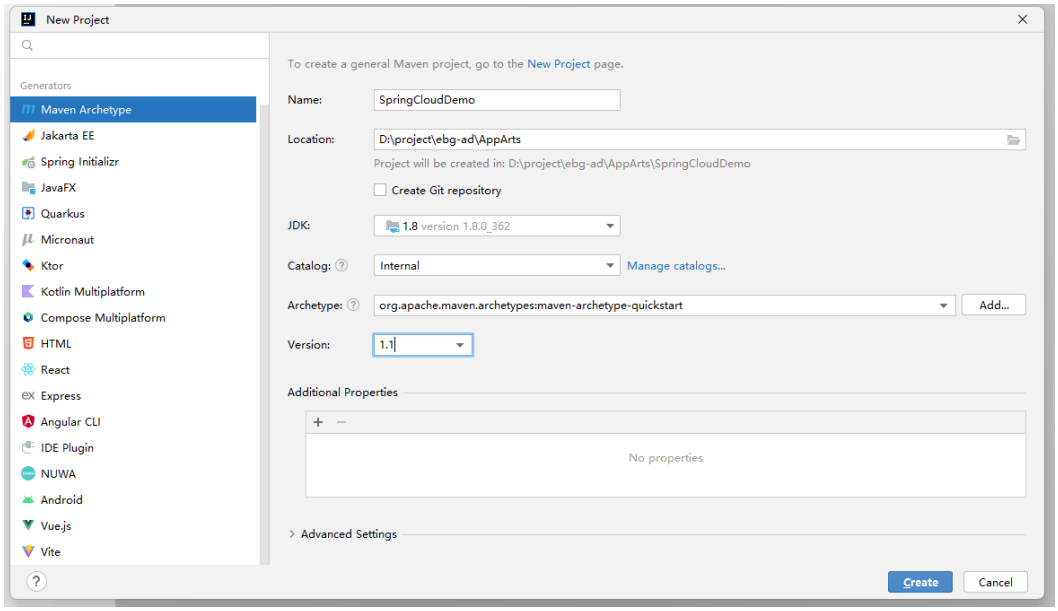
1.3 开发指导

1.3.1 构建 Spring Cloud 工程

创建父工程

步骤1 创建Maven工程。

图 1-2 创建 Maven 工程



步骤2 父pom添加依赖。

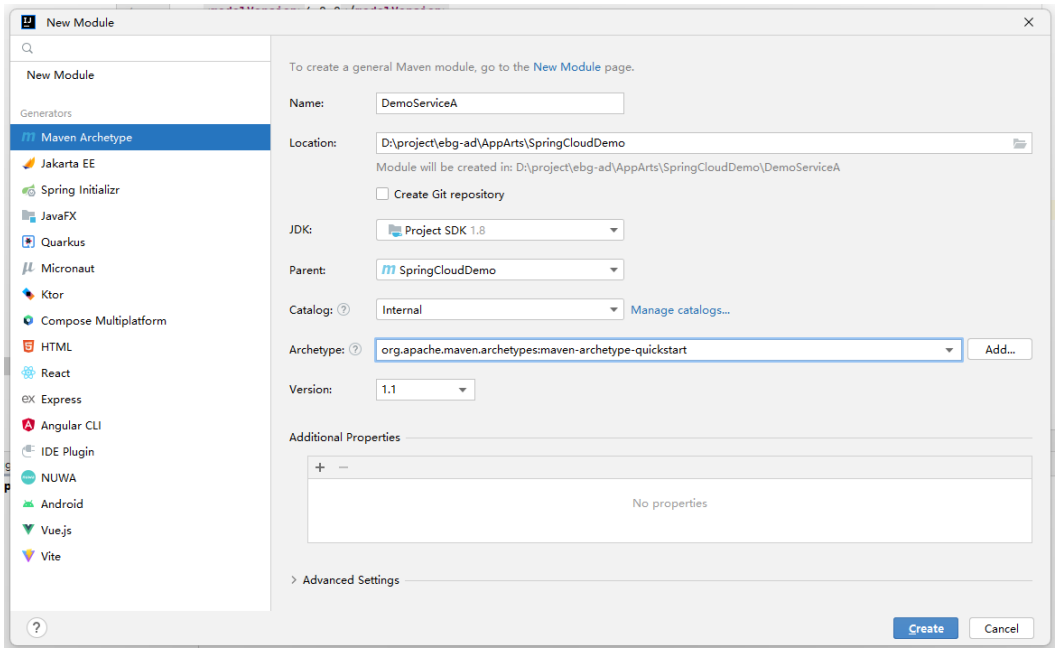
```
<dependencyManagement>
   <dependencies>
      <dependency>
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-dependencies</artifactId>
         <version>2.2.8.RELEASE</version>
         <type>pom</type>
         <scope>import</scope>
      </dependency>
   </dependencies>
</dependencyManagement>
```

----结束

创建子工程 ServiceA

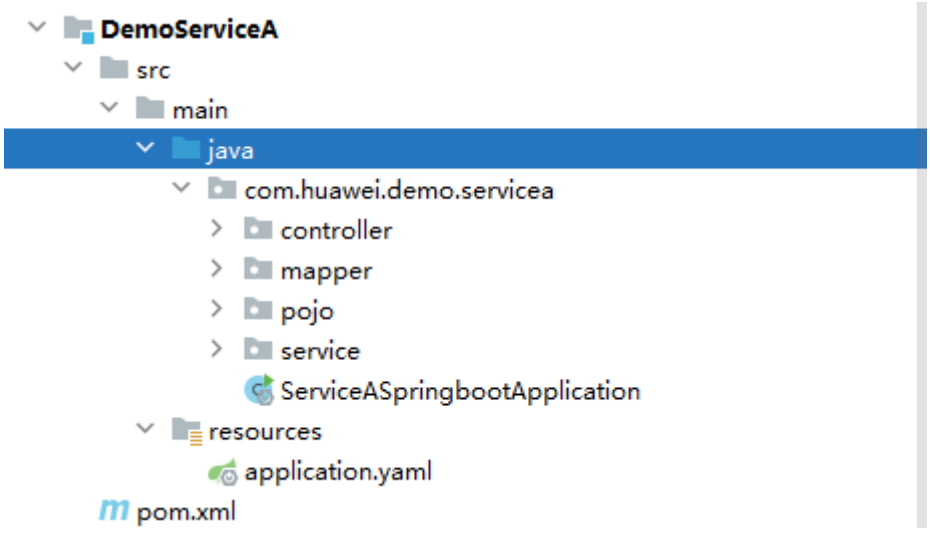
步骤1 创建Maven工程。

图 1-3 创建 Maven 工程



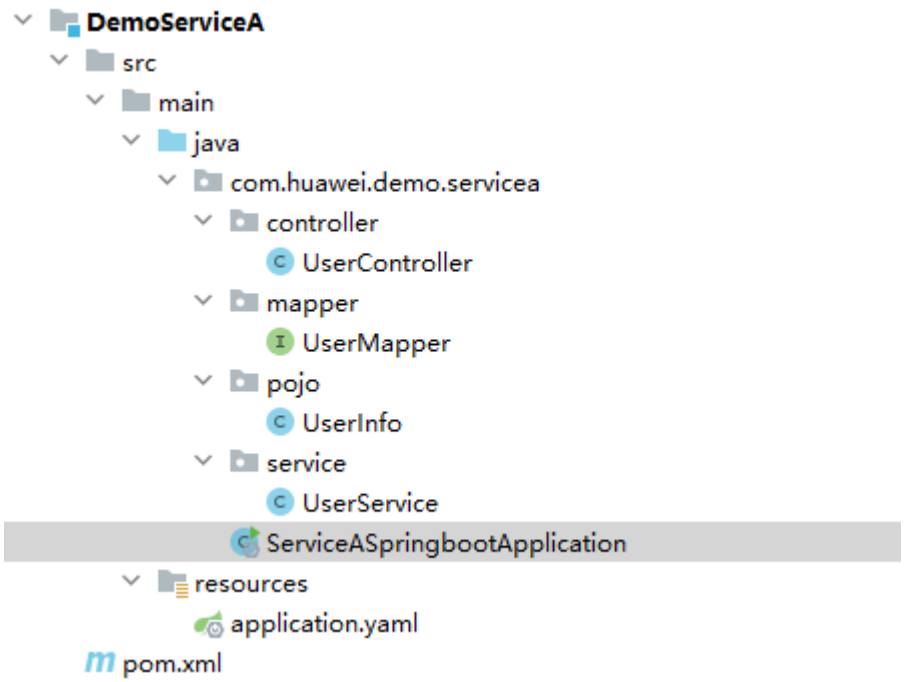
步骤2 新建src目录。

图 1-4 新建 src 目录



步骤3 编写业务代码。

图 1-5 业务代码文件



1. 编写启动类

```
package com.huawei.demo.servicea;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.netflix.eureka.EnableEurekaClient;

/**
 * 启动类
 *
 * @author XXX
 * @since 2023-12-05
 */
@SpringBootApplication
@EnableEurekaClient
public class ServiceASpringbootApplication {
    public static void main(String[] args) {
        SpringApplication.run(ServiceASpringbootApplication.class, args);
    }
}
```

2. 编写Controller类

```
package com.huawei.demo.servicea.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.huawei.demo.servicea.pojo.UserInfo;
import com.huawei.demo.servicea.service.UserService;

/**
 * 用户对外接口
 *
 * @author XXX
 * @since 2023-12-06
 */
```

```
@RestController
@RequestMapping("/user")
public class UserController {
    @Autowired
    private UserService userService;

    @GetMapping("/{userId}")
    public UserInfo getUserByName(@PathVariable String userId) {
        return userService.getUserById(userId);
    }
}
```

3. 编写Mapper类

```
package com.huawei.demo.servicea.mapper;

import org.apache.ibatis.annotations.Mapper;
import org.apache.ibatis.annotations.Select;

import com.huawei.demo.servicea.pojo.UserInfo;

/**
 * 用户查询
 *
 * @author XXX
 * @since 2023-12-06
 */
@Mapper
public interface UserMapper {
    @Select("select * from demo_user_info where user_id = #{userId}")
    UserInfo getUserById(String userId);
}
```

4. 编写Pojo类

```
package com.huawei.demo.servicea.pojo;

import lombok.Data;

/**
 * user信息
 *
 * @author XXX
 * @since 2023-12-06
 */
@Data
public class UserInfo {
    private String userId;

    private String userName;

    private String phone;

    private String address;
}
```

5. 编写Service类

```
package com.huawei.demo.servicea.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.huawei.demo.servicea.mapper.UserMapper;
import com.huawei.demo.servicea.pojo.UserInfo;

/**
 * userService
 *
 * @author XXX
 * @since 2023-12-06
 */
@Service
public class UserService {
```

```
    @Autowired
    private UserMapper userMapper;

    public UserInfo getUserById(String userId) {
        return userMapper.getUserById(userId);
    }
}
```

6. 配置微服务

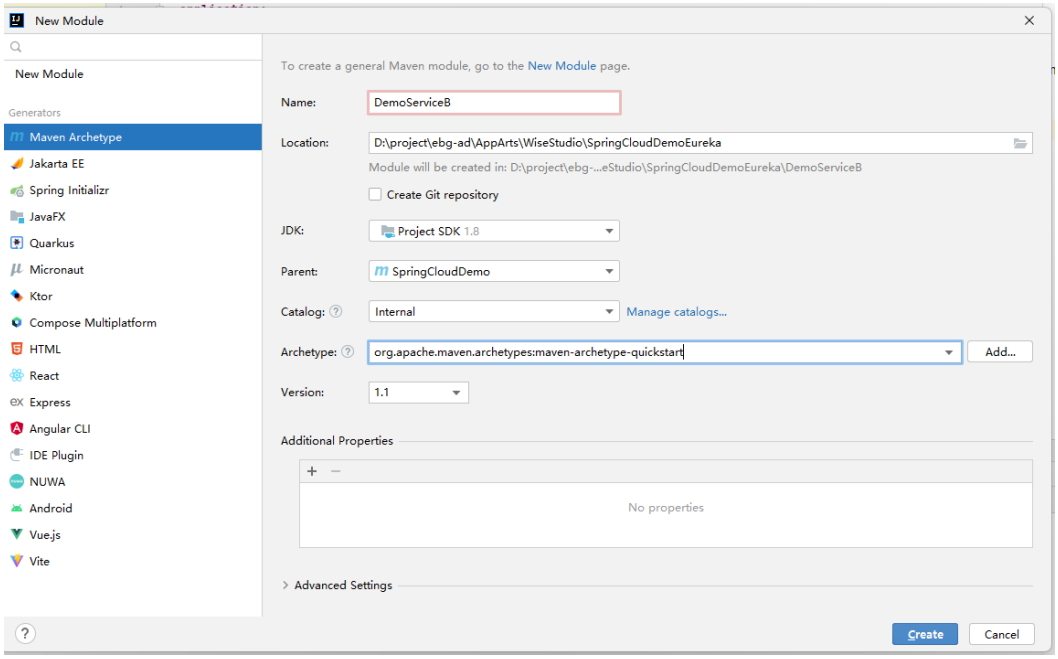
```
server:
  port: 8081
spring:
  application:
    name: demoServiceA
  datasource:
    url: jdbc:mysql://127.0.0.1:3306/spring_cloud_demo?
useUnicode=true&characterEncoding=UTF-8&serverTimezone=Asia/Shanghai
    username: root
    password: ***
    driver-class-name: com.mysql.jdbc.Driver

mybatis:
  configuration:
    map-underscore-to-camel-case: true
  type-aliases-package: com.huawei.dmo.servicea
```

----结束

创建子工程 ServiceB

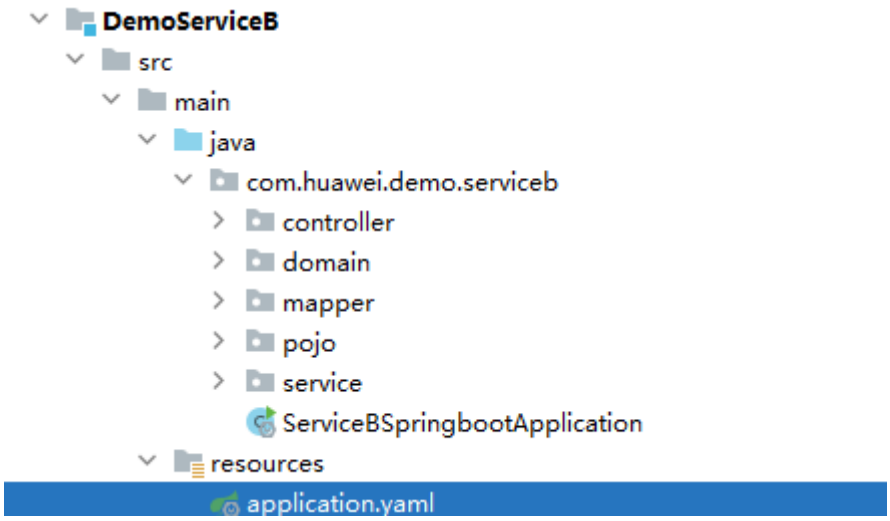
步骤1 创建Maven工程。

图 1-6 创建 Maven 工程



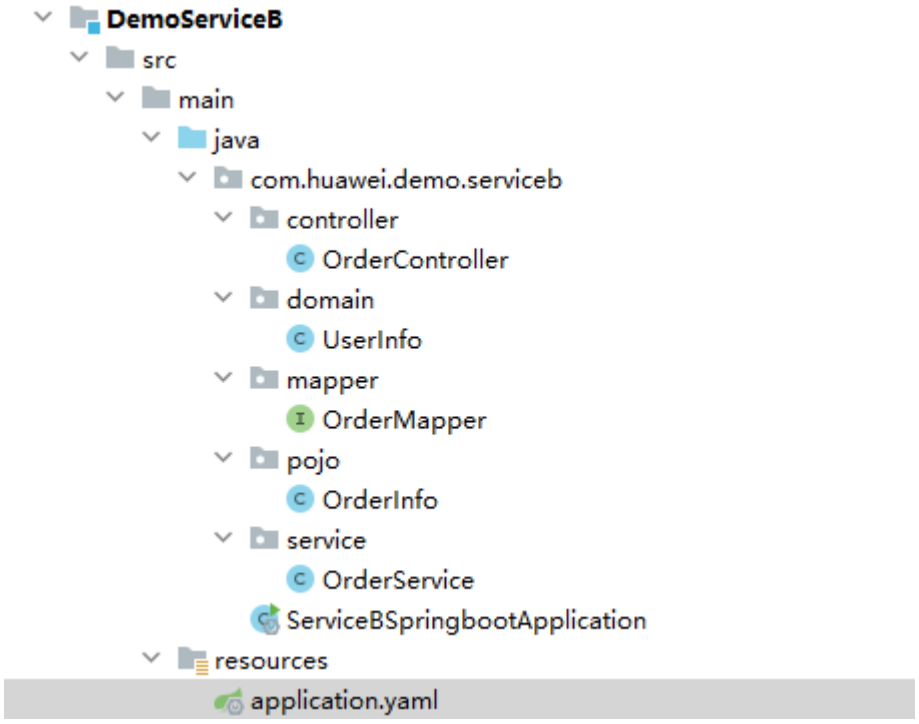
步骤2 新建src目录。

图 1-7 新建 src 目录



步骤3 编写业务代码。

图 1-8 业务代码文件



1. 编写启动类

```
package com.huawei.demo.serviceb;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.client.loadbalancer.LoadBalanced;
import org.springframework.cloud.netflix.eureka.EnableEurekaClient;
import org.springframework.context.annotation.Bean;
import org.springframework.web.client.RestTemplate;

/**
 * 功能描述
```



```
 *
 * @author XXX
 * @since 2023-12-05
 */
@SpringBootApplication
@EnableEurekaClient
public class ServiceBSpringbootApplication {
    public static void main(String[] args) {
        SpringApplication.run(ServiceBSpringbootApplication.class, args);
    }

    @Bean
    @LoadBalanced
    public RestTemplate restTemplate() {
        return new RestTemplate();
    }
}
```

2. 编写Controller类

```
package com.huawei.demo.serviceb.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.huawei.demo.serviceb.pojo.OrderInfo;
import com.huawei.demo.serviceb.service.OrderService;

/**
 * 用户对外接口
 *
 * @author XXX
 * @since 2023-12-06
 */
@RestController
@RequestMapping("/order")
public class OrderController {
    @Autowired
    private OrderService orderService;

    @GetMapping("/{orderId}")
    public OrderInfo findOrder(@PathVariable String orderId) {
        return orderService.findOrder(orderId);
    }
}
```

3. 编写Domain类

```
package com.huawei.demo.serviceb.domain;

import lombok.Data;

/**
 * user信息
 *
 * @author XXX
 * @since 2023-12-06
 */
@Data
public class UserInfo {
    private String userId;

    private String userName;

    private String phone;

    private String address;
}
```

4. 编写Mapper类

```
package com.huawei.demo.serviceb.mapper;

import org.apache.ibatis.annotations.Mapper;
import org.apache.ibatis.annotations.Select;

import com.huawei.demo.serviceb.pojo.OrderInfo;

/**
 * 用户查询
 *
 * @author XXX
 * @since 2023-12-06
 */
@Mapper
public interface OrderMapper {
    @Select("select * from demo_order_info where order_id = #{orderId}")
    OrderInfo getOrderById(String orderId);
}
```

5. 编写Pojo类

```
package com.huawei.demo.serviceb.pojo;

import com.huawei.demo.serviceb.domain.UserInfo;

import lombok.Data;

/**
 * order信息
 *
 * @author XXX
 * @since 2023-12-06
 */
@Data
public class OrderInfo {
    private String orderId;

    private String orderName;

    private String price;

    private String userId;

    private UserInfo userInfo;
}
```

6. 编写Service类

```
package com.huawei.demo.serviceb.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.web.client.RestTemplate;

import com.huawei.demo.serviceb.domain.UserInfo;
import com.huawei.demo.serviceb.mapper.OrderMapper;
import com.huawei.demo.serviceb.pojo.OrderInfo;

/**
 * 功能描述
 *
 * @author XXX
 * @since 2023-12-06
 */
@Service
public class OrderService {

    @Autowired
    private RestTemplate restTemplate;

    @Autowired
```



```
    private OrderMapper orderMapper;

    public OrderInfo findOrder(String orderId) {
        OrderInfo orderInfo=orderMapper.getOrderById(orderId);
        String url = "http://demoServiceA/user/" + orderInfo.getUserId();
        UserInfo userInfo = restTemplate.getForObject(url, UserInfo.class);
        orderInfo.setUserInfo(userInfo);
        return orderInfo;
    }
}
```

7. 配置微服务

```
server:
  port: 8082
spring:
  application:
    name: demoServiceB
  datasource:
    url: jdbc:mysql://127.0.0.1:3306/spring_cloud_demo?
useUnicode=true&characterEncoding=UTF-8&serverTimezone=Asia/Shanghai
    username: root
    password: ***
    driver-class-name: com.mysql.jdbc.Driver

eureka:
  client:
    service-url:
      defaultZone: http://localhost:11011/eureka/

mybatis:
  configuration:
    map-underscore-to-camel-case: true
  type-aliases-package: com.huawei.dmo.servicea
```

----结束

创建 Eureka 注册中心

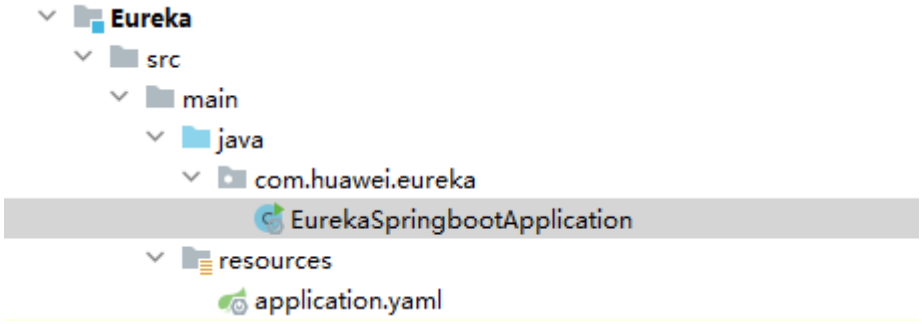
步骤1 创建Eureka微服务。

步骤2 添加依赖。

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
</dependency>
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-eureka-server</artifactId>
</dependency>

<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

步骤3 新建启动类。

图 1-9 新建启动类

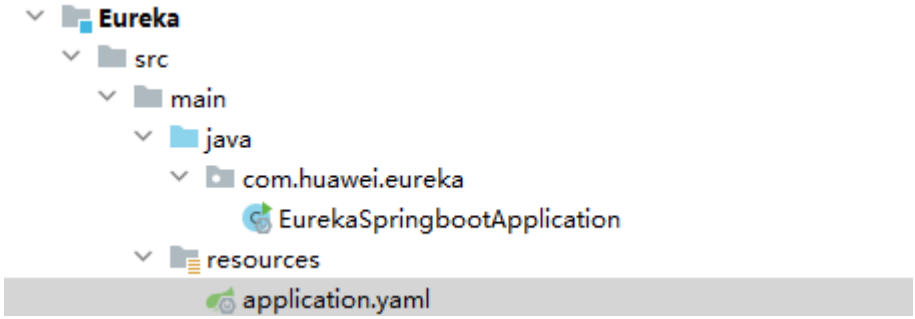


步骤4 启动类中增加注解@EnableEurekaServer，标明注册中心微服务。

```
@SpringBootApplication
@EnableEurekaServer
public class EurekaSpringbootApplication {
    public static void main(String[] args) {
        SpringApplication.run(EurekaSpringbootApplication.class, args);
    }
}
```

步骤5 在Eureka微服务中增加配置文件。

图 1-10 增加配置文件



步骤6 配置注册中心微服务端口等。

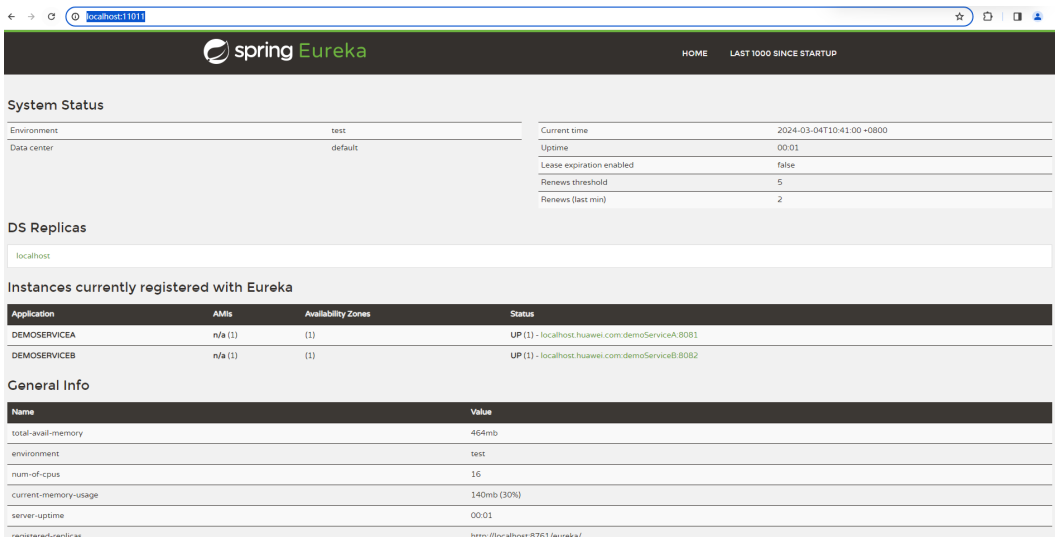
```
server:
  port: 11011
spring:
  application:
    name: eureka

eureka:
  client:
    register-with-eureka: false
    fetch-registry: false
```

步骤7 其他微服务中增加注册中心配置。

```
eureka:
  client:
    service-url:
      defaultZone: http://localhost:11011/eureka/
```

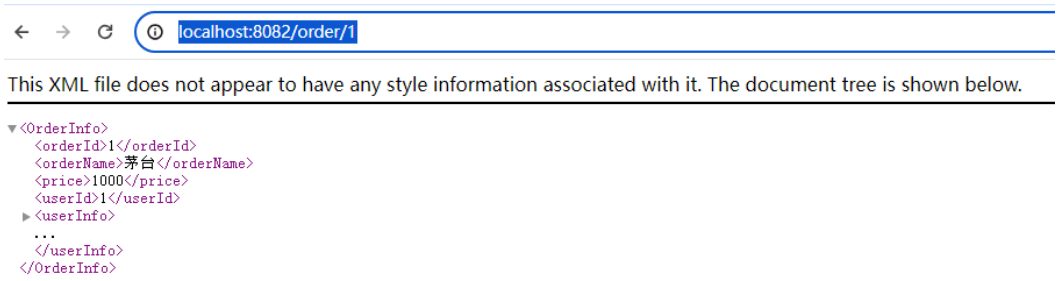
步骤8 依次启动Eureka、DemoServiceA、DemoServiceB，访问http://localhost:11011/，可以看到DemoServiceA、DemoServiceB的信息。

图 1-11 查看微服务注册信息



步骤9 调用接口：http://localhost:8082/order/1。

图 1-12 调用接口



----结束

1.3.2 接入 STS（ACMS）

STS（Security Token Service）提供了微服务注册以及敏感配置项管理的功能，STS是接入Cloud Map的前提条件，Cloud Map依赖STS认证能力。

操作步骤

接入STS的具体操作请参见**使用STS SDK（Spring Cloud框架）**。

1.3.3 敏感配置项托管

由于业务的敏感配置不能明文地存放在版本包、配置中心、IaC代码中，因此业务可以借助STS敏感配置项的功能，存放业务的敏感配置。

操作步骤

步骤1 使用STS的敏感配置项管理功能，需要在ACMS中录入敏感配置项，具体请参见**录入敏感配置**。

步骤2 在IaC脚本中的业务配置项配置文件中指定敏感配置项坐标，敏感配置项坐标在ACMS中生成。

此处以增加一个名为spring.redis.password的敏感配置项为例，这个敏感配置项是访问Redis的密码。

```
spring.redis.password: MicroService/{service}/{microservice}/spring.redis.password/default   # 敏感配置项坐标为：MicroService/服务名/微服务名/敏感配置项名称/敏感配置项标签
```

步骤3 在IaC脚本中的业务配置项属性定义文件中，声明该配置项为敏感配置项。

```
type: object
properties:
  spring.redis.password:
    format: sensitive
```

步骤4 在application.yml配置文件中增加敏感配置项名称的配置。

```
nuwa:
  security:
    config:
      sensitiveWords: spring.redis.password,org.app.protocol-login.oauth.clientSecret,org.app.jwt-key
```

步骤5 启动敏感配置项自动解密。

在启动类中添加@EnableStsEncryptableProperties注解。

----结束

1.3.4 接入 Cloud Map

Spring Cloud通常是使用其自带的Eureka注册中心，使用应用平台可以将Eureka注册中心替换为Cloud Map，Cloud Map除了能够提供服务发现的功能，还可以提供数据库、敏感信息等的纳管功能。

前提条件

Cloud Map依赖STS认证能力，接入Cloud Map前需要先接入STS。

操作步骤

接入Cloud Map的具体操作请参见**使用Cloud Map SDK（Spring Cloud框架）**。

1.3.5 使用 WiseDBA 进行数据库纳管

前提条件

- WiseDBA需要依赖Cloud Map，接入WiseDBA前需要先接入Cloud Map。
- 在AppStage运维中心的WiseDBA中申请数据库并创建Schema，具体请参见创建数据库实例及**在WiseDBA中创建Schema**。

操作步骤

使用WiseDBA进行数据库纳管的具体操作请参见**使用Rainbow SDK（Spring Cloud框架）**。

1.3.6 集成 OrgID 登录功能

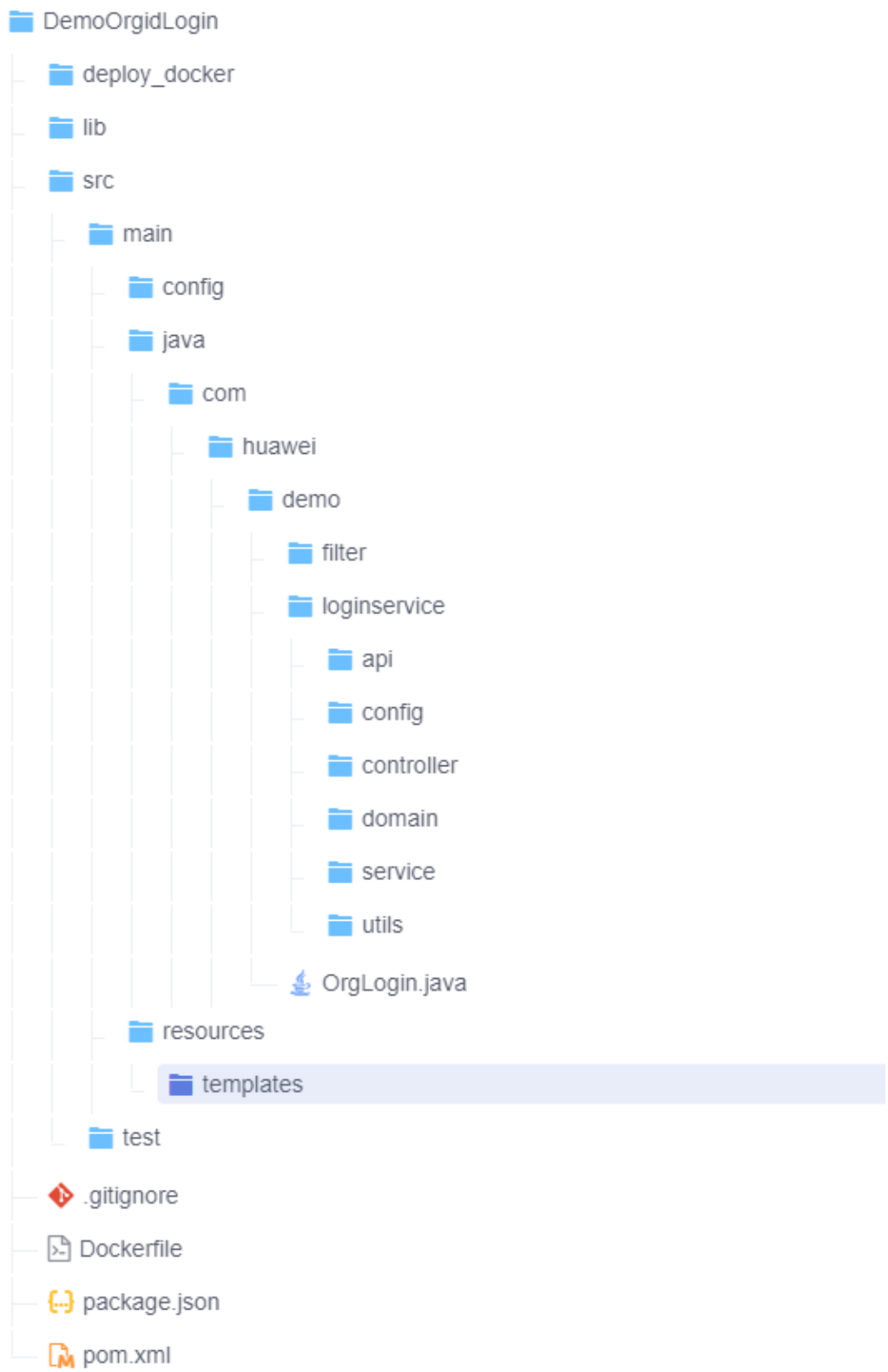
1.3.6.1 概述

支持对接OrgID组织成员账号服务，对接后，通过标准Oauth2.0协议登录到OrgID的应用，从而实现使用OrgID服务对自身应用的组织、部门、成员账号进行管理。

1.3.6.2 了解代码结构

OrgID登录功能Demo的代码结构如**图1-13**所示。

图 1-13 代码结构



- deploy_docker为docker部署配置信息。

```
├─deploy_docker
│  ├─bin
│  │      configparser
```

```
│   │   start.sh                          ----服务在服务器上的启动脚本
│   │
│   ├─configtemplate
│   │   application.properties.tmpl        ----STS域名及微服务STS证书路径
│   │   application.yml.tmpl               ----服务器上环境变量配置信息
│   │   logback-spring.xml.tmpl            ----logback日志配置
│   │   meta.txt
│   │
│   └─resources
│       index.html                         ----demo应用的静态页面（登录后的首页欢迎页）
│       login.html                         ----demo应用的静态页面（登录页，单击按钮跳转登录）
```

- lib目录为工程依赖的jar包。

```
├─lib
│   cloudsoa-security-1.1.14.101.jar
│   gpaas-middleware-common-2.2.6.100.jar
│   nuwa-cloudmap-core-1.0.12.100.jar
│   nuwa-cloudmap-spring-boot-starter-1.0.12.100.jar
│   nuwa-core-3.1.5.100.jar
│   rainbow-core-drds-1.2.18.201.jar
│   rainbow-proxy-1.2.18.102.jar
│   spring-cloud-starter-cloudmap-discovery-1.0.12.100.jar
│   sts-key-sdk-1.1.19.100.jar
│   sts-sdk-base-1.1.19.100.jar
│   sts-spring-boot-1.1.19.100.jar
```

- src目录为项目工程源码目录。

```
└─src
    ├─main
    │  ├─config
    │  │   application.properties                    ----sts域名及微服务sts证书路径（本地启动时配置）
    │  │   application.yml                           ----工程的环境变量配置（本地启动时配置）
    │  │   logback-spring.xml                        ----logback日志配置
    │  │
    │  ├─java
    │  │  └─com
    │  │     └─huawei
    │  │        └─demo
    │  │           │  OrgLogin.java                  ----项目启动类
    │  │           │
    │  │           ├─filter
    │  │           │     LogRecordPostFilter.java    ----接口日志过滤器（打印所有请求的异常信息及请求信息）
    │  │           │
    │  │           └─loginservice
    │  │              ├─api
    │  │              │     UserLoginApi.java        ----接口api定义
    │  │              │
    │  │              ├─config
    │  │              │     LoginConfig.java         ----登录拦截器配置（决定对哪些路径放行）
    │  │              │     RedisConfig.java         ----Redis数据库连接工具参数配置
    │  │              │     RestTemplateConfig.java  ----RestTemplate，http请求参数配置
    │  │              │     UserLoginInterceptor.java ----拦截器拦截逻辑，对未登录用户进行拦截
    │  │              │
    │  │              ├─controller
    │  │              │     UserLoginController.java ----接口实现controller层
    │  │              │
    │  │              ├─domain
    │  │              │  │  Constance.java           ----常量定义
    │  │              │  │
    │  │              │  └─entity
    │  │              │        BaseResponse.java     ----基础响应体定义
    │  │              │        UserInfo.java         ----Oauth协议登录后，认证源（OrgID）返回的用户信息
    │  │              │
```

```
        ├─service
        │      LoginService.java                    ----接口实现service层
        │
        └─utils
               ApplicationContextUtil.java          ----上下文获取工具类
               JsonUtils.java                       ----JsonUtils工具类，json串与对象之间的相互转换
               JwtUtil.java                         ----Jwt工具类，用于生成token、解析token信息
               RedisUtil.java                       ----Redis工具类，对Redis数据库做数据操作
```

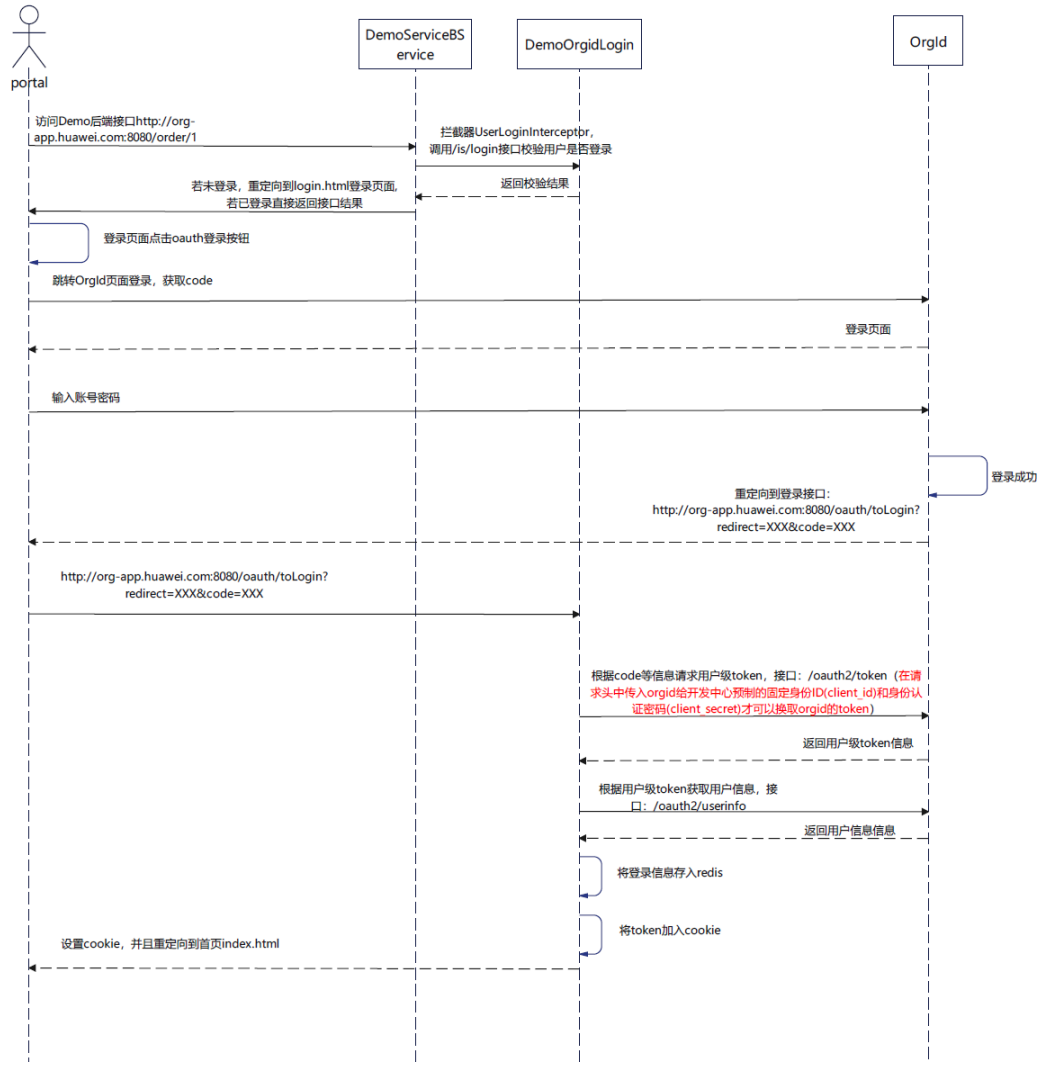
- resources目录，用于存放本地启动时静态资源（欢迎页与登录页html文件），同docker部署时resources目录。

```
    └─resources
        └─templates
               index.html
               login.html
```

1.3.6.3 接口详解

了解 Oauth2.0 协议登录流程（与 OrgID 的交互流程）

图 1-14 Demo 登录流程图



1. 登录获取code：应用A首先需要在OrgID平台上进行注册，并进行相应的配置，比如，首页登录url，退出地址url等，然后用户通过浏览器在OrgID界面单击应用或者直接访问应用服务地址，浏览器向应用A发起登录请求，应用A接收后，会向OrgID发起认证，认证的请求信息为：

```
method: get,
url:    https://orgin-dev.huawei.com/oauth2/authorize,
param:  response_type: code,
        client_id: 应用A的clientID（需要在OrgID页面上获取）
        scope: phone+profile+email（授予应用A的用户信息范围）
        redirect_uri: https://myApp.com/login (在orgID上配置的应用A的首页登录url)
```

如果OrgID监测到用户未登录，则会跳转到OrgID的登录页，进行登录。登录成功后，OrgID会返回一个带有code参数的重定向请求（重定向的地址为应用A在OrgID上配置的首页登录url）。重定向的url示例为：

```
url: https://myApp.com/login?code=lwooqdwgqfCqQkZ-kiZF7zwuiYyYg8MH4gZ0EE1NFk-
L4rnl9v9Nqjw1cW1awpbw5RFhCLYdi-zPa2e-Qru8qBma6KIN7f-
HBrwRLdAm4kNx24B0D0gUXOEs2eezH5UE
```

2. 通过code获取token：通过在第一步中的重定向url路径中获取code。应用A拿着code，向orgID发送一个post请求来获取token，请求信息如下：

```
method:   post,
url:      https://orgid-dev.huawei.com/oauth2/token
formdata: grant_type: authorization_code,
          code: 第一步中获取到的code,
          redirect_url: https://myApp.com/login,
          client_id: 应用A的clientID
          client_secret: 应用A的client_secret
```

获取的响应为：

```
{
   "access_token": "J2Bp_wnA_EJDxCqEdaw2VrE_xol60YeGG1_zKbKzGLBX0l407dkjfe9O-
iOxl7dn87nGekGcxZCwlSzyGH9yp71QRtO36R18qki6foIQYyIN58o1mi8c4-0dFGCwmHII",
   "refresh_token": "pTEUCnloqhCRuJpkueXAeqTNXo2gPRja90Ba9nAe6l1si-d9hK-
njhB3W_3rbDIvH4rIs_59FrJ1Bb6iVkDkKj81NOV7uiNtRbSois8Eweh2QTfBR9Dx9aZ6PcJdJg0-",
   "scope": "phone profile email",
   "token_type": "Bearer",
   "expires_in": 7200
}
```

3. 通过token获取登录的用户信息：得到token后，应用A拿着token，向orgID发送一个get请求来获取用户信息，请求信息如下：

```
method:     get
url:        https://orgid-dev.huawei.com/oauth2/userinfo
headers:
            Authorization: Bearer access_token (其中access_token是第二步请求中的响应体)
```

获取的响应为：

```
{
   "tenant_name": "zzxTest",
   "role": "admin",
   "user_id": "1008600000020011612",
   "user_name": "hid_prsll6r9d8w784c",
   "name": "150******53",
   "mobile": "150******53",
   "tenant": "919008600000119****",
   "employee_code": "00001"
}
```

demo 工程中的接口介绍

Demo工程中的接口如**图1-15**所示。

图 1-15 Demo 工程中的接口

```
    1 implementation
    @RequestMapping(value = "/login")
    String login();

    1 implementation
    @RequestMapping(value = "/is/login")
    BaseResponse<String> isLogin(HttpServletRequest httpServletRequest) throws Exception;

    1 implementation
    @RequestMapping(value = "/index")
    String index();

    1 implementation
    @RequestMapping(value = "/oauth2/toLogin")
    void toOauth2Login(HttpServletResponse httpServletResponse,
        @RequestParam(value = "code", required = true)
        String code) throws IOException;

    1 implementation
    @RequestMapping(value = "/app/logout")
    void logout(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse) throws Exception;

    1 implementation
    @RequestMapping(value = "/callback/logout")
    void callBackLogout(HttpServletRequest request, HttpServletResponse httpServletResponse) throws Exception;
}
```

- /login接口

  本Demo应用引入了thymeleaf，在配置文件中配置thymeleaf基本参数后，该接口返回为登录页的静态资源login.html，即登录页的界面。

  图 1-16 配置 thymeleaf

  ```
  spring:
    application:
      name: DemoOrgidLogin
    thymeleaf:
      prefix: classpath:/dist
      encoding: UTF-8
      suffix: .html
      mode: HTML5
      cache: false
      servlet:
        content-type: text/html
  ```

- /is/login接口

  该接口为查询当前访问的用户是否登录，如果登录，则返回当前登录的用户信息，否则返回消息为空。与OrgID的登录流程在本Demo中进行，当其他微服务需要判断当前用户登录状态时，内部调用此接口来获取当前登录的用户信息。

- /index接口

  DemoOrgidLogin登录成功后的访问地址首页，同/login接口，返回为登录页的静态资源，即index.html。

- /oauth2/toLogin接口

该接口的接口地址为OrgID侧配置的回调地址，即上述Oauth2.0登录流程中的：回调应用，提供授权码code。

当用户输入账号密码登录后，OrgID会携带授权码code回调此接口，该接口需要依次完成Oauth2.0登录流程中的1~3步，最终拿到用户信息后，根据用户信息生成cookie，建立demo应用与用户浏览器的会话。

- /app/logout接口

  该接口为demo应用的退出接口，当用户需要退出应用时，需完成以下两件事：

  - 需要清除自身会话，完成自身应用退出逻辑。
  - 重定向到OrgID的退出页完成OrgID侧的退出逻辑。

- /callback/logout接口

  该接口为OrgID的回调接口，当用户从OrgID侧发起退出登录时，会通知到应用侧。此时会回调该接口，不同于/app/logout退出接口，该接口只需清理demo应用自身会话，完成自身退出登录。

1.3.6.4 开发者使用 demo 应用配置详细说明

```
spring:
  application:
    name: DemoOrgidLogin                                  # 应用名称，用户可自行决定自身应用名
  thymeleaf:                                              # thymeleaf默认配置
    prefix: classpath:/dist
    encoding: UTF-8
    suffix: .html
    mode: HTML5
    cache: false
    servlet:
      content-type: text/html
  redis:                                                  # Redis连接信息，该信息开发者需配置自己的Redis连接信息
    host: {{spring.redis.host}}                           # Redis连接信息
    port: {{spring.redis.port}}                           # Redis端口号
    password: {{spring.redis.password}}                   # Redis连接密码
  web:
    resources:
      add-mappings: false
      static-locations:
        - classpath:/dist/
server:
  port: 8083                                              # 服务端口号，开发者可根据需求配置自身服务端口号
org:
  url: {{org.url}}                                        # orgid访问域名，为OrgID固定域名不可更改，用于访问
OrgID接口
  web-url: {{org.web-url}}                                # orgid访问域名，为OrgID固定域名不可更改，用于访
问OrgID接口
  app:
    cookie-domain: {{org.app.cookie-domain}}              # 为demo应用与用户建立的会话的cookie域，
需配置为开发者自身应用域名或ip
    jwt-key: {{org.app.jwt-key}}                          # jwt加密密钥，用于创建和解析jwt串。该值开发者可
自行配置，妥善保管长度需为4的倍数
    ent-point-url: {{org.app.ent-point-url}}              # demoApp自身应用的域名或ip，开发者自行配置
    protocol-login:
      oauth:
        clientId: {{org.app.protocol-login.oauth.clientId}}         # Oauth协议在OrgID侧生成的应用凭证，开发者
需自行配置
        clientSecret: {{org.app.protocol-login.oauth.clientSecret}} # Oauth协议在OrgID侧生成的应用凭证密钥，
开发者需自行配置
demo:
  login:
    url: {{demo.login.url}}                               # demo应用自身登录url，用于拦截未登录后，跳转重定
向的登录地址，开发者需自行配置
nuwa:                                                     # Cloud Map配置信息
  security:
    config:
```

```
    sensitiveWords: spring.redis.password,org.app.protocol-login.oauth.clientSecret,org.app.jwt-key
 cloudmap:
   read: cloudmap
   clusterName: {{nuwa.cloudmap.clusterName}}
   provider:
     cluster: {{nuwa.cloudmap.provider.cluster}}
   serverAddr: {{nuwa.cloudmap.serverAddr}}
   version: {{nuwa.cloudmap.version}}
   namespaceName: {{nuwa.cloudmap.namespaceName}}
logging:
 config: /opt/huawei/app/service/config/logback-spring.xml          # logback日志配置路径
```

1.3.6.5 应用对接的整体流程

操作步骤

步骤1 申请注册Huawei ID账号（已有华为ID账号跳过此步）。

注册链接：**华为云**，打开后单击“注册”，填写完整信息。

图 1-17 注册 HuaweiID 账号

步骤2 **访问OrgID服务**，创建OrgID组织。

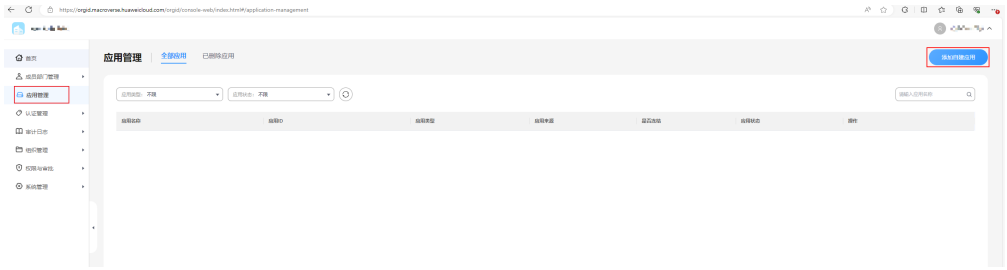
1. 单击“创建组织”，填写组织信息。
2. 单击“进入控制台”，进入控制台。

步骤3 在OrgID管理中心创建自身对接Demo应用信息。

进入控制台单击刚创建好的组织即可进入业务侧管理中心，或直接访问**管理中心**。

1. 选择“应用管理”，单击右上角的“添加自建应用”。

图 1-18 添加自建应用



2. 填写应用信息，上传应用图标，选择Oauth协议。

图 1-19 添加自建应用

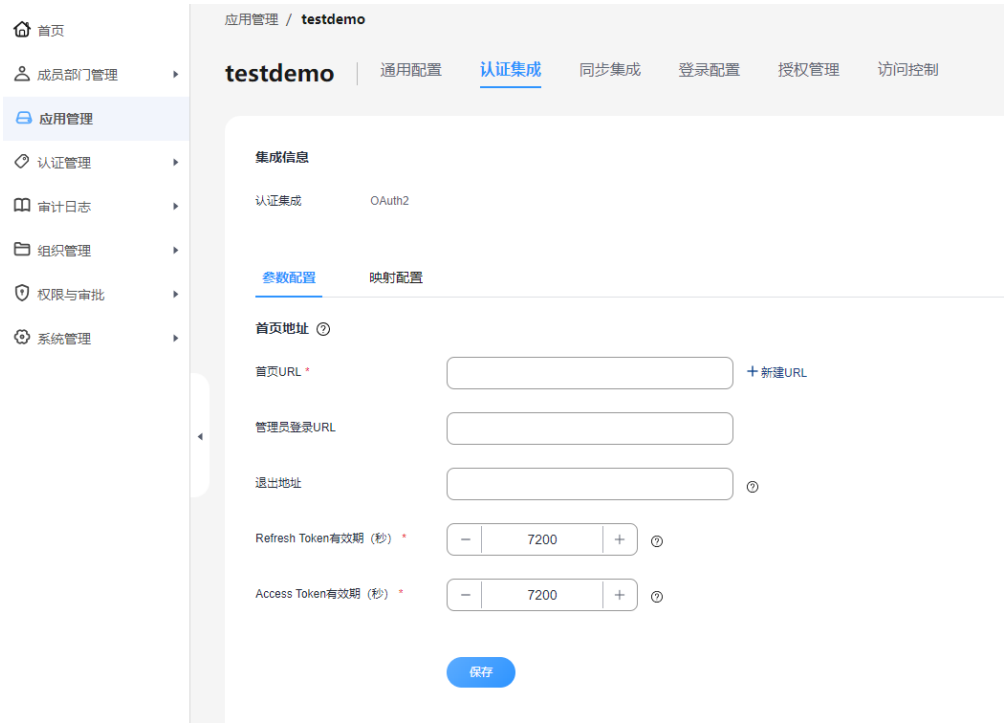
3. 配置Demo应用信息。

表 1-1 应用配置参数说明

参数	说明
首页URL	Oauth登录交互流程的回调地址。
管理员登录URL	如果Demo应用区分管理员角色，且登录后为管理后台，可自行配置。

参数	说明
退出地址	OrgID退出登录时，通知应用的回调接口。
Token有效期	OrgID侧会话最长时间，默认两小时。

图 1-20 应用参数配置



4. 授权登录用户。
   a. 在应用管理页面选择“授权管理”，单击“授权设置”。

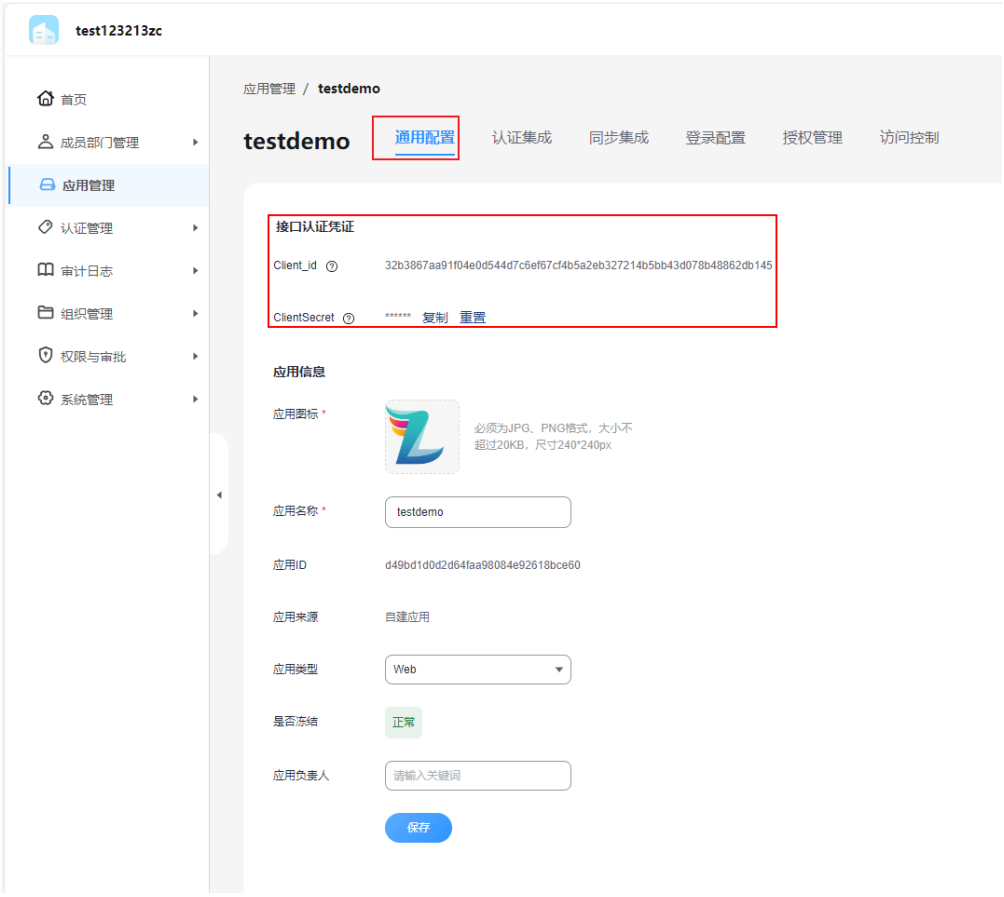
图 1-21 授权设置

   b. 自行选择授权范围。

图 1-22 选择授权范围

5. 查看创建好的Demo应用配置密钥。
   在应用管理页面，选择“通用配置”，复制接口认证凭证。
   接口认证凭证为OrgID侧生成的应用密钥，请开发者妥善保管。

图 1-23 查看应用密钥



6. 修改Demo工程的应用配置信息。

   将application.yml.tmpl中的clientSecret配置为创建好的应用凭证信息，同时将login.html中的按钮登录链接改为已创建好的应用的登录链接，链接地址如图1-24所示。

图 1-24 查看应用的登录链接

7. 修改完成后，启动工程，访问首页登录地址。

通过“域名+端口号+/login”访问首页地址，单击“登录”跳转OrgID登录页，输入账号后，可成功跳转登录后的首页地址。

----结束

1.4 实践案例

本实践以Spring Cloud Demo项目为例，带您体验使用AppStage的开发中心、运维中心进行工程创建、代码开发、打包发布，部署上线的全过程。具体请参考**基于Spring Cloud框架进行应用上云**。

2 应用平台 IaC 部署代码开发

2.1 IaC 概述

IaC 简介

基础设施即代码（Infrastructure as Code，简称IaC）是一种以YAML作为输入，经由云原生环境管理服务、IaC执行引擎、Operator平台解析和执行，实现环境自动部署以及管理动态基础设施的方法。它强调一致，可重复的供给和变更系统及其配置。当代码发生变更后，可以进行自动化测试，测试完成后可自动化的应用变更到运行系统中。使用基础设施即代码的方法，可以使用敏捷工程的优秀实践（如测试驱动开发，持续集成，持续发布）来更加快速安全的变更基础设施。

IaC3.0 支持的部署模式

IaC3.0支持如下两种场景的部署模式。

- IaC Spec包

  同一个服务下所有微服务的IaC代码在一个仓中管理，打包生成IaC Spec的包，可以实现服务下所有的微服务在同一个服务环境下一键部署。

- IaC Patch包

  微服务的IaC代码单独管理，通过IaC Spec包创建了服务环境之后，可以通过微服务级别的IaC Patch包进行微服务的独立部署。

IaC 代码结构介绍

IaC代码支持单文件描述结构、多文件描述结构以及带global的多文件描述结构，具体介绍请参见**IaC代码结构介绍**。

- 单文件描述结构：在IaC主体描述文件meta.yaml中进行变更流程编排同时定义所有资源。
- 多文件描述结构：通过引入resources.yaml和文件引用语法，将单文件结构改造为多文件描述目录结构，避免诸多资源的描述都集中于meta.yaml，而造成文件内容过长难以管理。
- 带global的多文件描述结构：为解决不同规格目录间配置复用问题，IaC3.0支持带global的多文件描述结构。

带global的多文件描述结构为IaC3.0的典型目录结构，IaC Spec目录结构请参见**IaC Spec包典型目录结构**。

IaC 代码开发介绍

在一次完整的业务变更中，往往会涵盖多种类型、多个模块的变更，如集群扩容、申请ELB、创建数据库、软件升级等等。在IaC的语境下，每一个变更本质上都是IaC资源的变更。在一次完整的业务变更中，部分资源的变更依赖于其他资源的变更，如为一个微服务创建NUWA实例之前往往需要先创建该微服务的数据库。

通过IaC代码对各资源在具体变更过程中的依赖关系、先后顺序进行代码化描述。本质上就是描述各模块、各资源之间的依赖关系。在变更过程中，IaC将根据由依赖关系生成的有向无环图顺序执行各资源的变更过程。

IaC代码开发主要围绕声明资源和变更流程编排两个方面展开。

- **在IaC代码中声明资源**
  - 定义component：定义component是IaC将一个环境的资源组织起来的方式，将同一类资源组织起来成为一个component。
  - 定义资源：一个component下可以定义多个资源，所有的资源描述都存放于resources.yaml中。
- **在IaC代码中定义流水线**

  component间的编排在spec包中的meta.yaml文件中描述，用户可以根据自己的需求定义整个环境在变更时的执行过程。

2.2 准备工作

开发技能要求

熟悉YAML语法。

下载 Demo

下载Spring Cloud项目的Demo，参考本文档对Demo源码进行理解，您可以基于Demo进行二次开发，节省开发成本。

Demo下载链接：**huaweicloud-appstage-demo-java-codeHub**。

2.3 IaC 代码结构介绍

单文件描述结构

单文件描述结构样例如下：

```
├── package.json                # 包描述文件（必须）
└── specs                       # 规格总目录（必须）
    ├── cn_dev_default          # cn_dev_default规格目录，可用于描述一个开发用途的服务环境所使用的基础设施
    └── cn_product_default      # cn_product_default规格目录，可用于描述一个生产用途的服务环境所使用的基础设施
        └── meta.yaml           # IaC主体描述文件
```

IaC主体描述文件meta.yaml

```
type: WiseCloud::Environment              # 描述环境类型，当前仅支持 WiseCloud::Environment
components:                               # 定义服务环境所包含的组件列表，每个组件包含一个资源列表
- name: environment                       # 组件名称
  resources:                             # 资源列表
  - name: fgc_cloudmap                   # 资源名称，同类型的资源的名称在整个服务环境中唯一
    type: WiseCloud::Endpoint::CloudMap  # 资源类型
    properties:                          # 资源属性：具体包含哪些属性，由资源类型决定
      ...
- name: FGCActionInvokerService
  resources:
    ...
- name: FGCAbilityCenterService
  resources:
    ...
applyPipeline: default                   # 默认选用名为default的组件编排流水线
pipelines:                               # 定义可用的组件编排流水线
- name: default                          # 流水线的名称，作为流水线被引用的唯一标识
  action: Serial                         # 串行编排
  tasks:                                 # 串行编排的任务列表
  - name: apply-environment              # 任务名称
    action: Apply                        # 应用变更
    component:                           # 变更组件的相关约束
      name: environment                  # 变更的组件名称
  - name: parallel-others
    action: Parallel                     # 并行编排
    tasks:                               # 并行编排的任务列表
    - name: apply-FGCActionInvokerService  # 任务名称
      action: Apply                      # 应用变更
      component:                         # 变更组件的相关约束
        name: FGCActionInvokerService    # 变更的组件名称
    - name: apply-FGCAbilityCenterService  # 任务名称
      action: Apply                      # 应用变更
      component:                         # 变更组件的相关约束
        name: FGCAbilityCenterService    # 变更的组件名称
```

以上示例清晰地展示了IaC3.0的资源组织结构：

- Nuwa，CloudMap等资源，依照业务需要，可划分到不同的组件中。
- 一个组件可对应于一个微服务，或是服务内共享的中间件集合。
- 全体组件的集合，则汇总描述了整个服务环境的期望部署状态。
- 组件编排流水线，则是以组件为最小粒度来描述服务环境是如何做部署状态的迁移的。其可以处理组件间的升级依赖关系，以及通过多阶段方式提供灰度升级能力。

多文件描述结构

为了避免诸多资源的描述都集中于meta.yaml，而造成文件内容过长难以管理。通过引入resources.yaml和文件引用语法，可以将单文件结构改造为多文件描述目录结构，样例如下：

```
├── package.json                  # 包描述文件（必须）
└── specs                         # 规格总目录（必须）
    ├── cn_dev_default            # cn_dev_default规格目录，可用于描述一个开发用途的服务环境所使用的基础设施
    └── cn_product_default        # cn_product_default规格目录，可用于描述一个生产用途的服务环境所使用的基础设施
        └── meta.yaml
        └── VirtualAppManangerService
            ├── config
            │   └── business_config.yaml
            │   └── nginx.conf
            ├── db
            │   └── schema.sql
            ├── values.yaml
            └── resources.yaml
```

- 引入resources.yaml

  IaC3.0 支持当meta.yaml不编写components时，通过提取具体规格目录（如：cn_dev）的直接子目录（也称为组件目录）里的resources.yaml来自动构造出meta.yaml的components。

  引入resources.yaml后目录结构样例如下：

```
├── meta.yaml                    # IaC主体描述文件，内容不包含components
├── WiseEyeChaosMonkeyMgrService    # 组件一：ChaosMonkey管理服务
│   └── resources.yaml           # 组件一的资源列表
├── WiseEyeChaosMonkeyPortal       # 组件二：ChaosMonkey网页服务
│   └── resources.yaml           # 组件二的资源列表
├── environment                  # 组件三：环境
│   └── resources.yaml           # 组件三的资源列表
└── functions                   # 组件四：函数
    └── resources.yaml           # 组件四的资源列表
```

- 使用文件引用语法

  尽管meta.yaml已支持分散到诸多的组件目录中，但是每个组件仍然存在更新频率不同的配置。例如：微服务所用的镜像版本号更新频率较高。IaC3.0支持使用文件引用语法：允许YAML通过$ref键值对跨文档引用其他文本文件的内容。应用$ref语法，可以自由实现将资源、属性分散到更多文件中去。建议将更新频率较高的属性分散到被引用文件中，保持主体文件的结构稳定。

  – YAML文档整体引用

    resources.yaml原始内容：

```
# resources.yaml
- name: virtualapp
  type: Wisecloud::Nuwa::Runtime::MicroService
  properties:
    configs:
      - name: common.java.http.ssl_protocols
        value: TLSv1.2
        sensitive: false
      - name: common.java.http.enabled_cipher_suites
        value: aes_128_gcm|aes_256_gcm
        sensitive: false
    globalConf: |-
      #user  slb slb;
      worker_processes auto;
      #worker_cpu_affinity 0001 0010 0100 1000;
      pid        logs/nginx.pid;
      ......
     #  yaml 文件中多行字符串可以使用|保留换行符，如：globalConf
     # |：文中自动换行，默认仅保留一行空行
     # |+：文中自动换行，保留字符串后面所有的空行
     # |-：文中自动换行，删除字符串后面所有的空行
```

    将 properties > configs下的内容，放置于business_config.yaml。

```
# business_config.yaml
- name: common.java.http.ssl_protocols
  value: TLSv1.2
  sensitive: false
- name: common.java.http.enabled_cipher_suites
  value: aes_128_gcm|aes_256_gcm
  sensitive: false
```

    则resources.yaml 可修改为：

```
# resources.yaml
- name: virtualapp
  type: Wisecloud::Nuwa::Runtime::MicroService
  properties:
    configs:
      $ref: 'config/business_config.yaml#' # 运行时，会将config/business_config.yaml的整个YAML文档对象，替换掉整个$ref键值对
```

    将properties > globalConf下的内容，放置于nginx.conf

```
# nginx.conf
#user  slb slb;
worker_processes auto;
#worker_cpu_affinity 0001 0010 0100 1000;
pid        logs/nginx.pid;
......
```

则 resources.yaml 可修改为：

```
# resources.yaml
- name: virtualapp
  type: Wisecloud::Nuwa::Runtime::MicroService
  properties:
    configs:
      $ref: 'config/business_config.yaml#' # 运行时，会将config/business_config.yaml的整个YAML文档对象，替换掉整个$ref键值对
    globalConf:
      $ref: 'config/nginx.conf' # 运行时，引用的是config/nginx.conf这个配置文件内容所构成的字符串
```

- YAML文档片段引用

  若values.yaml存在以下内容：

  ```
  values:
    example_name:
      peak_tps: 200000
  ```

  则在resources.yaml，可按照如下方式进行引用200000。

  ```
  $ref: 'values.yaml#/values/example_name/peak_tps'
  ```

- 文本内容字符串引用

  ```
  $ref: 'db/schema.sql'    #引用的是schema.sql这个文本文件内容所构成的字符串
  ```

带 global 的多文件描述结构

Spec包通过不同规格目录来描述同一个服务在不同用途环境下所需的基础设施。但是，同一服务的不同的规格仍然存在大量相同的配置，需要一种机制来完成不同规格间配置的复用。因此，IaC支持放置一个global目录，其与specs目录同级，用于放置被所有规格目录所复用的配置文件。而各具体规格目录，只需包含与 global 目录的增量差异文件即可。

当某个规格被选用于部署时，会先将该规格目录下所有文件与global目录进行合并，得到该规格目录最终的所有配置文件，再进行部署动作。

合并策略：若文件的相对路径相同，则规格目录下的文件保留， global目录下的文件被覆盖，其他文件则共存。

带global的多文件描述结构样例如下：

```
├── package.json                    # 包描述文件（必须）
├── global                          # global目录：放置所有规格目录所复用的配置文件
│   │
│   ├── meta.yaml                   # IaC 主体描述文件，内容不包含 components
│   ├── WiseEyeChaosMonkeyMgrService    # 组件一：ChaosMonkey 管理服务，名称需要为该服务下的微服务名称。
│   │   └── resources.yaml          # 组件一的资源列表
│   ├── WiseEyeChaosMonkeyPortal        # 组件二：ChaosMonkey 网页服务
│   │   └── resources.yaml          # 组件二的资源列表
│   ├── environment                 # 组件三：环境
│   │   └── resources.yaml          # 组件三的资源列表
│   └── functions                   # 组件四：函数
│       ├── resources.yaml          # 组件四的资源列表
│       └── values.yaml             # 组件四的values.yaml
│
└── specs                           # 规格总目录（必须）
    ├── cn_dev_default              # cn_dev_default 规格目录，一般可用于描述一个开发用途的服务环境所使用的基础设施
    └── cn_product_default          # cn_product_default 规格目录，一般可用于描述一个生产用途的服务
```

```
环境所使用的基础设施
        └── functions              # 与global下functions目录的相对路径一致
            └── values.yaml        # 用于覆盖global下functions目录的values.yaml
```

2.4 IaC Spec 包典型目录结构

IaC Spec包用于描述环境。

目录结构介绍

表 2-1 IaC Spec 包结构说明

位置	类型	个数	描述
iacspec_{service}_{version}.zip	文件	1	IaC压缩包。
└── package.json	文件	1	包描述文件，相关说明请参见**包描述文件介绍**。
└── global/	文件夹	1	全局默认的IaC描述，包含完整文件结构，放置被所有规格目录所复用的配置文件。
│ └── meta.yaml	文件	1	变更策略描述，相关说明请参见**在IaC代码中定义流水线**。
│ └── environment/	文件夹	1	定义component1，公共资源。
│ └── resources.yaml	文件	1	公共资源列表，相关说明请参见**在IaC代码中声明资源**。
│ └── values.yaml	文件	1	公共资源参数值，在resources.yaml中通过$ref的方式来引用。
│ └── {microservice}/	文件夹	0-N	定义component2，微服务资源。
│ └── resources.yaml	文件	1	微服务资源列表，相关说明请参见**在IaC代码中声明资源**。
│ └── values.yaml	文件	1	微服务资源参数值，在resources.yaml中通过$ref的方式来引用。
│ └── configs/	文件夹	1	微服务配置目录。
│ └── config_schema.yaml	文件	1	声明微服务的业务配置项属性，敏感业务配置项需要声明，非敏感配置项可以不声明。在resources.yaml中通过$ref的方式来引用。

位置	类型	个数	描述
│ └── {cluster}_config_records.yaml	文件	0-N	微服务的业务配置项，在resources.yaml中通过$ref的方式来引用。
└── specs/	文件夹	1	环境特定的IaC描述，结构与global相同，但仅包含与global有差异的文件。
│ └── cn_product_cbu/	文件夹	1	中国区生产环境，命名采用站点级Cloud Map的名称，可以在环境管理界面查看可选的站点级Cloud Map名称列表。
│ └── environment/	文件夹	0-1	环境公共资源。
│ └── values.yaml	文件	0-1	公共资源参数值。
│ └── {microservice}/	文件夹	0-N	微服务资源。
│ └── values.yaml	文件	0-1	微服务资源参数值。
│ └── configs/	文件夹	0-1	微服务配置目录。
│ └── {cluster}_config_records.yaml	文件	0-N	微服务的业务配置项。
│ └── aaa_product_cbu/	文件夹	1	亚非拉生产环境。
│ └── eu_product_cbu/	文件夹	1	欧洲生产环境。

global 与 specs 的协同关系

- global文件夹：放置被所有规格目录所复用的配置文件。

  global文件夹里面的微服务都可以被规格文件夹specs中的代码复用（可根据meta.yaml指定复用哪些微服务，取决于在相应环境的部署规划）。

  global文件夹的作用类似于Java中的父类，spec类似于继承了global的子类，实际部署时还是使用的specs中的文件，但specs中的文件可以继承和复用global文件。

  - meta.yaml：描述变更的组件与过程。
  - {microservice}：描述要变更的微服务。
    - resources.yaml：微服务变更的主体文件，其他所有的values.yaml、config文件夹中的yaml等文件都围绕此文件展开。文件名必须为resources.yaml。
    - 其他文件：为变量配置文件，其定义的内容都会被resources.yaml引用，文件名称可自定义。
- spec文件夹：同一个服务在不同用途环境下所需配置文件（基础设施）。这个文件目录是必须的。

  specs是在环境上部署服务时，最终使用的配置文件，当部署服务时，第一关注点和入口就是specs。

specs目录下的规格文件夹，命名采用站点级Cloud Map的名称（cn_product_cbu、eu_product_cbu）。可以在环境管理界面查看可选的站点级Cloud Map名称列表。

当某个规格被选用于部署时，会先将该规格目录下所有文件与global目录进行合并，得到该规格目录最终的所有配置文件，再进行部署动作。

合并策略：如果文件的相对路径相同，则规格目录下的文件保留，global目录下的文件被覆盖，其他文件则共存。

global目录应包含完整的YAML内容，即：其下meta.yaml通过$ref引用的YAML内容都存在于global目录中。这样能确保即使规格目录为空，也能引用到global目录的默认参数，从而完成部署。

此处WiseChaos的IaC代码为例，详细描述代码运行原理及涉及的各个IaC文件的作用，微服务WiseChaos的整体IaC代码结构如下：

```
iac3.0                                  # IaC3.0代码根目录：目录名字可自定义
└─ WiseEyeChaosMonkeyService             # 此级目录为服务级目录，名字与所部署的服务名称相同
   ├─ global                             # global目录：放置所有规格目录所复用的配置文件
├─ WiseEyeChaosIssueMgrService
├─ WiseEyeChaosManageService
├─ WiseEyeChaosMonkeyExecutor
│  └─ config
│        ├─ business_config.yaml
│        ├─ env.yaml
│        ├─ envs_dynamic.yaml
│        ├─ hosts.yaml
│        └─ sidecar_aiops_param.json
│     └─ resources.yaml
│  └─ values.yaml
├─ WiseEyeChaosMonkeyPortal
      │  └─ config
      │        └─ sidecar_aiops_param.json
      │  └─ resources.yaml
      │  └─ values.yaml
└─ WiseEyeChaosPortal
      └─meta.yaml
   ├─specs              # 规格文件，描述了每个规格的定制化需求，最终部署的时候以specs文件为准
      ├─cn_product_cbu
         ├─WiseEyeChaosMonkeyExecutor
      │  └─config
            ├─ envs_dynamic.yaml
            ├─ hosts.yaml
            └─ sidecar_aiops_param.json
      └─values.yaml
   ├─WiseEyeChaosMonkeyPortal
            │  └─config
                  └─ sidecar_aiops_param.json
            └─values.yaml
            └─ meta.yaml
      ├─aaa_product_cbu
      └─eu_product_cbu
   └─package.json
```

当对此微服务的IaC代码打包合并时：

- specs中存在而global中不存在的文件，使用specs中的文件。
- specs中不存在而global中存在的文件，使用global中的文件。
- specs和global中都存在的，则使用specs中的文件。

以/iac3.0/WiseEyeChaosMonkeyService/specs/cn_product_cbu为例，在specs中的cn_product_cbu目录和global中都有WiseEyeChaosMonkeyExecutor和WiseEyeChaosMonkeyPortal微服务文件夹。

合并后最终呈现的文件目录如下：

“覆盖”指的是完全取代，不是内容合并。

```
cn_product_cbu
├── meta.yaml                    # 来自于specs目录，覆盖global中的文件
├── WiseEyeChaosMonkeyExecutor
│   ├── values.yaml              # 来自于specs目录，覆盖global中的文件
│   ├── resources.yaml           # 来自于global中的文件
│   └── config
│       ├── business_config.yaml    # 来自于global中的文件，因为specs中不存在
│       ├── env.yaml             # 来自于global中的文件，因为specs中不存在
│       ├── envs_dynamic.yaml       # 来自于specs目录，覆盖global中的文件
│       ├── hosts.yaml           # 来自于specs目录，覆盖global中的文件
│       └── sidecar_aiops_param.json # 来自于specs目录，覆盖global中的文件
├── WiseEyeChaosMonkeyPortal
│   ├── values.yaml              # 来自于specs目录，覆盖global中的文件
│   ├── resources.yaml           # 来自于global中的文件，因为specs中不存在
│   └── config
│       └── sidecar_aiops_param.json # 来自于specs目录，覆盖global中的文件
├── WiseEyeChaosIssueMgrService      # 来自于global中的文件，因为specs中不存在
├── WiseEyeChaosManageService        # 来自于global中的文件，因为specs中不存在
└── WiseEyeChaosPortal           # 来自于global中的文件，因为specs中不存在
```

2.5 IaC Patch 包典型目录结构

IaC Patch包用于描述环境中的一个组件。IaC Patch包典型目录结构如下：

表 2-2 IaC Patch 包结构说明

位置	类型	个数	描述
iacpatch_{microservice}_{version}.zip	文件	1	IaC压缩包。
└── package.json	文件	1	包描述文件，相关说明请参见**包描述文件介绍**。
└── global/	文件夹	1	全局默认的IaC描述，包含完整文件结构。全局默认的IaC描述，包含完整文件结构，放置被所有规格目录所复用的配置文件
│ └── resources.yaml	文件	1	微服务资源列表，相关说明请参见**在IaC代码中声明资源**。
│ └── values.yaml	文件	1	微服务资源参数值，在resources.yaml中通过$ref的方式来引用。
│ └── configs/	文件夹	1	微服务配置目录。
│ └── config_schema.yaml	文件	1	声明微服务的业务配置项属性，敏感业务配置项需要声明，非敏感配置项可以不声明。在resources.yaml中通过$ref的方式来引用。

位置	类型	个数	描述
│ └── {cluster}_config_records.yaml	文件	0-N	微服务的业务配置项，在resources.yaml中通过$ref的方式来引用。
└── specs/	文件夹	1	环境特定的IaC描述，结构与global相同，但仅包含与global有差异的文件。
│ └── cn_product_cbu/	文件夹	1	中国区生产环境，命名采用站点级Cloud Map的名称，可以在环境管理界面查看可选的站点级Cloud Map名称列表。
│ └── values.yaml	文件	0-1	微服务资源参数值。
│ └── configs/	文件夹	0-1	微服务配置目录。
│ └── {cluster}_config_records.yaml	文件	0-N	微服务的业务配置项。
│ └── aaa_product_cbu/	文件夹	1	亚非拉生产环境。
│ └── eu_product_cbu/	文件夹	1	欧洲生产环境。

IaC Patch包样例：

```
└── iacpatch_WiseEyeDeployConsoleService_1.1.2.zip
    ├── global/                          # global目录：放置所有规格目录所复用的配置文件
    │   ├── config/                      # 微服务配置目录
    │   │   ├── config_records.yaml
    │   │   └── config_schema.yaml
    │   ├── resources.yaml                 # 微服务的资源列表
    │   ├── packages.yaml
    │   └── values.yaml                  # 变量定义文件，被resources.yaml引用
    ├── specs/                           # 环境特定的IaC描述，结构与global相同，但仅包含与global有差异的
文件
    │   ├── eu_product_cbu/...             # 欧洲生产环境
    │   ├── aaa_product_cbu/...            # 亚非拉生产环境
    │   └── cn_product_cbu/                # 中国区生产环境，命名采用站点级Cloud Map的名称，可以在环境
管理界面查看可选的站点级Cloud Map名称列表
    │       ├── values.yaml              # 变量定义文件，被resources.yaml引用
    │       └── config/                  # 微服务配置目录
    │           └── config_records.yaml
    └── package.json                     # 包描述文件（必须）
```

2.6 在 IaC 代码中声明资源

定义component是IaC将一个环境的资源组织起来的方式，我们可以把同一类资源组织起来成为一个component。所有被IaC定义的资源必须属于某一个component。一个component下可以定义多个资源，所有的资源描述都存放于resources.yaml中，资源的type和name构成资源的唯一标记，以列表的形式存在。

component 定义

定义component是IaC将一个环境的资源组织起来的方式，我们可以把同一类资源组织起来成为一个component。所有被IaC定义的资源必须属于某一个component。在以下样例中，IaC代码定义了WiseEyeChaosMonkeyMgrService和WiseEyeChaosMonkeyPortal两个component。

```
├── global/
│   ├── meta.yaml
│   ├── WiseEyeChaosMonkeyMgrService/...
│   └── WiseEyeChaosMonkeyPortal/...
```

资源定义

一个component下可以定义多个资源，所有的资源描述都存放于resources.yaml中，资源的type和name构成资源的唯一标记，以列表的形式存在。

在以下样例中，WiseEyeChaosMonkeyPortal下的resources.yaml中定义了该组件下的全量资源。

图 2-1 资源定义

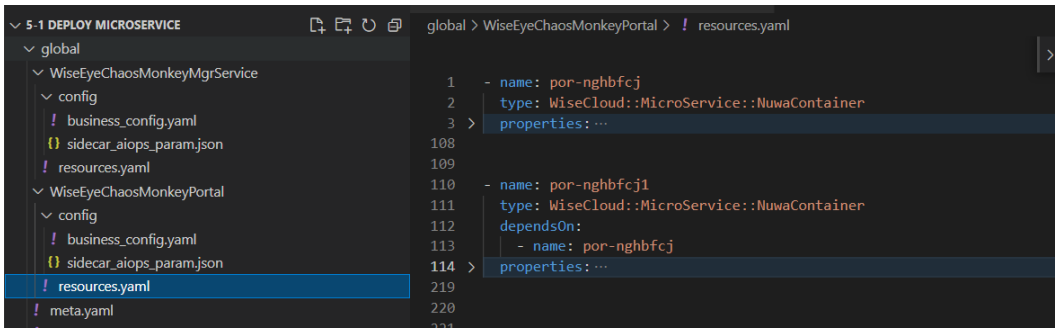
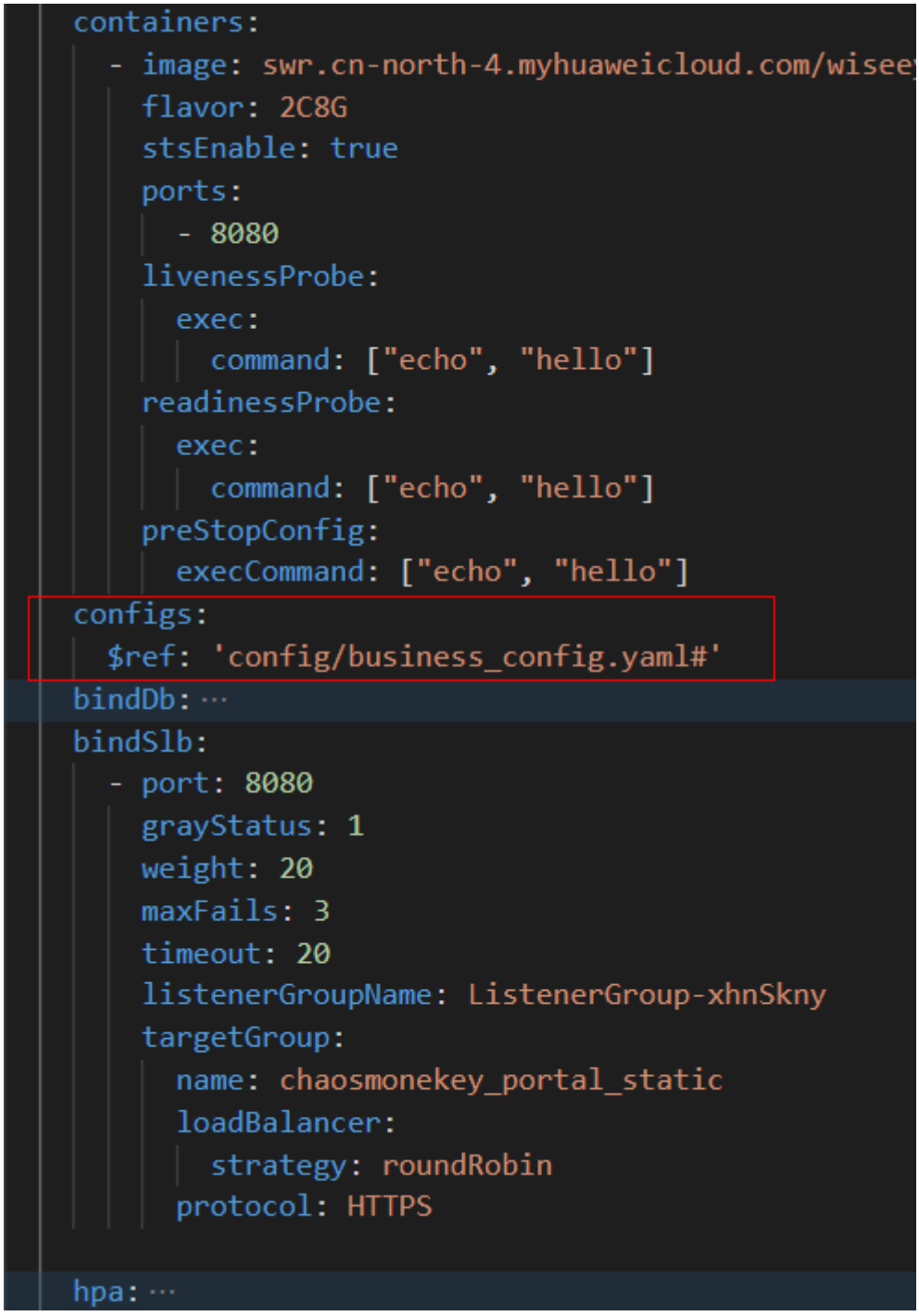


表 2-3 resources.yaml 字段说明

字段	含义
name	微服务平台显示的资源名称，最大长度为16字符。
type	资源类型。支持配置管理、NUWA Container及SLB。
properties	属性值，包含资源的详细参数。详细参数介绍请参见**IaC资源参数介绍**。

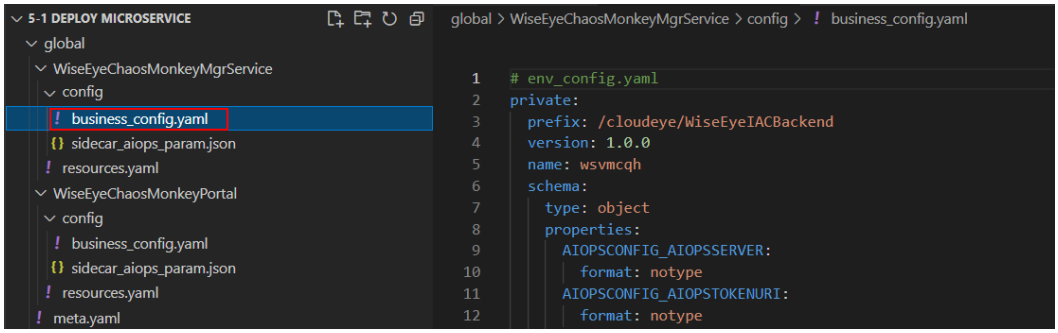
如果资源文件过大可以通过引用的方式对文件进行拆分及复用如图，我们可以把配置的定义放在config/business_config.yaml文件下。

图 2-2 resources.yaml 文件拆分



business_config.yaml文件可以复用：

图 2-3 business_config.yaml 文件复用



component 内部资源编排

component内部允许同一个资源出现多次，这表示同一个资源的不同变更阶段，这一场景下该资源的所有节点必须声明alias字段并且alias取值必须在component内全局唯一，同一个资源的所有alias之间必须显式地在dependsOn字段中声明串行依赖。

component的resources属性中描述资源列表，通过设置资源的dependsOn属性描述对其他资源的依赖。

dependsOn是列表类型，每个元素使用type、name、alias等字段描述对其他资源的引用，其中name字段是必填字段，type、alias是可选字段；component解析某资源的dependsOn时，会根据元素属性从相同component中搜索满足条件的资源作为当前资源的依赖。

部分资源之间已经有隐式引用关系，系统自动添加dependsOn，不需要再显式声明。每种类型的哪些属性隐含引用关系，可以参考其文档，具体请参见**IaC资源参数介绍**。

资源不能有循环依赖（A dependsOn B,B dependsOn C,C dependsOn A），不能依赖自己。

以下示例定义了两个资源，一个名为chaosmonkey-elb的ELB，一个名为chaosmonkey-slb的SLB；chaosmonkey-slb依赖于chaosmonkey-elb。在变更时，先变更chaosmonkey-elb，变更成功后再变更chaosmonkey-slb，如果chaosmonkey-elb变更失败，则不会变更chaosmonkey-slb。

```
- name: chaosmonkey-elb                # 资源名称
  type: WiseCloud::LoadBalancer::ELBV2   # 资源类型
  properties:
    listeners:
      - name: listener
        protocol: HTTP
        protocolPort: 80
        poolName: pool_a
    pools:
      - name: pool_a
        protocol: HTTP
- name: chaosmonkey-slb                 # 资源名称
  type: WiseCloud::LoadBalancer::SLB     # 资源类型
  dependsOn:                            # 定义对其他资源的依赖
    - name: chaosmonkey-elb             # 依赖的资源名称
  properties:
    elbName: chaosmonkey-elb
    elbPoolNames: ["pool_a"]
    deployVersion: 1.4.12
    slbConfigs:
      targets:
        - clusterName: mgr
      routes:
```

```
    - location: /
      target: mgr
```

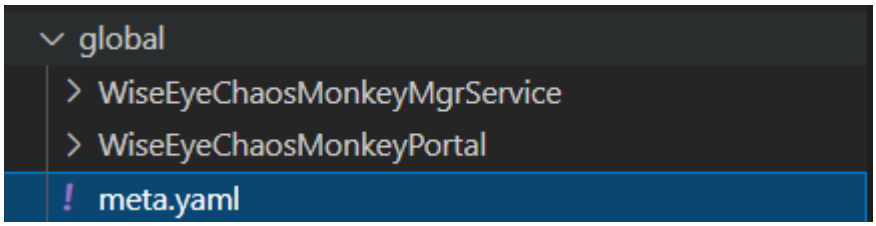
2.7 在 IaC 代码中定义流水线

IaC代码中的流水线可以由用户定义，用户可以根据自己的需求定义整个环境在变更时的执行过程，在变更执行过程中，系统只会变更被流水线引用的资源，本章介绍如何定义流水线。

在哪里定义流水线

component间的编排在spec包中的meta.yaml文件中描述，涉及applyPipeline/pipelines两个字段。pipelines中支持定义多个流程，applyPipeline描述本次变更要使用的流程。pipeline能力丰富，通过设计pipeline可以实现精巧的多阶段部署、部分变更、主动暂停等复杂场景的变更编排。

图 2-4 meta.yaml



如何定义流水线

meta.yaml文件涉及applyPipeline/pipelines两个字段。pipelines中支持定义多个流程，applyPipeline描述本次变更要使用的流程。pipeline能力丰富，通过设计pipeline可以实现精巧的多阶段部署、部分变更、主动暂停等复杂场景的变更编排。

样例如下：

```
type: WiseCloud::Environment      # 保留字，声明这是一个针对环境的IAC代码
applyPipeline: default            # 代码中指定的默认pipeline，指定的pipeline必须是在pipelines中声明的
pipelines:                        # 列表，可以定义多个pipeline，并在执行任务时选择
  - name: default                 # pipeline名称
    action: Serial                # 此pipeline执行任务的策略 Serial（串行）/ Parallel（并行）
    tasks:                        # 声明流水线的子任务，通过声明pipeline的任务，实现对资源部署流程的编排
      - name: apply-chaosmonkey-stage1
        action: Serial
        tasks:
          - name: deploy
            action: Serial
            tasks:
              - name: apply-chaosmonkey-por
                action: Apply
                component:
                  name: WiseEyeChaosMonkeyPortal
  - name: all
    action: Serial
    tasks:
      - name: apply-chaosmonkey-stage1
        action: Serial
        tasks:
          - name: deploy
            action: Serial
            tasks:
              - name: apply-chaosmonkey-por
```

```
          action: Apply
          component:
            name: WiseEyeChaosMonkeyPortal
  - name: apply-chaosmonkey-stage2
    action: Serial
    tasks:
      - name: deploy
        action: Serial
        tasks:
          - name: apply-chaosmonkey-mgr
            action: Apply
            component:
              name: WiseEyeChaosMonkeyMgrService
```

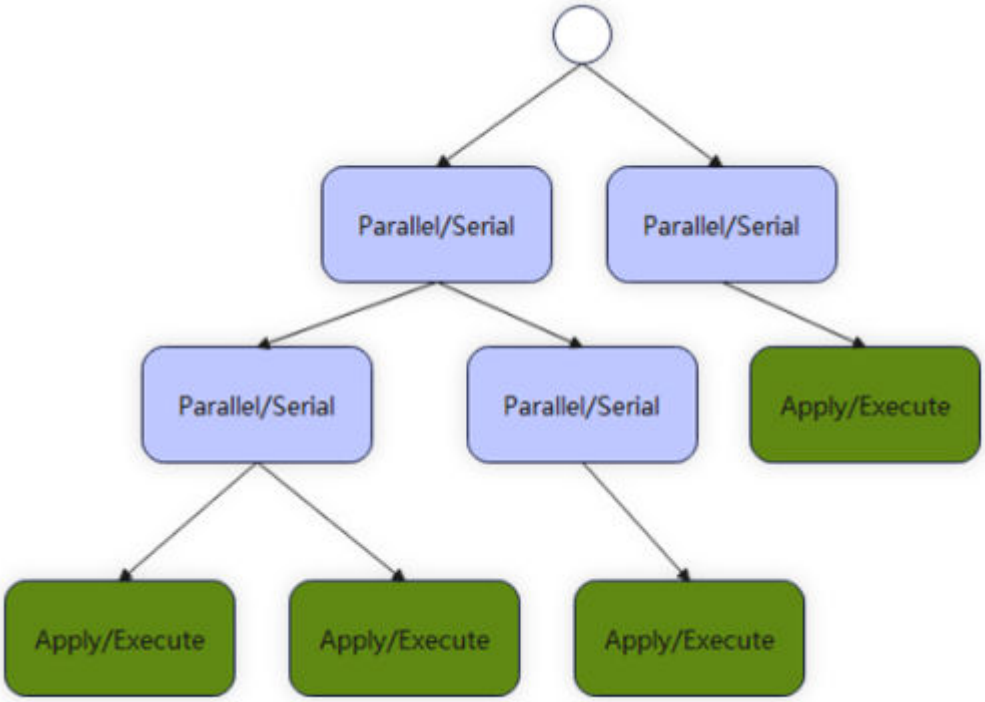
表 2-4 meta.yaml 字段说明

字段	说明
type	描述当前环境类型，当前为固定值WiseCloud::Environment。
applyPipeline	定义默认选用的组件编排流水线名称，当前默认使用environment-deploy。
pipelines	pipelines中支持定义多个流程。

pipeline 的任务声明

name/action都是一个流水线的基本信息，任务的编排是需要声明不同类型的task实现；当前支持的任务类型包括：Serial（串行）/Parallel（并行）/Apply（声明式任务执行），其中，Serial/Parallel不涉及资源部署，我们仅通过此类任务将需要部署的资源串联起来，从而环境变更的过程。如果把pipeline比作一棵树的声明，那么所有tasks都是这棵树的子节点，Apply是这棵树的叶子节点，只有叶子节点被执行时才会对环境产生影响。

图 2-5 pipeline 任务声明



- Serial（串行）/Parallel（并行）任务定义

  定义Serial（串行）/Parallel（并行）任务将环境的资源变更流程串联起来。

  – Serial，串行执行，其参数tasks下的子任务会根据定义的顺序依次执行。

  – Parallel，并行行执行，其参数tasks下的子任务会同时执行。

```
- name: deploy    # 任务名称
  action: Serial  # 此任务下其子任务的执行策略 Serial（串行）/ Parallel（并行）
  tasks:          # 子任务，必填，子任务可以是任意类型的的任务；
    - name: apply-chaosmonkey-mgr
      action: Apply
      component:
        name: WiseEyeChaosMonkeyMgrService
```

- Apply（声明式任务执行）任务定义

  我们可以通过定义Apply类型的任务声明一个环境的资源，这些资源可以是NuwaContainer，数据库，配置等等；

```
- name: apply-chaosmonkey-por      # 任务名称
  action: Apply                    # 任务类型
  inform:                         # 执行后通知配置
    message: finish stage2 notification
  confirm:                        # 执行前确认配置
    message: pause before execute task
  component:                      # 需要执行变更的component
    name: WiseEyeChaosMonkeyPortal # component名称
    resources:                    # component下需要执行变更的资源
      - name: por-nghbfcj1
        type: WiseCloud::MicroService::NuwaContainer #
        properties:
          grayStage:
            grayInstances: 100    # 灰度百分比
            grayStatus: 2         # 表灰度
            grayProcess: FINISHED # 表完成
```

表 2-5 组件编排流水线（pipeline）对象定义

字段名称	字段类型	必选	描述
name	String	是	流水线的名称，用户可自定义。
action	String	是	编排方式，可选值Serial/Parallel。 ● Serial，串行执行，其参数tasks下的子任务会根据定义的顺序依次执行。 ● Parallel，并行执行，其参数tasks下的子任务会同时执行。
tasks	List<PipelineTask>	是	声明流水线的子任务，通过声明pipeline的任务，实现对资源部署流程的编排。

表 2-6 PipelineTasks 字段说明

字段名称	字段类型	必选	描述
name	String	否	子任务的名称，用户可自定义。
action	String	是	编排方式，可选值：Serial/Parallel/Apply。
tasks	List<PipelineTask>	否	编排方式为Serial/Parallel时必选。
component	ApplyComponent	否	需要执行变更的component，编排方式为Apply时必选。

任务的数据结构在设计上支持了无限层级嵌套，配合action属性可以实现任意的任务串并行编排。

表 2-7 component 字段说明

字段名称	必选	描述
name	是	组件名称，填写的值必须为已声明的组件，否则在校验阶段会报错。

字段名称	必选	描述
resources	否	选择变更的资源，选填，该参数可用于部分部署。 如果不填写则会变更此component下所有资源，使用name+type+alias确认唯一资源（alias是资源的别名，如果component下不存在同名资源则不需要填写）。 如果填写component中不存在的资源，系统会在校验阶段报错；resources下可以定义properties，这样在执行此任务时，这些properties的值就会去覆盖你在resources.yaml下定义的资源参数。

2.8 包描述文件介绍

包描述文件package.json样例如下：

```
{
  "type": "iacspec",                                  # 代码包类型
  "name": "service/1180196813870297088",              # 代码包名称，格式：service/{服务Id}（必须）
  "version": "1.0.0"                                  # 代码包版本号（必须）
}
```

表 2-8 package.json 字段说明

位置	类型	必填	描述
type	string	是	包类型，常量：iacspec或iacpatch。
name	string	是	包名称 ● iacspec包名称格式：service/{service-id}，其中service-id为服务ID。 ● iacpatch包名称格式：service/{service-id}/{component-name}，其中{service-id}为服务ID，{component-name}为组件名称。 您可以在AppStage运维中心右上角的个人账号信息管理中，选择“租户管理”，查看服务ID。
version	string	是	版本号。

2.9 IaC 资源参数介绍

2.9.1 NUWA Container

2.9.1.1 参数配置说明

基础参数

表 2-9 基础参数

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
name	string	必选	-	IaC3.0资源名	只能包括数字、字母、'-'、'_'、'.'，必须以字母开头，字母或数字结尾。长度2-64。 **说明** 如果没有配置clusterName，则资源名会被当做微服务集群名，参数规范则会以微服务集群名的为准。	WiseCloudFGCEventBuilderService_cluster1
type	string	必选	-	BaaS服务类型，operator要求必填，固定为WiseCloud::MicroService::NuwaContainer	固定为WiseCloud::MicroService::NuwaContainer	WiseCloud::MicroService::NuwaContainer
microserviceName	string	必选	-	微服务名称	微服务名称	WiseCloudFGCEventBuilderService

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
clusterName	string	非必选（建议填写）	默认与name值相同	微服务集群名	只能包括数字、字母、'-'、'_'、'.'，必须以字母开头，字母或数字结尾。长度2-56。 **说明** 不填与name值相同，由于两字段限制不同，超过限制会报错。	cluster1
replicas	int	必选	-	Pod副本数	整数类型 **说明** • 多AZ要配置AZ的倍数，如果部署了双AZ，那么此处要配置为2的倍数。 • 如果使用了evs盘或者elb，为保证滚动升级每个AZ至少保留一个节点，那么单AZ至少要配置为2，双AZ至少配置为4。	1
pdbMaxUnavailable	string	非必选	-	Pod干扰预算	整数百分比，整数范围为[1, 50]	-
terminationGracePeriodSeconds	integer	非必选	-	优雅下线宽限期	1-65535	-

示例：

```
- name: WiseCloudFGCEventBuilderService_cluster1
  type: WiseCloud::MicroService::NuwaContainer
  properties:
    microserviceName: WiseEyeChaosMonkeyExecutor
```

```
clusterName: cluster1
replicas: 5
```

挂载信息

表 2-10 挂载信息

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
name	string	必选	-	• log_volume：日志卷 • data_volume：data卷 • sfs_volume：sfs卷 **说明** sfs的卷命名并非固定，此处的名称主要还是供container的volumeMounts引用。	• log_volume 每个POD的日志卷大小，log_volume_type不配置或者配置为local时，代表使用集群节点的本地盘，有如下注意事项：容器场景下对/opt/huawei/logs的日志卷大小限制为100G，当日志磁盘写满之后，k8s会驱逐该POD并重新拉起一个新的POD，在此过程中，会影响业务。请在log配置文件中配置绕接策略对总的日志大小不超过此限制，由于NUWA及中间件也会记录一部分日志，建议业务配置保存的log总大小不要超过90G。另外业务需要将所有的日志配置到AIOps sidecar日志服务中，以通过AIOps日志服务进行日志采集，否则POD销毁时会造成日志丢失。 如果log_volume_type配置为evs，则根据实际的大小进行配置。 • data_volume data卷大小，默认挂载了/opt/huawei/data的路径，最大支持配置为5Gi。data_volume_type配置为evs则不受此限制。	volumes: - name: log_volume size: 12Gi type: local - name: data_volume size: 2Gi type: local
size	string	必选	-	定义存储空间		

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
type	string	可选	默认是local，代表使用集群节点本地存储。	• evs • sfs 参见右侧样例	sfs: volumes: - name: sfs_volume id: 11a701c9-529f-4992-9291-bc47bbf5b4c5 # 如果是已有的存储，该值代表存储的资源id type: sfs shareLocation: xxx # 共享路径 mountOptions: ["vers=3","hard","nolock"]	local

示例：

```
- name: WiseCloudFGCEventBuilderService_cluster1
  type: WiseCloud::MicroService::NuwaContainer
  properties:
    volumes:
      - name: log_volume
        size: 12Gi
        type: local
      - name: data_volume
        size: 2Gi
        type: local
```

容器配置

表 2-11 容器配置

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
image	string	必选	-	镜像地址 Runtime已预置如下镜像仓库地址，如果业务是从如下镜像仓库地址下载镜像，则只需要从组织名开始填写。 北京四：swr.cn-north-4.myhuaweicloud.com 乌兰察布一：swr.cn-north-9.myhuaweicloud.com 华南广州：swr.cn-south-1.myhuaweicloud.com	长度：1-256 **说明** 镜像仓库（:前的一串）只能是小写。	-

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
ports	int[]	可选	0	如果使用Cloud Map，将此端口填写为在Cloud Map注册中心注册的端口 如果不使用Cloud Map，可将此项注释掉或者填写为0。 此参数只是起一个标识作用，并不是代表配置了此端口就代表端口一定打开。实际打开的端口以业务使用的为准，目前仅支持一个端口配置，数组类型是为后面可扩展。	端口范围：0-65535	ports: - 8080
flavor	string	-	"2C4Gi"	主容器CPU&memory的规格	长度范围：1-32。需要满足CPU和memory的规格，比如1C2G，2C4G，4C8G，8C16G等。	flavor: 2C4G

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
armFlavor	string	可选	"2C4Gi"	参考flavor 双AZ arm混部时独立控制arm配置 arm配置建议: 按照业内通用的指导，arm cpu算力相对于x86下降，具体下降指标和使用场景密切相关。 对于计算密集型的业务，可以考虑增加配置，IO密集型可以同规格，具体性能还是以各自业务的实际性能测试为准。	同上	同上
gpu	int	可选	null	主容器使用的GPU规格（GPU即显卡）；当前ERS管理的资源池中尚未提供GPU。	取值为整数，配1代表占用一块显卡，显卡不可分割。	-
stsEnable	bool	可选	TRUE	是否启用sts	-	TRUE
commandArgs	list(string)	可选	[]	启动参数	长度为0-256	-

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
volumeMounts	list(object)	可选	[]	此参数与volumes下的sfs挂载卷搭配使用，volumes中定义挂载卷的信息，此处引用并挂载在容器中运行	name要在volumes中有对应的值。	volumeMounts: - name: sfs_volume1 mountPath: /opt/huawei/sfs1 - name: sfs_volume2 mountPath: /opt/huawei/sfs2
envs	type = list(object({ name = string value = string }))	可选	[]	配置环境变量	name和value的长度为不超过5000。	envs: - name: "EVS_TEST" value: "test_ENV"
hostAliases	type = list(object({ hostName = string ip = string }))	可选	[]	配置hosts	-	hostAliases: - hostName: "a.b.com" ip:"1:1:1:1:1:1:1:1" - hostName: "c.d.com" ip:"2.2.2.2"

为保证业务容器的稳定运行和资源利用率的提升，建议业务容器采用标准化容器规格，标准容器规格如表2-12所示。

表 2-12 容器资源规格

vCPU (cpu request)	Memory, GiB (memory request/limit)	cpu和内存配比	备注
0.5	500M、1、2、4	1:1、1:2、1:4、1:8	X86和ARM
1u	1、2、4、8	1:1、1:2、1:4、1:8	X86和ARM

vCPU (cpu request)	Memory, GiB (memory request/limit)	cpu和内存配比	备注
2u	2、4、8、16	1:1、1:2、1:4、1:8	X86和ARM
3u	3、6、12、24	1:1、1:2、1:4、1:8	ARM专属规格
4u	4、8、16、32	1:1、1:2、1:4、1:8	X86和ARM
6u	6、12、24、48	1:1、1:2、1:4、1:8	ARM专属规格
8u	8、16、32	1:1、1:2、1:4	X86和ARM
12u	12、24、48	1:1、1:2、1:4	ARM专属规格
14u	14、28、52	1:1、1:2、1:4 （3.714）	X86和ARM
22U	22、44、76	1:1、1:2、1:4 （3.45）	ARM专属规格
28U	28、56、104	1:1、1:2、1:4 （3.714）	X86和ARM
42U	42、84、150	1:1、1:2、1:4 （3.571）	ARM专属规格

- 16C的规格请降低为14C，大于等于64G的内存请降低为52G或者48G。14C52G是标准规格，是从节点的分配率考虑的。
- 如果要使用0.5G的内存，需要在IaC里面配置为500M，不能配置为0.5G。
- 最小组网：CPU规格<=1u，且双AZ POD数量<=6个。

示例1（双AZ同构，或者不同构但同配置）：

```
- name: WiseCloudFGCEventBuilderService_cluster1
  type: WiseCloud::MicroService::NuwaContainer
  properties:
    containers:
      - image: swr.cn-north-4.myhuaweicloud.com/wiseeye/wiseeyechaosmonkeyservicewebsite:3.0.9.204.SP2
        flavor: 2C8G
        stsEnable: true
        ports:
          - 8080
```

示例2（arm、x86混部时，独立配置flavor）：

```
- name: WiseCloudFGCEventBuilderService_cluster1
  type: WiseCloud::MicroService::NuwaContainer
  properties:
    containers:
      - image: swr.cn-north-4.myhuaweicloud.com/wiseeye/wiseeyechaosmonkeyservicewebsite:3.0.9.204.SP2
        flavor: 2C8G
        armFlavor: 4C16G
        stsEnable: true
        ports:
          - 8080
```

容器健康检查

表 2-13 容器健康检查

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
readinessProbe: exec: command:	list(string)	必须配置其中一种，且只能配置一种	[]	命令行检查方式	命令行检查方式\http请求检查方式\tcp端口检查三种方式只能选择一种。	readinessProbe: exec: command: ["echo", "hello"]
readinessProbe: httpGet:	object		-	http请求检查方式	命令行检查方式\http请求检查方式\tcp端口检查三种方式只能选择一种。	readinessProbe: httpGet: path: /health port: 8080 scheme: HTTP
readinessProbe: tcpSocket: port:	int		-	tcp端口检查	命令行检查方式\http请求检查方式\tcp端口检查三种方式只能选择一种。	readinessProbe: tcpSocket: port:8080
livenessProbe: exec: command:	list(string)	必须配置其中一种，且只能配置一种	[]	命令行检查方式	命令行检查方式\http请求检查方式\tcp端口检查三种方式只能选择一种。	livenessProbe: exec: command: ["echo", "hello"]
livenessProbe: httpGet:	object		-	http请求检查方式	命令行检查方式\http请求检查方式\tcp端口检查三种方式只能选择一种。	livenessProbe: httpGet: path: /health.html port: 8080 scheme: HTTP
livenessProbe: tcpSocket: port:	int		-	tcp端口检查	命令行检查方式\http请求检查方式\tcp端口检查三种方式只能选择一种。	livenessProbe: tcpSocket: port:8080

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
initialDelaySeconds	int	可选	10s	表示容器启动多少秒之后开始探测，单位秒。默认值为10s。	不小于5	readinessProbe: exec: command: ["echo", "hello"] initialDelaySeconds: 10 periodSeconds: 20 successThreshold:1 failureThreshold: 3 timeoutSeconds: 5
periodSeconds	int	可选	20s	间隔周期，表示每多少秒探测一次容器，单位秒，默认值为20s。	不大于180	
successThreshold	int	可选	1	表示连续检测多少次成功后则记作成功。默认值为1。	不大于10，liveness探针只能为1。	
failureThreshold	int	可选	3	表示连续检测多少次失败当做是失败处理，并会重启容器。默认值为3。	不大于10	

示例：

```
livenessProbe:
  httpGet:
    path: /health
    port: 8080
    scheme: HTTP
  initialDelaySeconds: 20
  timeoutSeconds: 3
```

```
  periodSeconds: 10
  successThreshold: 1
  failureThreshold: 3
readinessProbe:
  httpGet:
    path: /health
    port: 8080
    scheme: HTTP
  initialDelaySeconds: 20
  timeoutSeconds: 3
  periodSeconds: 10
  successThreshold: 1
  failureThreshold: 10
      - name: APIGateway
        type: WiseCloud::Agent::APIGateway
        version: x.x.x.x
        flavor: 1C2G
        param:
          $ref: 'config/sidecar_apigw_param.json
```

sidecar 配置

表 2-14 sidecar 配置

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
sidecars	type = list(object)	-	[]	由于POD销毁或被驱逐时，日志文件会丢失，因此AIOps Log Sidecar为必选参数，需要业务配置此sidecar及时将日志进行采集。	• name：sidecar的名称，比如AIOps Log。 • type：sidecar的类型，详见代码示例。 • version：sidecar的版本。 • cpu：sidecar容器的cpu，配置举例：0.5 或者 2 或者 500m，配置为null或者""“时，代表使用系统默认值。 • memory：sidecar的内存，单位为MMi G	详见代码示例，按照业务需求配置所需的sidecar。

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
					Gi，配置举例：100M 0.5G 2G，配置为null或者”"时，代表使用系统默认值。注意M和Mi的区别，M是1000的倍比，Mi是1024的倍比。 • param：sidecar的配置参数，不同sidecar的配置参数不一样，具体可以参考各个sidecar的配置方式。	

示例：

```
sidecars:
    - name: AIOpsLog
      type: WiseCloud::Agent::AIOpsLog
```

```
    version: 3.3.0.100
    flavor: 0.4C500M
    param:
      $ref: 'config/sidecar_aiops_param.json'
  - name: RASP
    type: WiseCloud::Agent::RASP
    version: 2.2.0.102.SP5
    flavor: 0.3C500M
    param: ''
  - name: BIFlume
    type: WiseCloud::Agent::BIFlume
    version: x.x.x.x
    flavor: 0.2C500M
    param:
      $ref: 'config/sidecar_biflume_param.json'
  - name: APIGateway
    type: WiseCloud::Agent::APIGateway
    version: x.x.x.x
    flavor: 1C2G
    param:
      $ref: 'config/sidecar_apigw_param.json
```

网络配置

表 2-15 网络配置

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
isolatedDomain	string	可选	-	隔离域	只有CCE_TURBO集群支持，老CCE集群不支持。如果不配置，则使用ENS隔离域规划中对应的隔离域，见上面隔离域说明链接中的说明。	-

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
slbEnable	bool	可选	FALSE	是否启用slb	“否”仅针对第一次添加SLB信息时有效。 如果之前有部署过，并且配置为“是”，此时再修改为“否”是无效的。	-

示例：

```
- name: WiseCloudFGCEventBuilderService_cluster1
  type: WiseCloud::MicroService::NuwaContainer
  properties:
    network:
      slbEnable: true
      isolatedDomain: $ {创建的隔离域名称}
```

SLB 配置（可选）

表 2-16 SLB 配置

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
port	int32	必选	null	微服务在SLB上暴露的端口	-	8080
weight	int32	必选	null	微服务在SLB上负载的权重	-	20
timeout	int32	必选	null	微服务调用转发的超时时间	-	20
maxFails	int32	必选	null	微服务调用的失败次数	-	3

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例

targetGroup	object{
	name,
	loadBalancer
	}
		必选	""	微服务注册到SLB上的后端服务器组名	• 用于创建RouteRule，其中的name为必选，对应NuwaRuntime的backendClusterName，其它为可选。
					• 在routes配置的情况下，loadBalancer也属于必选，不配置routes的话，loadBalancer则是可选。
					• loadBalancer当前仅支持配置roundRobin（加权轮询）且不支持修改为其它策略，如需修
						name:
						chaosmonekey_portal_static
						loadBalancer:
						strategy:
						roundRobin

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
					改可登录SLB管理台修改。	
listenerGroupName	string	必选	""	微服务前端关联的SLB ListenserGroup实例IaC名称	对应NuwaRuntime中的slbServiceName，长度为[1, 64] SLB下需要存在这个监听，为空时不会创建RouteRule。	chaos_slb_listener_lhq
routes	array[location: string]	可选	[]	微服务的路由规则	不推荐使用，请直接使用slb的iac3.0创建。	/fgc/v1

示例：

```
- name: WiseCloudFGCEventBuilderService_cluster1
  type: WiseCloud::MicroService::NuwaContainer
  properties:
    bindSlb:
      - port: 8080
        grayStatus: 1
        weight: 20
        maxFails: 3
        timeout: 20
        listenerGroupName: chaos_slb_listener_lhq
        targetGroup:
          name: chaosmonekey_portal_static
          loadBalancer:
            strategy: roundRobin
```

证书配置

表 2-17 证书配置

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
certConfigs	type = list(object({ name = string tag = string }))	可选	[]	证书配置	• name：证书名称 • tag：证书tag	# 业务证书配置 certConfigs: name: "server" # 证书名称 tag: "default" # 证书tag

daemonSet

表 2-18 daemonSet

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
daemonSet	type = list(object)	可选	[]	目前仅支持AIOps DaemonSet和BI DaemonSet，挂载hostpath提供存储持久化到node的能力。	• name：AIOps or BI • type：DaemonSet的类型，详见代码示例 • enable：true|false，是否启用 • logPath：推送日志路径（非文件名），选填。 • AIOps固定为'/opt/huawei/logs' • limitSize：日志存储限制，选填，默认值为100G。仅AIOps生效。 • groups：日	详见代码示例

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
					志配置组名，必填。当前BI场景支持1个或多个分组，AIOps场景有且只能填写一个分组并且提前在AIOps管理面创建好。	

示例：

```
daemonSet:
    - name: AIOps
      type: WiseCloud::Agent::AIOps
      enable: true
      logPath: '/opt/huawei/logs'
      limitSize: 100G
      groups: ["logConfigGroupName"]
    - name: BI
      type: WiseCloud::Agent::BI
      enable: false
      logPath: '/opt/huawei/logs/bi'
      groups: ["ODS_V001_DM_service1", "ODS_V001_DM_service2"]
      paramJson: "{\"dataGroups\":[{\"dataGroup\":\"ODSName_BatchFileExampleDS\",\"agentType
\":\"batch\",\"batchConfig\":{\"datapushInputs\":{\"jobType\":\"file\",\"dayPeriod\":{\"startTime
\":\"10:00:00\",\"offset\":\"1\"},\"file\":{\"sources\":[{\"pattern\":\"/opt/huawei/hcy/*.txt\",\"filename
\":\"test.txt\"}]}},\"datapushOutput\":{\"postfix\":\"txt\",\"permitEmptyFile\":true},\"advanced\":
{\"extendFields\":{\"datapushInput.isUtc\":false,\"datapushInput.file.sourcePolicy
\":3,\"datapushInput.file.countThreshold\":0,\"datapushInput.file.sizeThreshold\":0,\"sendThreadCount
\":3,\"datapushInput.file.fileRetryTimes\":3,\"datapushInput.file.fileWaitTimes\":3}}}},{\"dataGroup
\":\"ODSName_StreamFileExampleDS\",\"agentType\":\"stream\",\"streamConfig\":{\"filebeatInputs\":{\"type
\":\"log\",\"enabled\":true,\"paths\":[\"/opt/huawei/logs/*.log\",\"/opt/huawei/logs/*.txt\"]},\"advanced\":
{\"extendFields\":{\"filebeatInputs.harvester_limit\":5,\"queue.mem.events
\":4096,\"queue.mem.flush.min_events\":2048}}}}]}"
      # 配置json格式化
```

业务配置

表 2-19 业务配置

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
configs	object(private = object(name = string prefix = string version = string schema = object records = object) public = object(name = string prefix = string))	否	{}	配置项的根字段，包含两个属性，分别是private和public，分别为业务配置项和公共配置项，其下各个字段的描述如下所示。	-	见下文样例
prefix	string	否	""	配置项的归属路径	仅限于publicConfig	/public/cloudeye/wiseEyeConfigService

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
schema	object(type = string properties = object(key1 = object(type = string description = string format = string)))	否	{}	配置项属性，properties属性为key-object格式，key是配置名称，object是配置项各项描述，其中format指配置项类型，默认为notype，如果是敏感配置项为sensitive。 如果是非敏感配置项，可以不在schema中声明，以减少维护工作量。	仅限于privateConfig	见下文样例

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
records	object(key1 = string key2 = string)	否	{}	描述配置，key-value格式，key为配置名称，value为配置值。 value仅支持字符串类型 如果是数字、布尔值、对象和数组，需要加单引号，例：'10'、'true'、'{"test": 1}'、'[1,2]'。	仅限于privateConfig	records: test: '{"a":"a","b":"b"}' timeout: '10' enableSa: 'true'
name	string	否	""	配置项名称，对应NuwaRuntime的Container[0].configTag，对应PublicConfig和PrivateConfig的name。 配合动态配置生效，需要nuwa基础镜像版本要保持在3.0.11版本以上，否则报错。	publicConfig/privateConfig均有 限制：8位以内的小写字母和数字	见下文样例

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
version	string	否	""	配置版本，对应NuwaRuntime的Container[0].configVeersion，对应PrivateConfig的version。 配合动态配置生效，需要nuwa基础镜像版本要保持在3.0.11版本以上，否则报错。	仅限于privateConfig	见下文样例

示例：

```
# resources.yaml
- name: WiseCloudFGCEventBuilderService_cluster1
  type: WiseCloud::MicroService::NuwaContainer
  properties:
    configs:
      $ref: 'config/business_config.yaml#'
# business_config.yaml
public:
  prefix: /com.huawei.wiseeye
  name: public5
private:
  version: 1.0.0      #（配合动态配置生效，需要nuwa基础镜像版本要保持在3.0.11版本以上，否则报错）
  name: fgcva          #（配合动态配置生效，需要nuwa基础镜像版本要保持在3.0.11版本以上，否则报错）
  schema:
    type: object
    properties:
      AIOPSCONFIG_AIOPSSERVER:      # 默认为format: notype，如果非敏感项，可以不填
        format: notype              # 默认为notype，如果非敏感项，可以不填
      AIOPSCONFIG_AIOPSTOKENURI:    # 敏感项必须填
        format: sensitive           # 敏感项必须填
  records:                         # 必填
    AIOPSCONFIG_AIOPSSERVER: https://XX.XX.XX.XX:XXXX/
    test: '{"a":"a","b":"b"}'   # 仅支持字符串类型，[]、{}的值，yaml会识别为对象和数组，必须加单引号
    timeout: '10'               # 仅支持字符串类型，数字和布尔值也要加引号
    enableSa: 'true'
```

滚动升级策略

表 2-20 滚动升级策略

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
rollingUpdateStrategy	type = object({ maxSurge = string maxUnavailable = string })	可选	maxSurge : "25%" maxUnavailable: 0	滚动升级策略配置 • maxSurge：滚动升级时最多可以多启动多少个pod。 • maxUnavailable：滚动升级时最大可以删除多少个pod。	整数：最小值为0 百分比：0% ~ 100% • maxSurge和maxUnavailable不能同时为0 • maxUnavailable不能设置为100或者100%，避免升级时集群没有可用的pod。	rollingUpdateStrategy: maxSurge = "50%" maxUnavailable = "25%"

优雅下线

表 2-21 优雅下线

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
preStopConfig	type = object({ execCommand = list })	必选	[]	优雅退出处理	-	preStopConfig: execCommand: ["/bin/bash", "-c", "sleep 20"]

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
terminationGracePeriodSeconds	int	可选	30	优雅退出宽限时间，此时间为整个POD的最大退出时间。	-	terminationGracePeriodSeconds: 30

灰度策略

表 2-22 灰度策略

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
grayStage	type = object({grayInstances = number grayProcess= stringgrayStatus = number })	可选	null	灰度升级策略配置。 ● grayInstances：灰度升级的实例数比例，范围为1~100。 ● grayStatus：灰度实例对接SLB时的状态，1：生产，2：灰度。 ● grayProcess：灰度状态，只能填"INGRAY"和"FINISHED"。	-	grayStage: grayInstances: 50 grayProcess: "INGRAY" grayStatus: 2

示例：

```
- name: WiseCloudFGCEventBuilderService
  type: WiseCloud::MicroService::NuwaContainer
  properties:
    grayStage:
      grayInstances: 50
      grayProcess: "INGRAY"
      grayStatus: 2
```

水平自动伸缩（HPA）

如果业务不需要使用hpa，请不要配置hpa相关参数。hpa开关配置关闭，也会创建hpa资源，只是不会生效扩缩容。

表 2-23 水平自动伸缩（HPA）

参数	说明	是否必填	备注
hpa_scale_disabled	是否禁用hpa，false表示不禁用；true表示禁用。	是	取值为false时开启hpa。
polling_interval	负载检测周期，单位秒。	否	-
origin_instances	初始副本数，不配置则使用min_instances值。	否	-
min_instances	最小副本数	是	不配置默认为1。
max_instances	最大副本数	是	不配置默认为1。
hpa_scale_triggers	扩缩容指标配置，目前仅支持cpu/memory。	是	仅统计主容器资源。
hpa_scale_up_rules	扩容规则，定义稳定时间窗，减少扩容毛刺。	否	-
hpa_scale_down_rules	缩容规则，定义稳定时间窗，减少缩容毛刺。	否	-
hpa_scale_up_policies	扩容策略，定义扩容步长。	否	-
hpa_scale_down_policies	缩容策略，定义缩容步长。	否	-

示例：

在resources.yaml中添加hpa参数如下：

```
hpa:     #弹性伸缩配置
    $ref: 'config/hpa.yaml#/recommend'
```

在config目录，增加一个hpa.yaml文件，存放hpa的相关配置项。

```
recommend:
  disabled: true        #true表示关闭hpa，false表示开启hpa
  pollingInterval: 5    #负载检测周期，单位秒
  minReplicas: 2        #最小副本数
  maxReplicas: 4        #最大副本数
  triggers:             #业务根据时间情况选择弹性伸缩策略
```

```
  -type: CPU        #业务容器的CPU利用率大于40%则触发扩容条件
   metadata:
       averageUtilization: 40%
  -type: Memory                    #业务容器的内存利用率大于60%则触发扩容条件
   metadata:
      averageUtilization: 60%
```

指定分组和资源标签

表 2-24 指定分组和资源标签

参数名称	参数类型	是否必选	默认值	说明	参数规范	举例
resourceTag	type = object ({group = stringfeatures = string})	可选	无	• group：指定集群部署的分组。 • features：指定集群部署的标签信息。	此处配置的信息要在Runtime页面提前预置好。	resourceTag: group: "common" features: "dev"

2.9.1.2 配置 demo

```
# resources.yaml
- name: sdkCluster
  type: WiseCloud::MicroService::NuwaContainer
  properties:
    microserviceName: WiseCloudNuwaSDK    # STS证书中服务的名称
    replicas:
      $ref: 'config/values.yaml#/values/replicas'
    terminationGracePeriodSeconds: 30
    network:
      slbEnable:
        $ref: 'config/values.yaml#/values/slbEnable'
      isolatedDomain: XXX
    volumes:
      - name: log_volume
        size: 10Gi
        type: local
      - name: data_volume
        size: 2Gi
        type: local
    containers:                          # 微服务基于容器化的部署
      - image:
          $ref: 'config/values.yaml#/values/image'
        flavor: 2C4G
        stsEnable: true
        ports:
          - 8080
        livenessProbe:
          httpGet:
            path: /health.html
            port: 8080
            scheme: HTTP
          initialDelaySeconds: 20
          timeoutSeconds: 3
```

```
        periodSeconds: 10
        successThreshold: 1
        failureThreshold: 3
      readinessProbe:
        httpGet:
          path: /health.html
          port: 8080
          scheme: HTTP
        initialDelaySeconds: 20
        timeoutSeconds: 3
        periodSeconds: 10
        successThreshold: 1
        failureThreshold: 10
      preStopConfig:
        execCommand: ["/bin/bash", "-c", "sleep 20"]
      envs:
        $ref: 'config/envs.yaml#'
      hostAliases:
        - hostName: "a.b.com"
          ip: "1:1:1:1:1:1:1:1"
        - hostName: "c.d.com"
          ip: "2.2.2.2"
  certConfigs:                    # 可选
        - name: console
          tag: default
  configs:
    $ref: 'config/business_config.yaml#'
  daemonSet:
    - name: AIOps
      enable: true
      logPath: '/opt/huawei/logs'
      limitSize: 100G
      groups: ["logConfigGroupName"]
    - name: BI
      enable: false
      logPath: '/opt/huawei/logs/bi'
      groups: ["ODS_V001_DM_service1", "ODS_V001_DM_service2"]
  sidecars:
    - name: AIOpsLog
      version: 1.11.3.100
      flavor: 0.4C500M
      param:
        $ref: 'config/sidecar_aiops_param.json'
     - name: RASP
      version: 2.2.0.102.SP5
      flavor: 0.3C500M
  bindSlb:
    - port: 8080
      grayStatus: 1
      weight: 20
      maxFails: 3
      timeout: 20
      instanceName: chaos_slb_listener_lhq
      listenerGroupName: chaos_slb_listener_lhq
      targetGroup:
        name: chaosmonekey_portal_static
```

```
# business_config.yaml
public:
  prefix: /com.huawei.wiseeye
  name: public5
private:
  schema:
    type: object
    properties:
      AIOPSCONFIG_AIOPSSERVER:
        format: notype
      AIOPSCONFIG_AIOPSTOKENURI:
        format: sensitive
```

```
records:
  AIOPSCONFIG_AIOPSSERVER: 'https://XX.XX.XX.XX:XXXX/'
  AIOPSCONFIG_AIOPSTOKENURI: openapi/XXX/XXX/
```

2.9.1.3 错误码说明

表 2-25 错误码说明

错误码	说明
NotFound	实例不存在
NuwaRuntime.Microservice.CreateError	实例创建失败
NuwaRuntime.Microservice.ReadError	实例读取失败
NuwaRuntime.Microservice.DeleteError	实例删除错误
NuwaRuntime.Microservice.UpdateError	实例更新错误
NuwaRuntime.Microservice.UnknownError	未知异常

2.9.2 配置管理

本章介绍如何通过IaC代码描述配置信息，支持私有配置和公共配置两种类型。

IaC3.0公共配置集模型与私有配置项绝大部分字段参数一样，具体请参见表2-26。

表 2-26 配置管理字段说明

参数	是否必选	类型	说明	格式校验
name	是	string	配置集名称 • 私有配置集：服务环境下name唯一。 • 公共配置集：产品环境下name唯一。	建议：^[A-Za-z0-9_]{1,64}
type	是	string	资源类型 • 私有配置集：WiseCloud::MicroService::PrivateConfig • 公共配置集：WiseCloud::MicroService::PublicConfig	-

参数	是否必选	类型	说明	格式校验
prefix	否	string	为虚机订阅坐标，虚机部署按prefix前缀来订阅配置，容器场景下不填。 • 私有配置样例：/app/cloudeye/WiseEyeConfig/v1/rnd • 公共配置样例：/public/cloudeye/test	• 私有配置：/app/${scope}/${module}/${version}/${tag} • 公共配置：/public/${scope}/${tag}
version	私有配置必填 公共配置无此字段	string	配置版本	满足正则：^[A-Za-z0-9_.:()-]{1,100}
schema	否	object	配置项属性，描述配置项类型和作用。 properties属性为key-object格式，key是配置名称，object是配置项各项描述。	N/A description：最大长度256。 format当前支持的类型如表2-27所示。
records	是	object	描述配置，key:value格式，key为配置名称，value为配置值。	records数组内： • key：满足正则^[A-Za-z0-9_.:-]{1,100}$ • value：最大长度5000

format为schema下的一个字段，对配置进行说明，不同的配置项在使用方式上有所不同。

表 2-27 format 说明

类型	中文	说明
notype	未分类	不提供时配置项默认为未分类的类型。
sensitive	敏感配置项	标记此配置项为敏感配置项，此类配置项存储的是配置在STS的路径，非配置密文。
env	环境	-
business	调优	-

类型	中文	说明
thirdparty	对接三方配置	-
system	业务系统配置项	-
middleware	中间件配置	-
internal	对接内部服务配置	-
function	服务功能调优配置	-
oap	OAP	-

私有配置样例：

```
name: my_config
    type: WiseCloud::MicroService::PrivateConfig
    properties:
      prefix: /app/cloudeye/WiseeyeConfig/v1/rnd
      version: v1
      schema:
        type: object
        properties:
          keyl:
            description:
            type: string
            format: notype
          key2:
            description:
            type: string
            format: sensitive
      records:
        key1: value1
        key2: value2
```

公共配置样例：

```
name: test_config_lhq
    type: WiseCloud::MicroService::PublicConfig
    properties:
      prefix: /public/cloudeye/wiseEyeConfigService
      schema:
        type: object
        properties:
          key_lhq1:
            description: KEY1相关描述
            type: string
            format: notype
          key_lhq2:
            description: KEY2相关描述
            type: string
            format: notype
      records:
        key_lhq1: value_lhq1
        key_lhq2: value_lhq2
```

2.9.3 SLB

2.9.3.1 SLB 资源概述

资源介绍

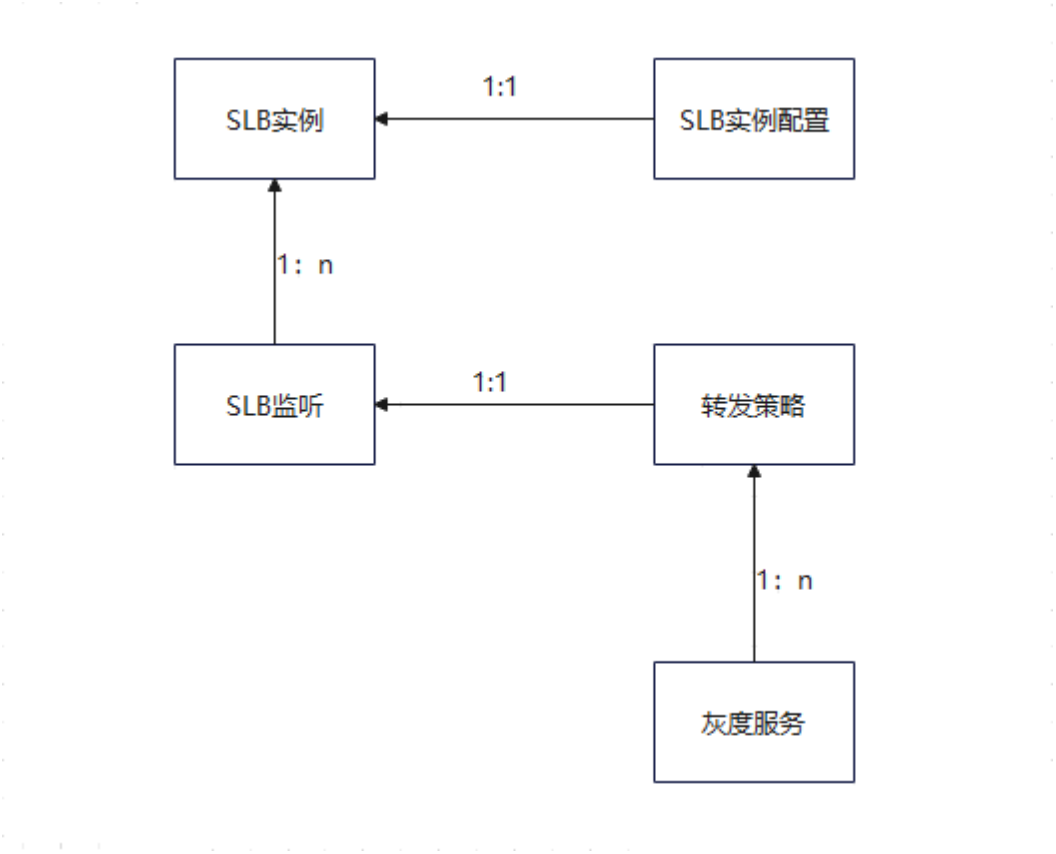
SLB当前提供了以下五种资源类型，来支持相关的配置。

表 2-28 SLB 资源类型

资源类型	归属部署服务	资源内容
WiseCloud::LoadBalancer::SLB	中间件	SLB实例的管理，包括SLB的扩容、部署及升级。
WiseCloud::LoadBalancer::SLB::Config	中间件	SLB实例配置的管理，包括nginx.conf、url重写/重定向、内网段、降级、黑白名单、流控及其他配置。
WiseCloud::LoadBalancer::SLB::ListenerGroup	中间件	SLB监听配置的管理，包括监听的域名、端口、协议、证书、监听级别的高级配置。
WiseCloud::LoadBalancer::SLB::RouteRule	中间件	转发策略配置的管理，包括监听下的转发策略、健康检查、后端服务器及动态路由。
WiseCloud::LoadBalancer::GrayConfig	一方服务	灰度服务配置的管理，包括灰度规则的管理及灰度阶段切换。

资源间的关系

图 2-6 资源间的关系



所有的资源必须归属于同一个服务下，才能绑定。

- SLB实例配置依赖SLB实例，比例关系为1:1。
- SLB监听依赖SLB实例，1个SLB实例可以对应多个SLB监听。
- 转发策略依赖SLB监听，比例关系为1:1。
- 灰度服务依赖转发策略，一个转发策略实例可以对应多个灰度服务，1个灰度服务只能对应一个转发策略实例。

2.9.3.2 SLB 实例

本章介绍通过IaC变更SLB实例，包括SLB的扩容、部署以及升级，对应的资源类型为WiseCloud::LoadBalancer::SLB。

表 2-29 SLB 实例字段说明

参数名	类型	是否必选	说明
version	String	是	SLB部署版本，格式为1.4.15，不带后缀，请和SLB管理台所选版本号区分。

参数名	类型	是否必选	说明
flavor	String	是	规格：2C8G、4C16G、8C32G、16C64G
replicas	int	是	单AZ主机数量，<=100 为0时创建自管理实例，大于0时创建平台管理实例。
isolatedDomain	String	否	隔离域，长度<=128，提前在运维中心ENS中规划。

申请SLB机器需要业务提前规划好隔离域，否则会申请失败。

样例：

```
- name: fgc-slb                                  #必传，SLB实例名称，长度<=50，不能包含-in-字符，不能以in-开头，不能以.conf结尾,不能包含特殊字符
  type: WiseCloud::LoadBalancer::SLB             #资源类型为SLB实例
  properties:
    version: 1.4.15                              #必传，SLB版本
    flavor: 2C8G                                 #必传，主机规格
    replicas: 5                                  #必传，主机数量
    isolatedDomain: xxxxxxx                      #非必传，隔离域，提前在运维中心ENS中规划
```

2.9.3.3 SLB 实例配置

本章介绍通过IaC进行SLB实例配置的管理，包括nginx.conf，url重写/重定向，内网段，降级，黑白名单，流控，其他配置以及自定义lua配置，对应的资源类型为WiseCloud::LoadBalancer::SLB::Config。

SLB 实例配置

表 2-30 SLB 实例配置字段说明

参数名	是否必选	说明
instanceName	是	SLB实例名，长度<=50
globalConf	是	Nginx配置，长度<=16777215
urlResetConf	否	url重写重定向
innerSegmentsConf	是	内网段配置
degradeConf	否	降级配置
blackListConf	否	黑名单配置
vipListConf	否	白名单配置
flowControlConf	否	流控配置

参数名	是否必选	说明
confLuaConf	否	其他配置，不填采用默认值
customLuaConf	否	自定义lua配置，每个lua文件大小不得超过16KB。

样例：

```
- name: fgc-slb-config                          #必传，SLB配置实例名
  type: WiseCloud::LoadBalancer::SLB::Config       #资源类型为实例配置
  properties:
    instanceName: fgc-slb                        #必传，SLB实例名
    globalConf:                                 #必传，nginx配置
      $ref: 'slb_instance_config/nginx.conf'
    urlResetConf:                               #非必传，url重写重定向配置
      $ref: 'slb_instance_config/url_reset_config.yaml#'
    innerSegmentsConf:                           #必传，内网段配置
      $ref: 'slb_instance_config/inner_segments_config.yaml#'
    degradeConf:                                #非必传，降级配置
      $ref: 'slb_instance_config/degrade_config.yaml#'
    blackListConf:                              #非必传，黑名单配置
      $ref: 'slb_instance_config/blacklist_config.yaml#'
    vipListConf:                                #非必传，白名单配置
      $ref: 'slb_instance_config/viplist_config.yaml#'
    flowControlConf:                             #非必传，流控配置
      $ref: 'slb_instance_config/flow_control_config.yaml#'
    confLuaConf:                                #非必传，其他配置
      $ref: 'slb_instance_config/conf_lua_config.yaml#'
    customLuaConf:                               #非必传，自定义lua配置
      slb100GlobalInit:
        $ref: 'slb_instance_config/SLB_100_Global_Init_iac3.lua'
      slb200WorkerInit:
        $ref: 'slb_instance_config/SLB_200_Worker_Init_iac3.lua'
      slb300PreFlowControl:
        $ref: 'slb_instance_config/SLB_300_Pre_FlowControl_iac3.lua'
      slb400OnFlowControled:
        $ref: 'slb_instance_config/SLB_400_On_FlowControled_iac3.lua'
      slb500PreGrey:
        $ref: 'slb_instance_config/SLB_500_Pre_Grey_iac3.lua'
      slb600PostGrey:
        $ref: 'slb_instance_config/SLB_600_Post_Grey_iac3.lua'
      slb700PostRoute:
        $ref: 'slb_instance_config/SLB_700_Post_Route_iac3.lua'
      slb750RespHeaderFilter:
        $ref: 'slb_instance_config/SLB_750_Resp_Header_Filter_iac3.lua'
      slb800RespBodyFilter:
        $ref: 'slb_instance_config/SLB_800_Resp_Body_Filter_iac3.lua'
```

nginx 配置

slb_instance_config/nginx.conf #nginx默认配置

```
#user  slb slb;
worker_processes auto;
#worker_cpu_affinity 0001 0010 0100 1000;
pid        logs/nginx.pid;
#############################################################################
### Default: Close the error log
error_log  /dev/null crit;
# nofile per worker around 20000-100000 is ok, eg, if have 8 worker, nginx will use no more than
8*worker_rlimit_nofile nofile, should make this result less than system nofile.
worker_rlimit_nofile 51200;
events {
```

```
    use epoll;
    # connections per worker, usually setup same or similar value as worker_rlimit_nofile.
    worker_connections 51200;
}
http {
    ##################################################################################
    ### load basic lua script
    include 'lua/nginx.http.lua.conf';
    init_by_lua_file 'conf/lua/initial.lua';
    init_worker_by_lua_file 'conf/lua/initialWorker.lua';
    log_by_lua_file 'conf/lua/monitor/LogRequest.lua';
    #rewrite_by_lua_no_postpone on;
    ##################################################################################
    uninitialized_variable_warn off;
    server_tokens off;
    autoindex off;
    port_in_redirect off;
    ssi off;
    proxy_hide_header X-Powered-By;
    add_header X-XSS-Protection "1; mode=block";
    add_header X-frame-options SAMEORIGIN;
    add_header X-Content-Type-Options  nosniff;
    add_header Strict-Transport-Security "max-age=31536000; includeSubDomains ";
    add_header Content-Security-Policy "default-src 'self'";
    add_header Cache-control "no-cache, no-store, must-revalidate";
    add_header Pragma no-cache;
    add_header Expires 0;
    client_header_timeout 60s;
    client_body_timeout 60s;
    keepalive_timeout 75s;
    send_timeout 60s;
    client_header_buffer_size 1k;
    large_client_header_buffers 4 8k;
    client_body_buffer_size 16k;
    client_max_body_size 1m;
    proxy_buffer_size 8k;
    proxy_buffers 8 8k;
    proxy_busy_buffers_size 16k;
    include mime.types;
    default_type text/html;
    ##################################################################################
    ### gzip compress
    gzip on;
    gzip_http_version 1.1;
    gzip_comp_level 5;
    gzip_min_length 1k;
    gzip_disable "MSIE [1-6].";
    gzip_types
        text/plain text/css text/xml text/javascript
        application/json application/x-javascript application/xml application/xml+rss application/xhtml+xml;
    ##################################################################################
    ### enabled the error page  process
    fastcgi_intercept_errors on;
    error_page 400 401 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 /4xx.html;
    error_page 500 501 502 503 504 505 /5xx.html;
    ##################################################################################
    ### log format and switch.
    log_format  main  '$time_local|$request_time|$upstream_response_time|$uri'
        '|$status|$body_bytes_sent|$request_length|$bytes_sent|$http_user_agent|$http_host'
        '|$upstream_addr|$upstream_status|$scheme|$is_grey_server|$resp_status|$server_protocol';
    access_log logs/access.log  main buffer=5m flush=10s;
    log_format  bigData  '$time_local|$server_addr|$upstream_addr|$is_grey_server|$uri'
        '|$upstream_status|$resp_status|$request_time|$upstream_response_time|$request_length|$bytes_sent|
$connections_active|$target_all|$remote_addr'
        '|$http_x_forwarded_for|$request_method|$http_user_agent|$status|$http_referer|$server_protocol|
$body_bytes_sent|$request_length|$http_host|$request'
        '|$server_port|-|$http_x_api_method||||||||||';
    access_log logs/access_for_big_data.log  bigData buffer=5m flush=10s;
    ##################################################################################
```

```
    ### load sub configure
    include vhosts/*.conf;
    include slb_conf/*.conf;
}
```

重写重定向配置

表 2-31 urlResetConf 字段说明

参数名	类型	是否必选	说明
transferType	String	是	转换类型，为以下枚举值： ● rewriteGrey 灰度重写 ● rewriteNormal 生产重写 ● redirectGrey 灰度重定向 ● redirectNormal 生产重定向
source	String	是	匹配路径，例：/abc/portal/login.jsp(.*)$
target	String	是	目标路径，例：/abc/def/$1

slb_instance_config/url_reset_config.yaml #重写重定向配置

```
- transferType: rewriteNormal          #必填，转发类型，rewriteNormal代表生产重写
  source: /a1                          #必填，匹配路径
  target: /b1                          #必填，目标路径
- transferType: rewriteGrey            #必填，转发类型，rewriteGrey代表灰度重写
  source: /a2                          #必填，匹配路径
  target: /b2                          #必填，目标路径
```

内网段配置

表 2-32 innerSegmentsConf 字段说明

参数名	类型	是否必选	说明
segment	String	是	IP地址段，格式为ip/子网掩码。

slb_instance_config/inner_segments_config.yaml #内网段配置

```
- segment: 10.0.0.0/8
- segment: 127.0.0.1/32
- segment: 172.16.0.0/12
- segment: 192.168.0.0/16
- segment: 100.125.0.0/16
```

降级配置

表 2-33 degradeConf

参数名	类型	是否必选	说明
switchStatus	String	是	降级开关，取值为on或off。
degradeUrl	String	否	修改时必传，不传表示置空此项，取值<=500。
currentLevel	int	是	当前降级等级，取值为整数1到5。
defaultLevel	int	是	接口默认等级，取值为整数1到5。
ruleItems	List<DegradeRule>	当降级开关为on时必填 当降级开关为off时非必填	降级列表，每个等级只能有一个。

表 2-34 DegradeRule

参数名	类型	是否必选	说明
level	String	是	降级等级，取值为1-5之间。
values	int	是	降级规则，对应等级的匹配规则表达式，例如： path equal[/abc/def,abc/ghi] <=100000

slb_instance_config/degrade_config.yaml #降级配置

```
switchStatus: 'on'                    #必填，降级开关，取值为off或on
defaultLevel: 4                       #必填，接口默认等级，取值为整数1到5
currentLevel: 3                       #必填，当前降级等级，取值为整数1到5
degradeUrl: @xxx                        #非必填，降级url
ruleItems:                          #降级规则信息
  - level: 1                        #必填，等级
    values: head[appid] equal[1,2,3]        #必填，降级规则
  - level: 2
    values: head[appid] equal[4,5,6]
```

黑白名单配置

表 2-35 blackListConf/vipListConf 字段说明

参数名	类型	是否必选	说明
switchStatus	String	是	开关，取值为on或off。

参数名	类型	是否必选	说明
groups	List<**BlackVipListGroup**>	是	名单列表
degradeUrl	String	否	定制响应location，以@olc_degrade开头，当不传递时，表示置空该字段。

表 2-36 BlackVipListGroup

参数名	类型	是否必选	说明
ruleItems	List<**BlackVipListRule**>	是	规则列表

表 2-37 BlackVipListRule

参数名	类型	是否必选	说明
type	String	是	匹配项类型，取值为：path、left-ip、right-ip、custom。
param	String	否	参数名，长度<=50 Type为custom时必传，其他type无需传递。
position	String	否	位置 • Type为custom时必传，为以下枚举值：queryString、header、resource、body-json、body-form。 • 当type为path、left-ip、right-ip时无需传递。
match	String	是	匹配条件 • 当type为left-ip或right-ip时，仅支持：sha256和rangeIP。 • 当type为custom或path时，仅支持：equal、sha256、pattern。

参数名	类型	是否必选	说明
values	String	是	对应匹配的值 ● match为equal时，例如填写1,2,3，表示1或2或3都可匹配。 ● 当match为rangeIP时，填写样例：10.1.1.1-10.1.1.100,10.2.2.1-10.2.2.100 ● match为sha256时，填写样例：5994471abb01112afcc18159f6cc74b4f511b99806da59b3caf5a9c173cacfc5,226fe7d26af82de76db083e6a29524ca68f4aaf80f2c2db058571fdb8f1fdeea ● match为pattern时，填写样例：^abc <=102400

slb_instance_config/blacklist_config.yaml #黑名单配置

```
switchStatus: 'on'                    #必填，黑名单开关，取值为off或on
degradeUrl: @xxx                      #非必填，降级url
groups:                               #必填，黑名单组
  - ruleItems:                        #必填，组内规则
    - type: custom                    #必填，类型
      param: x-app-id                 #非必填，参数
      position: header                #非必填，位置
      match: equal                    #必填，匹配条件
      values: '[1,2,3]'               #必填，值
```

slb_instance_config/viplist_config.yaml #白名单配置

```
switchStatus: 'on'                    #必填，白名单开关，取值为off或on
groups:                               #必填，白名单列表
  - ruleItems:                        #必填，组内规则
    - type: custom                    #必填，类型
      param: x-app-id                 #非必填，参数
      position: header                #非必填，位置
      match: equal                    #必填，匹配条件
      values: '[1,2,3]'               #必填，值
```

流控配置

表 2-38 flowControlConf 字段说明

参数名	类型	是否必选	说明
flowControlSwitch	String	是	流控总开关，取值为on或off。
autoDivideSwitch	String	是	分摊模式开关，取值为on或off。
flowControlOrder	String	是	流控类型执行顺序，和下面的流控配置相匹配。
nodeFlowControl	**FlowControlTypeBean**	否	节点级流控配置
interfaceFlowControl	**FlowControlTypeBean**	否	接口级流控配置
serviceFlowControl	**FlowControlTypeBean**	否	服务级流控配置
ipFlowControl	**FlowControlTypeBean**	否	IP流控配置
singleParamFlowControl	**FlowControlTypeBean**	否	自定义参数流控配置
multiParamFlowControl	**FlowControlTypeBean**	否	多参数组合流控配置
quotaFlowControl	**FlowControlTypeBean**	否	配额流控配置
concurrentFlowControl	**FlowControlTypeBean**	否	并发连接流控配置

表 2-39 FlowControlTypeBean

参数名	类型	是否必选	说明
switchStatus	String	是	流控开关，取值为on或off

参数名	类型	是否必选	说明
degradeUrl	String	是	降级URL，长度<500
limit	int	是	每秒请求量
burst	int	是	突发请求量
limitTag	String	否	自定义标签，长度<=200
ruleItems	List<**FlowControlRule**>	否	匹配规则列表
configGroups	List<**FlowControlGroup**>	否	匹配规则组列表

表 2-40 FlowControlRule

参数名	类型	是否必选	说明
type	String	否	匹配项类型，为以下枚举：path、left-ip、right-ip、custom。
param	String	否	参数名，长度<50。 type为custom时必传，其他type无需传递。
position	String	否	位置 • type为custom时必传，以下枚举值：queryString、header、resource、body-json、body-form、url。 • 其他type无需传递。
match	String	是	参数匹配方式 • 当type为left-ip或right-ip时，仅支持：sha256和rangeIP。 • 当type为custom或path时，仅支持：equal、sha256、pattern。

参数名	类型	是否必选	说明
values	String	是	对应匹配的值 ● match为equal时，例如填写1,2,3，表示1或2或3都可匹配。 ● 当match为rangeIP时，填写样例：10.1.1.1-10.1.1.100,10.2.2.1-10.2.2.100 ● match为sha256时，填写样例：5994471abb01112afcc18159f6cc74b4f511b99806da59b3caf5a9c173cacfc5,226fe7d26af82de76db083e6a29524ca68f4aaf80f2c2db058571fdb8f1fdeea ● match为pattern时，填写样例：^abc <=102400
isAgg	String	否	是否聚合，0不聚合 1 聚合，默认不聚合
limit	int	否	limit大小 接口级流控，ip流控，服务级流控，单参数流控必填，其他类型流控无效
burst	int	否	突发量 接口级流控，ip流控，服务级流控，单参数流控必填，其他类型流控无效
limitTag	String	否	自定义标签 <=200

表 2-41 FlowControlGroup

参数名	类型	是否必选	说明
timeWindow	int	否	配额周期，正整数。 timeWindow配合timeUnit，最多不超过86400秒，或不超过1440分钟，或不超过24小时，或不超过1天。
timeUnit	String	否	时间单位，取值为second、minute、hour、day。
limit	int	是	限流数量。
burst	int	否	突发量。
ruleItems	List< FlowControlRuleBean >	是	匹配规则列表。

slb_instance_config/flow_control_config.yaml #流控配置

```
flowControlSwitch: 'on'                          #必填，流控总开关，取值为off或on
autoDivideSwitch: 'off'                          #必填，分摊模式开关，取值为off或on
flowControlOrder: nodeFlowControl,urlFlowControl   #必填，流控类型顺序；和下面的流控配置相匹配
nodeFlowControl:                                 #非必填，节点级流控
  switchStatus: 'on'                            #必填，节点级流控开关，取值为off或on
  degradeUrl: xxx                               #非必填，限流降级url
  limit: 1000                                  #必填，流控值
  burst: 1000                                   #必填，突发量
  limitTag: xxxx                                #非必填，自定义标签
serviceFlowControl:                              #非必填，服务级流控
  switchStatus: 'on'
  degradeUrl: xxx
  limit: 1000
  burst: 1000
  ruleItems:                                   #必填，流控规则
    - match: equal                               #必填，匹配条件；取值有equal，pattern
      values: example.com                          #必填，域名值
      limit: 1000                               #必填，流控值
      burst: 1000                                #必填，突发量
      limitTag: xxxx                             #非必填，自定义标签
interfaceFlowControl:                            #非必填，接口级流控
  switchStatus: 'on'
  degradeUrl: xxx
  limit: 1000
  burst: 1000
  ruleItems:
    - match: equal                               #必填，匹配条件；取值有equal，pattern
      values: xxx                               #必填，接口url
      limit: 1000                               #必填，流控值
      burst: 1000                                #必填，突发量
      limitTag: xxxx                             #非必填，自定义标签
      isAgg: off                                #非必填，聚合统计开关
ipFlowControl:                                   #非必填，IP流控
  switchStatus: 'on'
  degradeUrl: xxx
  limit: 1000
  burst: 1000
  ruleItems:
    - type: left-ip                            #必填，类型
      match: rangeIP                             #必填，匹配条件
```

```
    values: 1.1.1.1-1.1.1.2          #必填，值
    limit: 1000                      #必填，流控值
    burst: 1000                      #必填，突发量
    limitTag: xxxx                   #非必填，自定义标签
singleParamFlowControl:              #非必填，自定义参数流控
  switchStatus: 'on'
  degradeUrl: xxx
  ruleItems:
    - param: xxx                     #必填，参数
      position: url                  #必填，位置
      match: equal                   #必填，匹配条件
      values: 1                      #必填，值
      limit: 1000                    #必填，流控值
      burst: 1000                    #必填，突发量
      limitTag: xxxx                 #非必填，自定义标签
multiParamFlowControl:               #非必填，多参数组合流控
  switchStatus: 'on'
  degradeUrl: xxx
  configGroups:                      #必填，配置组
    - limit: 1000                    #必填，组流控值
      burst: 1000                    #必填，组突发量
      limitTag: xxxx                 #非必填，自定义标签
      ruleItems:                     #必填，流控规则
        - type: path                 #必填，类型
          param:                     #非必填，参数
          position:                  #非必填，位置
          match: equal               #必填，匹配条件
          values: /a                 #必填，值
          isAgg: 'off'               #非必填，聚合统计开关
        - type: custom
          param: zzz
          position: queryString
          match: equal
          values: 1
          isAgg: off
quotaFlowControl:                    #非必填，配额流控
  switchStatus: 'on'
  degradeUrl: @example.com
  configGroups:                      #必填，配置组
    - timeWindow: 10                 #必填，时间周期窗口
      timeUnit: minute               #必填，时间单位
      limit: 1000                    #必填，配额大小
      limitTag: xxxx                 #非必填，自定义标签
      ruleItems:                     #必填，流控规则
        - type: path                 #必填，类型
          param:                     #非必填，参数
          position:                  #非必填，位置
          match: equal               #必填，匹配条件
          values: /a                 #必填，值
          isAgg: 'off'               #非必填，聚合统计开关
        - type: custom
          param: zzz
          position: queryString
          match: equal
          values: 1
          isAgg: off
concurrentFlowControl:               #非必填，并发连接流控
  switchStatus: 'on'
  degradeUrl: xxx
  configGroups:                      #必填，配置组
    - limit: 1000                    #必填，流控值
      burst: 1000                    #必填，突发量
      limitTag: xxxx                 #非必填，自定义标签
      ruleItems:                     #必填，流控规则
        - type: path                 #必填，类型
          param:                     #非必填，参数
          position:                  #非必填，位置
          match: equal               #必填，匹配条件
          values: /a                 #必填，值
```

```
  - type: custom
    param: zzz
    position: queryString
    match: equal
    values: 1
```

其他配置

表 2-42 confLuaConf 字段说明

参数名	说明
addGreyFlag	灰度标记开关，取值为on或off。
greyTestSwitch	灰度测试开关，取值为on或off。
greyTestServiceId	灰度测试服务Id。 greyTestSwitch为on时必传。
getIpType	取IP方式，取值为1或2，1表示从左取，2表示从右取，默认为1。
isBypassOnGreyDown	灰度服务器全部宕机后，请求路由到生产开关，取值为on或off，默认为off。
areaGreyGetIpFromLeft	地域灰度IP从左侧取值开关，取值为on或off，默认为off。
greyTestQpsLimit	灰度测试每秒转发量限制，取值为1到1000。
successRateAlarmAbsThreshold	成功率告警阈值绝对值，非负浮点数，取值为0到100，默认值为90。
successRateAlarmOffsetThreshold	成功率下降告警阈值（相比1分钟前或者5分钟前），非负浮点数，取值为0到100，默认值为5。
healthCheckAlarmServerCountThreshold	健康检查不健康机器数告警阈值，正整数，默认为1，表示有1台节点不健康就会告警。
healthCheckAlarmDurationThreshold	健康检查告警持续时间阈值，非负正整数，默认为0，表示不健康主机立即告警；如果配置为1，表示发现不健康持续1分钟以上，才会告警。
concurrentFlowControlAlarmThreshold	并发请求流控告警阈值，非负整数，默认为0，表示只要发生1次流控，就会告警。
serviceFlowControlAlarmThreshold	服务级流控告警阈值，非负整数，默认0，表示只要发生1次流控，就会告警。
singleParamFlowControlAlarmThreshold	自定义参数流控告警阈值，非负整数，默认0，表示只要发生1次流控，就会告警。
multiParamFlowControlAlarmThreshold	多参数流控告警阈值，非负整数，默认0，表示只要发生1次流控，就会告警。

参数名	说明
quotaFlowControlAlarmThreshold	配额流控告警阈值，非负整数，默认0，表示只要发生1次流控，就会告警。
manyRequestPreAlarmThreshold	请求数过多告警阈值，正整数，例如：25000，表示每个cpu每分钟平均处理达到25000请求，则触发告警。 以4C的主机为例，1分钟处理超过25000*4=100000请求，则开始告警。
manyRequestAlarmThreshold	请求数警阈值，非负整数，默认为0。
statisticsParams	统计日志参数，格式为格式为ParamName:position。 • ParamName为参数名称。 • Position为参数位置，取值为queryString或header。

slb_instance_config/conf_lua_config.yaml #其他配置

```
addGreyFlag: 'off'                                  #非必填，灰度标记开关，取值为off或on
greyTestSwitch: 'off'                               #非必填，灰度测试开关，取值为off或on
greyTestServiceId: 123456                           #非必填，灰度测试服务ID
isBypassOnGreyDown: 'off'                           #非必填，宕机时路由到生产集群开关，取值为off或on
areaGreyGetIpFromLeft: 'off'                        #非必填，地域灰度IP从左取值开关 ，取值为off或on
getIpType: 1                                        #非必填，取IP方式，取值为1或2
statisticsParams:                                   #非必填，统计日志参数
greyTestQpsLimit： 1000                             #非必填，灰度测试每秒转发量限制，取值为1到1000
manyRequestAlarmThreshold: 1000                     #非必填，请求数量告警
successRateAlarmAbsThreshold: 90                    #非必填，成功率告警阈值，取值为0到100
successRateAlarmOffsetThreshold: 90                 #非必填，成功率下降告警阈值，取值为0到100
healthCheckAlarmServerCountThreshold: 1000          #非必填，健康检查告警阈值
healthCheckAlarmDurationThreshold: 1000             #非必填，健康告警持续时间阈值
concurrentFlowControlAlarmThreshold: 1000           #非必填，并发连接流控告警阈值
serviceFlowControlAlarmThreshold: 1000              #非必填，服务级流控告警阈值
multiParamFlowControlAlarmThreshold: 1000           #非必填，多参数流控告警阈值
singleParamFlowControlAlarmThreshold: 1000          #非必填，自定义参数流控告警阈值
quotaFlowControlAlarmThreshold: 1000                #非必填，配额流控告警阈值
manyRequestPreAlarmThreshold: 1000                  #非必填，请求数量预告警
```

自定义 lua 配置

表 2-43 customLuaConf 字段说明

参数名	类型	是否必选	说明
slb100GlobalInit	String	否	大小不得超过16KB
slb200WorkerInit	String	否	大小不得超过16KB
slb300PreFlowControl	String	否	大小不得超过16KB
slb400OnFlowControled	String	否	大小不得超过16KB
slb500PreGrey	Int	否	大小不得超过16KB

参数名	类型	是否必选	说明
slb600PostGrey	String	否	大小不得超过16KB
slb700PostRoute	String	否	大小不得超过16KB
slb750RespHeaderFilter	int	否	大小不得超过16KB
slb800RespBodyFilter	float	否	大小不得超过16KB

slb_instance_config/SLB_100_Global_Init_iac3.lua #SLB_100_Global_Init_iac3.lua默认配置

```
--the custom point SLB_100_Global_Init, running when Global init.
--will be triggered in exec function, so pls make sure there is a exec function in global_init
local global_init = {};
function global_init.exec(gen_param)
    --eg:ngx.log(ngx.ERR,"i am in global_init")
end
return global_init
```

slb_instance_config/SLB_200_Worker_Init_iac3.lua #SLB_200_Worker_Init_iac3.lua默认配置

```
--the custom point SLB_200_Worker_Init, running when Worker init.
--will be triggered in exec function, so pls make sure there is a exec function in worker_init
local worker_init = {};
function worker_init.exec(gen_param)
    --eg:ngx.log(ngx.ERR,"i am in worker init");
end
return worker_init
```

slb_instance_config/SLB_300_Pre_FlowControl_iac3.lua #SLB_300_Pre_FlowControl_iac3.lua默认配置

```
--the custom point SLB_300_Pre_FlowControl, running before flowcontrol and blacklist.
--will be triggered in exec function, so pls make sure there is a exec function in pre_flowcontrol
local pre_flowcontrol = {};
function pre_flowcontrol.exec(gen_param)
    --eg:ngx.log(ngx.ERR,"I am in pre flowcontrol")
end
return pre_flowcontrol
```

slb_instance_config/SLB_400_On_FlowControled_iac3.lua #SLB_400_On_FlowControled_iac3.lua默认配置

```
--the custom point SLB_400_On_FlowControlled, running when request is flowcontrolled or blocked due to
blacklist.
--will be triggered in exec function, so pls make sure there is a exec function in on_flowcontroled
local on_flowcontroled = {};
function on_flowcontroled.exec(gen_param)
    --eg:ngx.log(ngx.ERR,"the flowcontrol type is:",gen_param.control_type)
end
return on_flowcontroled
```

slb_instance_config/SLB_500_Pre_Grey_iac3.lua #SLB_500_Pre_Grey_iac3.lua默认配置

```
--the custom point SLB_500_Pre_Grey, running before Greyrule judge.
--will be triggered in exec function, so pls make sure there is a exec function in pre_grey
local pre_grey = {};
```

```
function pre_grey.exec(gen_param)
    --eg:ngx.log(ngx.ERR,"I am in pre grey");
end
return pre_grey
```

slb_instance_config/SLB_600_Post_Grey_iac3.lua #SLB_600_Post_Grey_iac3.lua默认配置

```
--the custom point SLB_600_Post_Grey, running after get result of greyrule.
--will be triggered in exec function, so pls make sure there is a exec function in post_grey
local post_grey = {};
function post_grey.exec(gen_param)
    --eg:ngx.log(ngx.ERR,"I AM IN POST GREY");
end
return post_grey
```

slb_instance_config/SLB_700_Post_Route_iac3.lua #SLB_700_Post_Route_iac3.lua默认配置

```
--the custom point SLB_700_Post_Route, running after get result of routerule.
--will be triggered in exec function, so pls make sure there is a exec function in post_route
local post_route = {};
function post_route.exec(gen_param)
    --eg:ngx.log(ngx.ERR,"I AM IN POST Route");
end
return post_route
```

slb_instance_config/SLB_750_Resp_Header_Filter_iac3.lua #SLB_750_Resp_Header_Filter_iac3.lua默认配置

```
--the custom point SLB_750_Resp_Header_Filter, running during header filter.
--will be triggered in exec function, so pls make sure there is a exec function in header_filter
local header_filter = {};
function header_filter.exec(gen_param)
    --eg:ngx.log(ngx.ERR,"I am in header filter");
end
return header_filter
```

slb_instance_config/SLB_800_Resp_Body_Filter_iac3.lua #SLB_800_Resp_Body_Filter_iac3.lua默认配置

```
--the custom point SLB_800_Resp_Body_Filter, running during body filter.
--will be triggered in exec function, so pls make sure there is a exec function in body_filter
local body_filter = {};
function body_filter.exec(gen_param)
    --eg:ngx.log(ngx.ERR,"I am in body filter");
end
return body_filter
```

2.9.3.4 SLB 监听配置

本章介绍通过IaC进行SLB监听配置的管理，包括监听的域名、端口、协议、证书、监听级别的高级配置，对应的资源类型为WiseCloud::LoadBalancer::SLB::ListenerGroup。

表 2-44 SLB 监听配置字段说明

参数名	类型	是否必选	说明
instanceName	String	是	关联的slb实例名，仅能关联同一注册中心下的slb实例。

参数名	类型	是否必选	说明
domain	String	否	域名，多个域名不可重复，只允许数字、字母、下划线、“.”和“*”。
certificate	String	否	证书名，需为领域下已录入证书。
listeners	List<**PortConfig**>	是	监听端口列表。
config	GereralAndAdvancedConfigBean	否	通用及高级配置。

表 2-45 PortConfig

参数名	类型	是否必选	说明
port	String	是	监听端口，整型字符串，1-65535。
protocol	String	是	协议，取值为：http、https、http2、http2(ssl)。
otherConfig	String	否	其他配置。
elbReferences	List<**ElbPoolConfig**>	• 当所关联slb实例为平台管理模式时必传 • 当所关联slb实例为自管理模式时非必传	Elb关联列表。

表 2-46 ElbPoolConfig

参数名	类型	是否必选	说明
name	String	否	Elb名称，Elb名称可在ENS管理台查询。
pool	String	否	Elb后端集群。

样例：

```
- name: testListenerName                          #必传，slb监听名称，长度<=50，不能包含-in-字符，不能以in-开头，不能以.conf结尾,不能包含特殊字符
  type: WiseCloud::LoadBalancer::SLB::ListenerGroup  #资源类型为监听配置
  properties:
    instanceName: testSlbInstanceName              #必传，关联的slb实例名，仅能关联同一注册中心下的slb实例
    domain: www.test1.com,www.test2.com            #非必传，域名
    certificate: testCertificateName               #非必传，证书，需为领域下已录入证书
    listeners:                                     #必传
      - protocol: https                            #必传，协议，取值为：http、https、http2、http2(ssl)
        port: '9095'                               #必传，整型字符串，监听端口,1-65535
        elbReferences:                             #Elb关联列表
          - name: elbName1                         #Elb名称
            pool: elbPool1                         #Elb后端集群
          - name: elbName2
            pool: elbPool2
        otherConfig:                               #非必传
      - protocol: https
        port: '9096'
        elbReferences:
          - name: elbName3
            pool: elbPool3
          - name: elbName4
            pool: elbPool4
    config:                                        #非必传，监听通用配置和高级配置
      $ref: 'slb_listener_config/listener_monitor_config.yaml#'
```

配置监听

表 2-47 SLB 监听配置 config 字段说明

参数名	类型	是否必选	说明
proxyReadTimeout	int	否	响应超时时间，不超过10位的数字。
proxySendTimeout	int	否	转发超时时间，不超过10位的数字。
keepaliveSwitch	String	否	启用长连接转发，是否keepalive，取值为on或off。
hostSwitch	String	否	是否透传请求头，取值为on或off，不填或者其他字符均为off。
xForwardedForSwitch	String	否	是否追加XFF，取值为on或off。
accessLogSwitch	String	否	是否开启access日志，取值为on或off，不填或者其他字符均为off。
proxyNextUpstreamSwitch	String	否	是否在异常时尝试下一台，取值为on或off，不填或者其他字符均为off。
proxyNextUpstreamCondition	String	否	尝试下一台场景，"403"、"404"、"429"、"500"、"502"、"503"、"504"、"error"、"timeout"、"invalid_header"。

参数名	类型	是否必选	说明
clientConnectionFreeTime	int	否	客户端连接空闲时间，长度不超过10位的正整数。
proxyBuffersSwitch	String	否	开启响应缓存，取值为on或off，不填或者其他字符均为off。
proxyBuffersSize	String	否	响应缓存大小，由数字+空格+k/m组成，默认为8 k。
expiresSwitch	String	否	高级配置中Expires开关，取值为on或off，不填或者其他字符均为off。
expiresSize	String	否	设定页面缓存时间，不超过50位，不缓存或一直使用缓存。可以由字母、数字、空格、$、@、+、-、冒号、逗号组成。
indexPage	String	否	静态页面场景index页面设置，长度不超过255，非中文。
returnVal	String	否	固定返回响应码，长度不超过255，非中文。
allowMethod	String	否	允许的http方法，对于转发策略有效，对于监听仅为参考，在界面新增转发策略时会继承监听的此项配置，其余场景无效，"GET", "HEAD", "POST", "DELETE", "PUT", "OPTIONS", "PATCH", "MKCOL", "COPY", "MOVE", "PROPFIND", "PROPPATCH", "LOCK", "UNLOCK"。
command	String	否	nginx配置命令，非中文。
root	String	否	默认资源根目录，长度不超过500，非中文。
clientMaxBodySize	String	否	最大请求体大小，数字+空格+k/m组成。
clientBodyBufferSize	String	否	请求体buffer大小，数字+空格+k/m组成。
addHeaders	List<**GeneralKeyValueConfig**>	否	响应头
setVals	List<**GeneralKeyValueConfig**>	否	对应页面set
proxySetHeaders	List<**GeneralKeyValueConfig**>	否	请求头

参数名	类型	是否必选	说明
redirects	List<GeneralRedirectConfig>	否	重定向
rewrites	List<GeneralRedirectConfig>	否	重写

表 2-48 GeneralKeyValueConfig

参数名	类型	是否必选	说明
key	String	否	键，非中文
value	String	否	值，非中文

表 2-49 GeneralRedirectConfig

参数名	类型	是否必选	说明
source	String	否	匹配规则，非中文。
target	String	否	重写/重定向目标，非中文。
option	String	否	重写标记，取值为："last"、"break"、"permanent"。

slb_listener_config/listener_monitor_config.yaml

```
hostSwitch: 'on'                    #是否透传请求头,on/off,不填或者其他字符均为off
    root:                          #默认资源根目录，不超过500非中文
    clientConnectionFreeTime: 100   #客户端连接空闲时间，不超过10位的正整数
    proxyReadTimeout: 70            #响应超时时间，不超过10位的数字
    keepaliveSwitch: 'on'          #启用长连接转发是否keepalive，on/off
    proxyBuffersSize: '2 m'         #响应缓存大小，数字+空格+k/m组成，默认为8 k
    expiresSwitch: 'on'            #高级配置中Expires开关，on/off，不填或者其他字符均为off
    proxyBuffersSwitch: 'on'        #开启响应缓存，on/off，不填或者其他字符均为off
    accessLogSwitch: 'on'          #是否开启access日志，on/off，不填或者其他字符均为off
    returnVal: 200                 #固定返回响应码，不超过255非中文
    allowMethod: POST,GET,HEAD,PUT,DELETE,OPTIONS    #允许的http方法
    nuwaTraceSwitch: 'off'
    command:                        #nginx配置命令，非中文
    clientMaxBodySize:              #最大请求体大小，数字+空格+k/m组成
    clientBodyBufferSize: 10        #请求体buffer大小，数字+空格+k/m组成
    expiresSize: 10                #设定页面缓存时间
    indexPage:                     #静态页面场景index页面设置，不超过255非中文
    proxyNextUpstreamCondition: error,timeout      #尝试下一台场景
    proxySendTimeout: 80            #转发超时时间，不超过10位的数字
    xForwardedForSwitch: 'on'       #是否追加XFF，on/off
    proxyNextUpstreamSwitch: 'on'   #是否在异常时尝试下一台，on/off，不填或者其他字符均为off
    addHeaders:                     #响应头
      - value: v1
```

```
    key: k1
  - value: v2
    key: k2
rewrites:                #重写
  - source: s7
    option: break
    target: t1
setVals:                 #对应页面set
  - value: v3
    key: $k3
proxySetHeaders:              #请求头
  - value: v5
    key: k5
redirects:               #重定向
  - source: s1
    target: t1
```

2.9.3.5 转发策略配置

本章介绍通过IaC进行转发策略配置的管理，包括监听下的转发策略、健康检查、后端服务器及动态路由管理。对应的资源类型为WiseCloud::LoadBalancer::SLB::RouteRule。

表 2-50 转发策略配置字段说明

参数名	类型	是否必选	说明
listenerGroupName	String	是	SLB监听名称，一个转发策略实例只能绑定一个监听实例。
targetGroups	List<**TargetGroupConfigBean**>	否	后端集群
routes	List<**RouteRule**>	否	转发策略(Locations)
dynamicRoutes	List<**DynamicRouteRuleBean**>	否	动态路由

表 2-51 TargetGroupConfigBean

参数名	类型	是否必选	说明
name	String	是	后端集群名，不超过150位，不包含特殊字符。
protocol	String	是	协议，取值为：HTTP或HTTPS。
loadBalancer	**TargetGroupLoadBalancerBean**	是	后端集群负载均衡类型
healthCheck	**TargetGroupHealthCheckBean**	否	健康检查

表 2-52 TargetGroupLoadBalancerBean

参数名	类型	是否必选	说明
strategy	String	是	负载均衡算法，支持"roundRobin"、"protocolParam"、"IP_HASH"、"least_conn"。
slowStartDelay	int	否	预热时延，不超过30。
slowStartPeriod	int	否	预热周期，不超过30。
slowStartFactor	int	否	预热因子，默认为10，不小于1，不超过100。
hashParam	String	否	哈希转发字段名，Strategy类型为protocolParam时必填 长度不超过50，由字母加数字，下划线，横杠，点组成。
customParam	String	否	自定义参数，不包含by_lua。

默认取第一个后端集群的健康检查配置，如果为空，使用默认值。

表 2-53 TargetGroupHealthCheckBean

参数名	类型	是否必选	说明
switchStatus	String	否	是否开启健康检查，取值为on或off，不填或者其他字符均为off。
protocol	String	否	协议，默认HTTP，不可修改。
path	String	否	检查路径，长度不超过500，由字母、数字、横杠、下划线和点组成。
successCodes	String	否	成功码列表，100-600之间的数字，以逗号隔开。
healthyThreshold	int	否	健康阈值，取值范围2-10，默认为2。
interval	int	否	探测间隔时间，取值范围1-50，默认为5。
timeout	int	否	探测超时时间，取值范围1-300，默认为10。
unhealthyThreshold	int	否	不健康阈值，取值范围2-10，默认为3。

表 2-54 RouteRule

参数名	类型	是否必选	说明
location	String	否	转发策略地址，长度不超过500，非中文。
targetGroup	String	否	后端集群名，长度不超过150，由字母、数字、下划线、横杠和点组成。
grayServiceName	String	否	灰度规则
degradeUrl	String	是	降级url ● 不降级：unDegrade ● 默认降级：defaultDegrade 只有值为defaultDegrade时才为默认降级，其他值均认为是不降级。
configs	GeneralAndAdvancedConfigBean	否	通用与高级配置

表 2-55 DynamicRouteRuleBean

参数名	类型	是否必选	说明
name	String	是	名称，长度不超过50，由字母、数字、下划线、横杠和点组成，不能包含-in-且不能以-in结尾。
switchStatus	String	是	开关，取值为on或off。
defaultTargetGroup	String	是	默认后端集群
enableGrey	String	是	默认后端集群是否仍参考灰度进行分流，取值为on/off，默认为off，只有输入on才为打开状态。
rules	List<DynamicSubRuleBean>	否	动态路由规则
locations	List<String>	否	动态路由关联的转发策略列表。

表 2-56 DynamicSubRuleBean

参数名	类型	是否必选	说明
condition	String	是	名称。
targetGroup	String	是	后端集群。
enableGray	String	是	是否仍参考灰度进行分流，取值为on/off，默认为off，只有输入on才为打开状态。

表 2-57 GeneralAndAdvancedConfigBean

参数名	类型	是否必选	说明
proxyReadTimeout	int	否	响应超时时间，不超过10位的数字。
proxySendTimeout	int	否	转发超时时间，不超过10位的数字。
keepaliveSwitch	String	否	启用长连接转发，是否keepalive。
hostSwitch	String	否	是否透传请求头，取值为on/off，不填或者其他字符均为off。
xForwardedForSwitch	String	否	是否追加XFF。
accessLogSwitch	String	否	是否开启access日志，取值为on/off，不填或者其他字符均为off。
proxyNextUpstreamSwitch	String	否	是否在异常时尝试下一台，取值为on/off，不填或者其他字符均为off。
proxyNextUpstreamCondition	String	否	尝试下一台场景，取值为403、404、429、500、502、503、504、error、timeout、invalid_header。
clientConnectionFreeTime	int	否	客户端连接空闲时间，不超过10位的正整数。
proxyBuffersSwitch	String	否	开启响应缓存，取值为on/off，不填或者其他字符均为off。
proxyBuffersSize	String	否	响应缓存大小，数字+空格+k/m 组成，默认8 k。

参数名	类型	是否必选	说明
expiresSwitch	String	否	高级配置中Expires开关，取值为on/off，不填或者其他字符均为off。
expiresSize	String	否	设定页面缓存时间，不缓存或一直使用缓存。可以由字母、数字、空格、 $@+- 、冒号、逗号组成，长度不超过50位。
indexPage	String	否	静态页面场景index页面设置，长度不超过255，非中文。
returnVal	String	否	固定返回响应码，长度不超过255，非中文。
allowMethod	String	否	允许的http方法，对于转发策略有效，对于监听仅为参考，在界面新增转发策略时会继承监听的此项配置，其余场景个无效，GET、HEAD、POST、DELETE、PUT、OPTIONS、PATCH、MKCOL、COPY", "MOVE、PROPFIND、PROPPATCH、LOCK、UNLOCK。
nuwaTraceSwitch	String	否	消息头中透传nuwatrace相关的网元身份标识和时间戳。
command	String	否	高级配置，自定义参数设置。 不要直接填写转发到某个集群的名称，例如proxy_pass http://upstreamName，这样配置会使得动态加载配置失效。 nginx配置命令，非中文。
root	String	否	默认资源根目录，长度不超过500，非中文。
clientMaxBodySize	String	否	最大请求体大小，数字+空格+k/m组成。
clientBodyBufferSize	String	否	请求体buffer大小，数字+空格+k/m 组成。
addHeaders	List<GeneralKeyValueConfigBean>	否	响应头。
setVals	List<GeneralKeyValueConfigBean>	否	对应页面set。

参数名	类型	是否必选	说明
proxySetHeaders	List<GeneralKeyValueConfigBean>	否	请求头。
redirects	List<GeneralRedirectConfigBean>	否	重定向。
rewrites	List<GeneralRedirectConfigBean>	否	重写。

转发策略模板yaml

```
- name: hw_test                                  // 必填项，不可修改，不可重复
  type: WiseCloud::LoadBalancer::SLB::RouteRule  // 资源类型为转发策略配置
  properties:
    routes:                                     // 非必填
      - targetGroup: group1
        grayServiceName: grayServce_hw
        location: /abc_hw1
        degradeUrl: unDegrade                   // 必填项，unDegrade不降级，defaultDegrade默认降级
        configs:
          hostSwitch: 'off'
          addHeaders:
            - value: b
              key: aaa
          clientConnectionFreeTime: 60
          rewrites:
            - source: /test
              option: break
              target: /test_hw
          proxyReadTimeout: 60
          setVals:
            - value: bbb
              key: aaa
          keepaliveSwitch: 'off'
          proxyBuffersSize: 2 m
          expiresSwitch: 'on'
          proxyBuffersSwitch: 'on'
          accessLogSwitch: 'on'
          clientMaxBodySize: 3 k
          clientBodyBufferSize: 2 k
          expiresSize: 2 m
          proxyNextUpstreamCondition: 'error,timeout'
          proxySendTimeout: 60
          xForwardedForSwitch: 'off'
          proxyNextUpstreamSwitch: 'on'
          allowMethod: 'POST,PUT,DELETE'
    targetGroups:                               // 非必填
      - protocol: HTTP                          // 必填项
        loadBalancer:                           // 必填项，负载均衡类型
          strategy: roundRobin                  // 必填项，负载均衡策略
        healthCheck:                            // 非必填，健康检查
          path: /abc_test
          protocol: HTTP                        // 固定项
          successCodes: '200,302'
          healthyThreshold: '2'
          unhealthyThreshold: '3'
          switchStatus: 'on'                    // 默认关闭，on为打开，打开时，不传其他参数均使用默认值
```

```
      name: group1                        // 必填项，并且不可重复，长度不超过150，不包含特殊字符
    - protocol: HTTP
      loadBalancer:
        hashParam: $xxx
        strategy: protocolParam
        customParam: $xxx
      healthCheck:
        switchStatus: 'off'
      name: group2
  listenerGroupName: zwx_listener1            // 必填项，SLB监听名称，一个转发策略实例只能绑定一个监听实例
  dynamicRoutes:                              // 非必填
    - enableGrey: 'off'
      defaultTargetGroup: group1              // 必填并且存在于targetGroups中
      name: /abc_test1_hw
      locations:
        - location: /abc_hw1
      rules:
        - targetGroup: group1
          enableGray: 'off'
      switchStatus: 'on'
```

2.9.3.6 灰度服务配置

本章介绍通过IaC进行灰度服务配置的管理，包括灰度规则的管理及灰度阶段切换。对应的资源类型为WiseCloud::LoadBalancer::GrayConfig。

灰度比例转换为pod个数是通过向下取整法，例如：共6个pod，如果灰度比例配置为10%，灰度pod个数则为6 * 0.1 = 0.6，向下取整 = 0，但是必须保持1个的副本数，所以仍要置为1；如果灰度比例配置为20%，灰度pod个数则为6 * 0.2 = 1.2，向下取整 = 1。

使用限制

- 如果灰度集群（bindSlb.grayStatus配置为2）想要使用多阶段灰度升级，要求bindSlb.grayStatus在灰度期间保持配置为2。
- 首次部署时不要配置grayStage，因为要进行灰度升级首先要保证有一个正常运行的版本作为生产，而如果首次部署是满足不了这个条件的，所以要至少正确部署过一次之后才能进行灰度升级。
- 配置了grayInstances即代表要进入多阶段灰度升级，且百分比只能配置为1%~100%（字符串）；但是如果你第一阶段就直接配置为100%，则仍然执行普通升级模式。
- 灰度升级期间（包括进入灰度时），IaC配置中不允许修改grayStage.grayStatus、min_instances、max_instances和SLB相关配置，其他配置可以修改。
- 灰度升级过程中，如果grayInstances的比例达到了100%，grayProcess为INGRAY时，再进行比例缩小时，该场景只允许修改灰度比例。
- 灰度升级过程中，如果发现灰度版本配置错误，可以修改IaC配置将灰度集群进行重新部署，但是这时不允许修改grayInstances，即IaC配置和灰度比例只能修改其中一个。
- 多阶段灰度升级中，业务将grayInstances配置为100%（未配置grayProcess字段）或grayInstances配置为100%、grayProcess为FINISHED并部署成功后，才算灰度结束了；中间一个阶段部署成功了不算结束灰度，只算这一阶段部署成功了，此时仍然算在灰度升级过程中。
- 灰度过程中，不允许业务再执行普通部署，不允许在管理台上修改配置，只能通过IaC部署修改。

- 多阶段灰度升级和滚动升级同时使用时，maxUnavailable值必须配为0。

配置字段说明

灰度服务类型如表2-58所示，灰度服务配置字段说明请参见表2-59，各类型灰度路由引擎配置样例请参见配置样例，灰度升级配置demo请参见灰度升级配置demo。

表 2-58 灰度服务类型说明

灰度路由引擎	对应ruleType
SLB(>1.3.11)	SLB
微服务	MicroService
DMQ	DMQ
分布式Job	Job
函数	Function

表 2-59 灰度服务配置字段说明

参数名	类型	是否必选	说明
ruleType	String	是	类型：SLB、MicroService、Job、DMQ、Function。
currentStage	int	否	当前阶段，>=1，<=sizeOf(Stages)，默认为1。
currentStatus	String	否	当前状态，取值为Processing、Pause、Complete，默认为Complete。
functionName	String	否	ruleType为Function时必填，否则无效，长度<=200。
functionAliasName	String	否	ruleType为Function时必填，否则无效，长度<=255。 functionAliasName别名不能重复，一个别名只能有一条记录。
routeFlag	String	否	取值为Default/Special，默认为Default。
Stages	List<GrayStage>	是	灰度阶段

表 2-60 GrayStage

参数名	类型	是否必选	说明
stageIndex	int	否	阶段编号，最好按照顺序填写1，2，3，4…，缺省为0，后台会排序后，按顺序赋值为1.2.3.4….。
greyRuleRelation	String	否	取值为or或and RouteType为SLB时必选，不为SLB时无效。
rules	List<**GrayRule**>	否	RouteType为SLB时必选，不为SLB时无效。
groupRelation	String	否	取值为or或and RouteType不为SLB时必选，为SLB时无效。
userExtendedParameter	String	否	RouteType为Job时有意义，非必选，不为job时无效，必须是json字符串格式。
groups	List<**GrayRuleGroup**>	否	RouteType为非SLB时必选，为SLB时无效。

表 2-61 GrayRule

参数名	类型	是否必选	说明
param	String	是	参数名variable为percent时可以为空，否则不能为空，长度<=50。 • Position为header 时，[a-zA-Z0-9_\-]+ • Position不为header且RuleType不为Function时，[.a-zA-Z0-9_\-]+ • Position不为header且RuleType为Function时，[.a-zA-Z0-9_\-\[\]]+

参数名	类型	是否必选	说明
variable	String	是	• RuleType为SLB时，传输类型："province", "city", "countryArea", "custom", "advance", "path","CLIENT-IP", "REAL-IP" • RuleType非SLB时，传输类型： "province", "city", "countryArea", "custom", "percent" ， "CLIENT-IP", "REAL-IP"
position	String	否	位置 • RuleType非SLB时，不传 • variable为 "CLIENT-IP", "REAL-IP" 时，为 "NA" • variable为其他时，为 "url", "body-json", "body-form", "header", "locale","resource", "localeCountryArea", "NA", "path", "percent"
match	String	是	匹配方法 • RuleType为SLB时： "pattern", "sha256", "equal", "rangeIP", "locateIn", "tailEqual", "locateInCountryArea", "NA" • RuleType非SLB时： "pattern", "sha256", "equal", "rangeIP", "locateIn", "tailEqual", "locateInCountryArea", "NA" ， "gt", "ge", "lt", "le", "notEqual", "percent" • variable为 "CLIENT-IP", "REAL-IP" 时，为 "sha256", "rangeIP" • variable为 " path " 时，为 "pattern", "equal"

参数名	类型	是否必选	说明
values	String	是	匹配值列表，多个值逗号分隔 • match为 "tailEqual" 时，每个长度小于4 • match为 " rangeIP " 时，样例xx.xx.xx.x-xx.xx.xx.xx

表 2-62 GrayRuleGroup

参数名	类型	是否必选	说明
groupIndex	Int	否	建议按照顺序填写0，1，2，3，4… 不写默认从0开始排序，按顺序赋值为0，1，2，3，4….。
greyRuleRelation	String	是	取值为and或or
microServiceVersion	String	否	微服务版本号，MicroServiceVersion和useInstanceProperty二选一。 Function类型可以两个都不选，DMQ类型只支持版本号。 如果配置组之间关系为与，各个配置组的版本号相同。 长度<=50
useInstanceProperty	String	否	微服务自定义灰度参数，MicroServiceVersion和useInstanceProperty二选一。 Function类型可以两个都不选，DMQ类型只支持版本号。 长度<=50
rules	List<GrayRule>	是	灰度规则列表

配置样例

SLB类型样例：


```
- name: clf_iac3.0_example_gray_SLB            // name+type唯一定位到一个灰度服务资源
  type: 'WiseCloud::LoadBalancer::GrayConfig'  //资源类型为灰度服务配置
  properties:
    ruleType: SLB
    stages:
      - stageIndex: 1
        rules:
          - param: path
            values: '5555'
            match: equal
            variable: path
            position: path
        greyRuleRelation: or
```

微服务类型样例：

```
- name: clf_iac3.0_example_gray_MicroService
  type: 'WiseCloud::LoadBalancer::GrayConfig'
  properties:
    currentStatus: Processing
    ruleType: MicroService
    stages:
      - stageIndex: 1
        groupRelation: or
        groups:
          - microServiceVersion: latest
            rules:
              - param: aaa
                values: '111'
                match: equal
                variable: custom
            groupIndex: 1
            greyRuleRelation: or
```

DMQ类型样例：

```
- name: clf_iac3.0_example_gray_DMQ
  type: 'WiseCloud::LoadBalancer::GrayConfig'
  properties:
    ruleType: DMQ
    stages:
      - stageIndex: 1
        groupRelation: and
        groups:
          - microServiceVersion: gray_1
            rules:
              - param: aaa
                values: '2222'
                match: pattern
                variable: custom
              - param: bbb
                values: '1111'
                match: equal
                variable: custom
            groupIndex: 1
            greyRuleRelation: or
```

Job类型样例：

```
- name: clf_iac3.0_example_gray_Job
  type: 'WiseCloud::LoadBalancer::GrayConfig'
  properties:
    ruleType: Job
    stages:
      - stageIndex: 1
        groupRelation: or
        groups:
          - microServiceVersion: last
            rules:
              - param: ccc
```

```
            values: ccc
            match: equal
            variable: custom
            position: ''
          groupIndex: 1
          greyRuleRelation: or
      - stageIndex: 2
        groupRelation: or
        groups:
          - microServiceVersion: last
            rules:
              - param: ddd
                values: ddd
                match: tailEqual
                variable: custom
            groupIndex: 1
            greyRuleRelation: or
```

Function类型样例：

```
- name: clf_iac3.0_example_gray_Function
  type: 'WiseCloud::LoadBalancer::GrayConfig'
  properties:
    functionAliasName: >-

wisefunction:222cn:iot:mcb1fbf087634f199f3a4251c5b8a91e:function:pyd:quanzhong2wisefunction:cn:iot:mc
b1fbf087634f199f3a4251c5b8a91e:function:pyd:quanzhong2wisefunction:cn:iot:mcb1fbf087634f199f3a4251c
5b8a91e:function:pyd:quanzhong2
    functionName: pyd
    ruleType: Function
    stages:
      - stageIndex: 1
        groupRelation: or
        groups:
          - rules:
              - param: appid
                values: '123456,22222'
                match: equal
                variable: custom
            groupIndex: 1
            greyRuleRelation: or
```

2.9.3.7 灰度升级配置 demo

本文以灰度升级demo为例，介绍如何开发IaC代码。

容器部署

IaC主体描述文件meta.yaml：

```
type: WiseCloud::Environment
applyPipeline: cn_product_cbu
pipelines:
  - name: gray_other_service_cn_product_cbu
    action: Serial
    tasks:
      - name: gray_other_service_upgrade
        action: Parallel
        tasks:
          - name: gray_slbLogcollector_upgrade
            action: Serial
            tasks:
              - name: slbLogcollector_all_traffic_to_product_nodes    #启动灰度，灰度状态设置为阶段1，运行中
                action: Apply
                component:
                  name: WiseCloudGrayLogCollectorService
                  resources:
                    - type: "WiseCloud::LoadBalancer::GrayConfig"
```

```
                  name: "logCollector"
                  properties:
                    currentStage: 1           #当前灰度阶段
                    currentStatus: Processing  #当前状态，运行中
# 执行runtime灰度部署：4（生产） 0（灰度） -> 2（生产） 2（灰度），业务观察情况：pod正常，全网流量都引入生产节点（2个），灰度节点（2个）上没有流量
- name: gray_slbLogcollector_runtime_upgrade       #升级服务，将灰度流量设置为50%
            action: Apply
            component:
              name: WiseCloudGrayLogCollectorService
              resources:
                - type: "WiseCloud::MicroService::NuwaContainer"
                  name: "WiseCloudGrayLogCollectorService_runtime_tur"
                  properties:
                    grayStage:
                      grayInstances: 50        #灰度升级的实例数比例，范围为1~100
                      grayProcess: "INGRAY"    #灰度状态，灰度处理中
                      grayStatus: 2            #微服务平台实例管理页面pod的灰度状态，1：生产；2：灰度
# 进入第2阶段（切占全网10%的流量到灰度节点），业务观察情况：10%的灰度流量引入灰度节点（2个），剩余流量还是引入生产节点（2个）
          - name: slbLogcollector_gray_traffic_to_gray_nodes    #将灰度状态设置阶段2，运行中
            action: Apply
            component:
              name: WiseCloudGrayLogCollectorService
              resources:
                - type: "WiseCloud::LoadBalancer::GrayConfig"
                  name: "logCollector"
                  properties:
                    currentStage: 2           #当前灰度阶段
                    currentStatus: Processing  #当前状态，运行中

 - name: full_other_service_cn_product_cbu
   action: Serial
   tasks:
     - name: full_other_service_upgrade
       action: Parallel
       tasks:
         - name: full_slbLogcollector_upgrade
           action: Serial
           tasks:
# 先修改灰度阶段，进入第3阶段（切占全网20%的流量到灰度节点），业务观察情况：20%的灰度流量引入灰度节点（2个），剩余流量还是引入生产节点（2个）
           - name: full_slbLogcollector_upgrade_stage1       #灰度状态设置为阶段3，运行中
             action: Serial
             tasks:
               - name: slbLogcollector_all_traffic_to_gray_nodes
                 action: Apply
                 component:
                   name: WiseCloudGrayLogCollectorService
                   resources:
                     - type: "WiseCloud::LoadBalancer::GrayConfig"
                       name: "logCollector"
                       properties:
                         currentStage: 3                  #当前灰度阶段
                         currentStatus: Processing        #当前状态，运行中
#执行runtime灰度部署: 2（生产） 2（灰度） -> 0（生产） 4（灰度）
               - name: full_slbLogcollector_runtime_upgrade  #升级服务
                 action: Apply
                 component:
                   name: WiseCloudGrayLogCollectorService
                   resources:
                     - type: "WiseCloud::MicroService::NuwaContainer"
                       name: "WiseCloudGrayLogCollectorService_runtime_tur"
                       properties:
                         grayStage:
                           grayInstances: 100             #灰度升级的实例数比例，范围为1~100
                           grayProcess: "INGRAY"          #灰度状态，灰度处理中
                           grayStatus: 2                  #灰度实例对接SLB时的状态，1：生产，2：灰度
#修改灰度阶段：灰度比例100%，业务观察情况：所有流量随机分配到4个节点
```

```
          - name: full_slbLogcollector_upgrade_stage2
            action: Serial
            tasks:
              - name: slbLogcollector_complate_gray_release #灰度状态设置为阶段3，已完成
                action: Apply
                component:
                  name: WiseCloudGrayLogCollectorService
                  resources:
                    - type: "WiseCloud::LoadBalancer::GrayConfig"
                      name: "logCollector"
                      properties:
                        currentStage: 3                  #当前灰度阶段
                        currentStatus: Complete          #当前状态，已完成
#修改灰度阶段，完成灰度发布
              - name: slbLogcollector_runtime_switch_to_product #升级服务
                action: Apply
                component:
                  name: WiseCloudGrayLogCollectorService
                  resources:
                    - type: "WiseCloud::MicroService::NuwaContainer"
                      name: "WiseCloudGrayLogCollectorService_runtime_tur"
                      properties:
                        grayStage:
                          grayInstances: 100          #灰度升级的实例数比例，范围为1~100
                          grayProcess: "FINISHED"     #灰度状态，灰度处理完成
                          grayStatus: 2               #灰度实例对接SLB时的状态，1：生产，2：灰度
```

资源列表resources.yaml：

```
- name: WiseCloudGrayLogCollectorService_runtime_tur
  type: WiseCloud::MicroService::NuwaContainer
  metadata:
    annotations: {}
  properties:
    bindSlb:                    #引用config/values.yaml文件中的bindSlb配置，用于绑定SLB
      $ref: 'config/values.yaml#/values/bindSlb'
    configs:
      private:
        schema:
          $ref: config/config_schema.yaml#
        records:
          $ref: config/config_records.yaml#
    sidecars:
      - param: '{}'
        requestsFlavor: 0.1C500M
        name: RASP
        limitsFlavor: 0.3C500M
        version: 2.2.1.102
      - param:
          $ref: config/aiopslog_sidecar_param.json
        requestsFlavor: 0.1C1G
        name: AIOpsLog
        limitsFlavor: 0.5C1G
        version: 3.9.0.200
    terminationGracePeriodSeconds: 30  #优雅退出宽限时间，此时间为整个POD的最大退出时间，设置为30秒
    microserviceName: WiseCloudGrayLogCollectorService
    replicas: 4                 #单AZ主机数量，<=100
    clusterName: runtime_tur
    volumes:                    #引用config/values.yaml文件中的volumes配置
      $ref: 'config/values.yaml#/values/volumes'
    resourceTag:                 #指定集群部署的分组,此处配置的信息要在管理台提前预置好
      group: cce-turbo
    containers:                 #微服务基于容器化的部署
      - flavor: 2C8Gi
        image: >-               #镜像地址
          swr.cn-north-2.myhuaweicloud.com/appstage-trustservices/slb-log-collector:log_collector_version
        livenessProbe:
          $ref: 'config/common_config.yaml#/config/livenessProbe'
          httpGet:
            path: /health
```

```
        scheme: HTTP
        port: 18088
      envs:
        - name: JAVA_OPTS
          value: '-Dfastjson.parser.safeMode=true'
      readinessProbe:
        $ref: 'config/common_config.yaml#/config/readinessProbe'
        exec:
          command:
            - sh
            - /opt/huawei/app/bin/status.sh
            - '18088'
      stsEnable: true
      hostAliases:
        $ref: config/hosts.yaml#
      ports:
        - 18088
      preStopConfig:
        execCommand:
          $ref: 'config/common_config.yaml#/config/execCommand'
  network:
    slbEnable: true            #启用slb
    slbCloudMapConfigs: []

#实例logCollector需要在SLB管理台创建，并且名称一致，灰度阶段流量设置请根据实际情况进行调整
- name: logCollector
  type: WiseCloud::LoadBalancer::GrayConfig
  properties:
    ruleType: SLB
    currentStage: 2
    currentStatus: Complete
    stages:
      - stageIndex: 1            #阶段编号1
        rules:                  #灰度规则列表
          - values: percent[0]
            variable: advance
        greyRuleRelation: or    #设置当前阶段下多个分流配置子项之间的关系为或
      - stageIndex: 2           #阶段编号2
        rules:
          - values: percent[10]
            variable: advance
        greyRuleRelation: or    #设置当前阶段下多个分流配置子项之间的关系为或
      - stageIndex: 3           #阶段编号3
        rules:                  #灰度规则列表
          - values: percent[20]
            variable: advance
        greyRuleRelation: or   #设置当前阶段下多个分流配置子项之间的关系为或
      - stageIndex: 4          #阶段编号4
        rules:                 #灰度规则列表
          - values: percent[100]
            variable: advance
        greyRuleRelation: or  #设置当前阶段下多个分流配置子项之间的关系为或
```

公共资源参数值values.yaml：

在resources.yaml中通过$ref的方式来引用。

```
values:
  volumes:
    - size: 100Gi
      name: log_volume
    - size: 5Gi
      name: data_volume
  bindSlb:
    - targetGroup:
        name: ngx-log-collector-runtime
      port: 18088
      grayStatus: 1
      weight: 1
      listenerGroupName: WGP_SLB
```

```
    maxFails: 3
    timeout: 2
```

虚机部署

meta.yaml：

```
type: WiseCloud::Environment
applyPipeline: cn_product_cbu
pipelines:
  - name: gray_dispatch_cn_product_cbu
    action: Serial
    tasks:
      - name: dispatch_upgrade_gray
        action: Parallel
        tasks:
          - name: dispatch_upgrade_gray
            action: Serial
            tasks:
              - name: offine_dispatch_gray_az1
                action: Apply
                component:
                  name: WiseCloudGrayOpenService
                  resources:
                    - type: "WiseCloud::EAP::Workflow"
                      name: "dispatch_offline_gray_server" #执行action任务，实现灰度节点下线
              - name: upgrade_dispatch_gray_az1
                action: Apply
                component:
                  name: WiseCloudGrayOpenService
                  resources:
                    - type: "WiseCloud::EAP::Workflow"    #灰度节点升级服务
                      name: "dispatch_gray"
              - name: switch_to_gray_stage1
                action: Apply
                confirm:
                  message: 确认dispatch是否切到一阶段
                  users:
                    - l30035828
                component:
                  name: WiseCloudGrayOpenService
                  resources:
                    - type: "WiseCloud::EAP::Workflow"  #在Action任务管理创建action任务：执行action任务将灰度节点上线，并且将灰度规则切换为阶段1
                      name: "dispatch_switch_to_gray_stage1"

  - action: Serial
    name: full_upgrade_cn_product_cbu
    tasks:
      - action: Parallel
        name: switch_to_full_release
        confirm:
          message: 确认执行switch_to_full_release
          users:
            - l30035828
        tasks:
          - action: Serial
            name: dispatch_full_release
            tasks:
              - name: switch_dispatch_full_and_offline
                action: Apply
                component:
                  name: WiseCloudGrayOpenService
                  resources:
                    - type: "WiseCloud::EAP::Workflow" #功能验证完成后执行：在Action任务管理创建action任务，实现生产节点下线，并且灰度规则完成发布
                      name: dispatch_switch_to_full_release_stage

      - action: Parallel
```

```
    name: upgrade_product_server
    confirm:
      message: 确认执行upgrade_product_server
      users:
        - l30035828
    tasks:
      - action: Serial
        name: dispatch_upgrade_product
        tasks:
          - action: Apply
            name: dispatch_upgrade_product
            component:
              name: WiseCloudGrayOpenService    #生产节点升级服务
              resources:
                - type: "WiseCloud::EAP::Workflow"
                  name: dispatch_product

  - action: Parallel
    name: confirm_online_product_server
    confirm:
      message: 确认执行online_product_server
      users:
        - l30035828
    tasks:
      - action: Serial
        name: online_product_dispatch
        tasks:
          - action: Apply
            name: online_product_dispatch
            component:
              name: WiseCloudGrayOpenService   #在Action任务管理创建action任务，实现生产节点上线
              resources:
                - type: "WiseCloud::EAP::Workflow"
                  name: dispatch_online_product_server
```

resources.yaml：

```
#灰度
- name: dispatch_gray
  type: WiseCloud::EAP::Workflow
  properties:
    workflowName: iac
    params:
      instance: MicroService/cn/TrustServices/cn_product_cbu/WiseCloudGrayService/cn_product_cbu/
WiseCloudGrayOpenService/cn_product_cbu
      group: gray
      version: dispatch_version
      variables: {}

#生产
- name: dispatch_product
  type: WiseCloud::EAP::Workflow
  properties:
    workflowName: iac
    params:
      instance: MicroService/cn/TrustServices/cn_product_cbu/WiseCloudGrayService/cn_product_cbu/
WiseCloudGrayOpenService/cn_product_cbu
      group: product
      version: dispatch_version
      variables: {}


- name: dispatch_offline_gray_server
  type: WiseCloud::EAP::Workflow
  properties:
    workflowName: grayStageChange
    params:
      actionName: dispatch_offine_gray_server_cbu

- name: dispatch_switch_to_gray_stage1
```

```
  type: WiseCloud::EAP::Workflow
  properties:
    workflowName: grayStageChange
    params:
      actionName: dispatch_switch_to_gray_stage1_cbu

- name: dispatch_switch_to_full_release_stage
  type: WiseCloud::EAP::Workflow
  properties:
    workflowName: grayStageChange
    params:
      actionName: dispatch_switch_to_full_release_stage_cbu

- name: dispatch_online_product_server
  type: WiseCloud::EAP::Workflow
  properties:
    workflowName: grayStageChange
    params:
      actionName: dispatch_online_produce_server_cbu
```

3 打包规范

3.1 软件包

软件包一般用于虚拟机部署使用，其中包括有软件包（虚拟机部署使用），测试用例包，函数包（函数部署使用）。

文件名

- 文件名后缀只支持zip。
- 文件名只允许包含英文、数字、“-”、“_”、“（）”、“.”、空格，最大长度不超过200。

以上符号是英文符号，不支持中文符号。文件名不合规时，会导致发布电子流失败，并且只能重走电子流。

zip 包大小限制

组合包解压后不超过50G，单个子包解压前不超过30G，解压后不能超过50G（可配置），超出限制将导致电子流发布失败。

文件结构

图 3-1 软件包结构

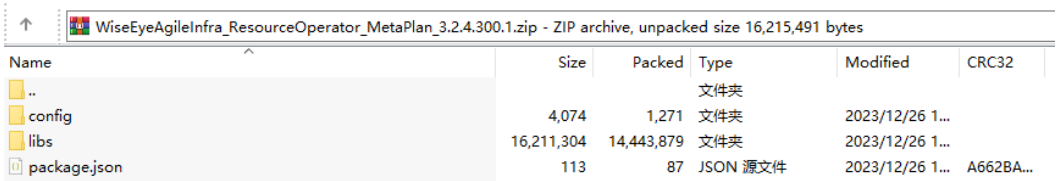


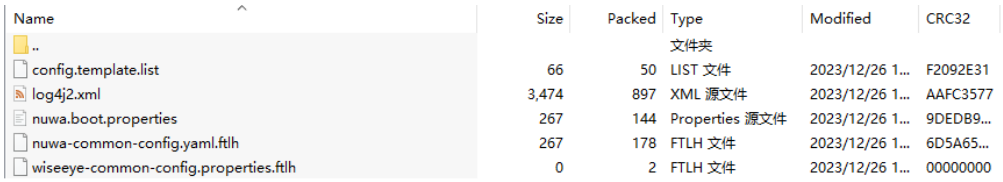
表 3-1 软件包结构说明

位置	类型	描述
config/	文件夹	配置文件所在的目录。

位置	类型	描述
libs/	文件夹	打成的依赖包所在的目录。
package.json	文件	包描述文件。 • 无论是否使用自动部署，都必须包含package.json文件。 • package.json文件必须放在zip包的根目录中。

- config目录

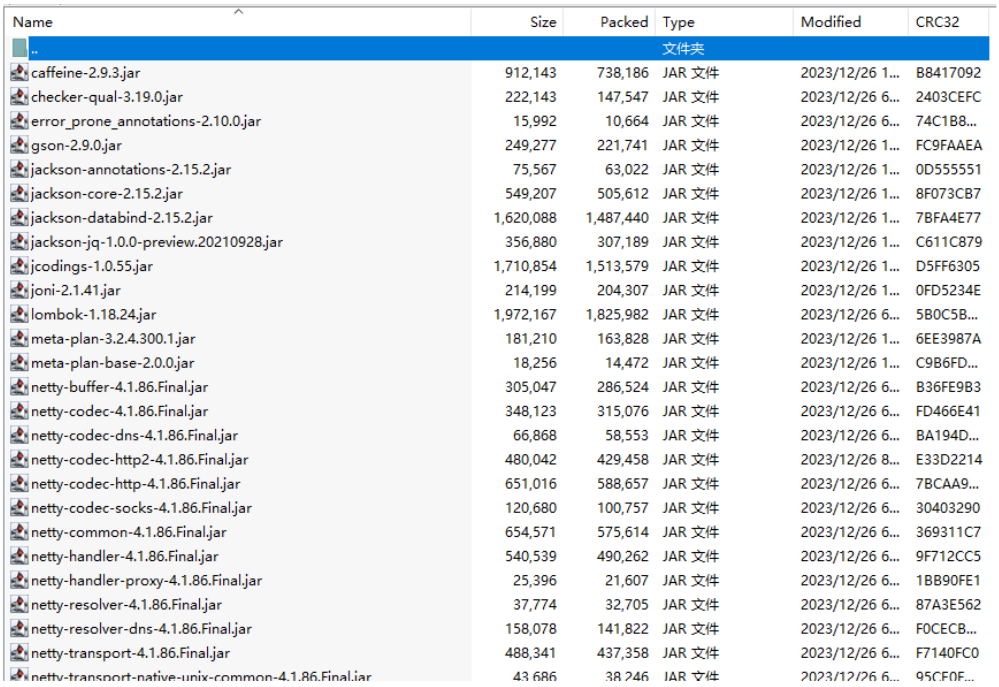
  图 3-2 config 目录



Name	Size	Packed	Type	Modified	CRC32
..			文件夹		
config.template.list	66	50	LIST 文件	2023/12/26 1...	F2092E31
log4j2.xml	3,474	897	XML 源文件	2023/12/26 1...	AAFC3577
nuwa.boot.properties	267	144	Properties 源文件	2023/12/26 1...	9DEDB9...
nuwa-common-config.yaml.ftlh	267	178	FTLH 文件	2023/12/26 1...	6D5A65...
wiseeye-common-config.properties.ftlh	0	2	FTLH 文件	2023/12/26 1...	00000000

- libs目录

  图 3-3 libs 目录



Name	Size	Packed	Type	Modified	CRC32
..			文件夹		
caffeine-2.9.3.jar	912,143	738,186	JAR 文件	2023/12/26 1...	B8417092
checker-qual-3.19.0.jar	222,143	147,547	JAR 文件	2023/12/26 6...	2403CEFC
error_prone_annotations-2.10.0.jar	15,992	10,664	JAR 文件	2023/12/26 6...	74C1B8...
gson-2.9.0.jar	249,277	221,741	JAR 文件	2023/12/26 1...	FC9FAAEA
jackson-annotations-2.15.2.jar	75,567	63,022	JAR 文件	2023/12/26 1...	0D555551
jackson-core-2.15.2.jar	549,207	505,612	JAR 文件	2023/12/26 1...	8F073CB7
jackson-databind-2.15.2.jar	1,620,088	1,487,440	JAR 文件	2023/12/26 1...	7BFA4E77
jackson-jq-1.0.0-preview.20210928.jar	356,880	307,189	JAR 文件	2023/12/26 1...	C611C879
jcodings-1.0.55.jar	1,710,854	1,513,579	JAR 文件	2023/12/26 1...	D5FF6305
joni-2.1.41.jar	214,199	204,307	JAR 文件	2023/12/26 1...	0FD5234E
lombok-1.18.24.jar	1,972,167	1,825,982	JAR 文件	2023/12/26 6...	5B0C5B...
meta-plan-3.2.4.300.1.jar	181,210	163,828	JAR 文件	2023/12/26 1...	6EE3987A
meta-plan-base-2.0.0.jar	18,256	14,472	JAR 文件	2023/12/26 1...	C9B6FD...
netty-buffer-4.1.86.Final.jar	305,047	286,524	JAR 文件	2023/12/26 6...	B36FE9B3
netty-codec-4.1.86.Final.jar	348,123	315,076	JAR 文件	2023/12/26 6...	FD466E41
netty-codec-dns-4.1.86.Final.jar	66,868	58,553	JAR 文件	2023/12/26 6...	BA194D...
netty-codec-http2-4.1.86.Final.jar	480,042	429,458	JAR 文件	2023/12/26 8...	E33D2214
netty-codec-http-4.1.86.Final.jar	651,016	588,657	JAR 文件	2023/12/26 6...	7BCAA9...
netty-codec-socks-4.1.86.Final.jar	120,680	100,757	JAR 文件	2023/12/26 6...	30403290
netty-common-4.1.86.Final.jar	654,571	575,614	JAR 文件	2023/12/26 6...	369311C7
netty-handler-4.1.86.Final.jar	540,539	490,262	JAR 文件	2023/12/26 6...	9F712CC5
netty-handler-proxy-4.1.86.Final.jar	25,396	21,607	JAR 文件	2023/12/26 6...	1BB90FE1
netty-resolver-4.1.86.Final.jar	37,774	32,705	JAR 文件	2023/12/26 6...	87A3E562
netty-resolver-dns-4.1.86.Final.jar	158,078	141,822	JAR 文件	2023/12/26 6...	F0CECB...
netty-transport-4.1.86.Final.jar	488,341	437,358	JAR 文件	2023/12/26 6...	F7140FC0
netty-transport-native-unix-common-4.1.86.Final.jar	43,686	38,246	JAR 文件	2023/12/26 6...	95CF0F...

- package.json

  软件包的package.json内容一般如下：

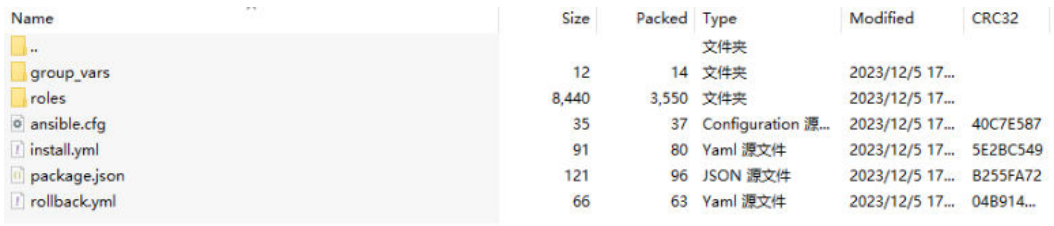
```
{
  "type": "software",                    //软件包类型标识，固定写法，不能随便填写，否则导致电子流异常
  "scope":"1180196813870297011",         //填写为common时，组织下的所有产品都可以使用该软件包；填写产品ID时，该产品下的所有服务可以使用该软件包，在AppStage运维中心右上角的个人账号信息
```

```
管理中，选择“租户管理”，查看产品ID
 "name": "WiseEyeDeployReleaseCenter",    //可以自定义，建议使用微服务名称
 "version": "{{VERSION_ID}}",             //版本号，版本号如果重复，在后面加时间戳，不会覆盖原来的版本，支持英文、数字、“-”、“_”、“（）”、“.”、空格，包含其他字符会导致电子流异常
}
```

3.2 部署包

部署包就是ansible-playbook，作为业务虚拟机部署的脚本使用。

图 3-4 部署包结构



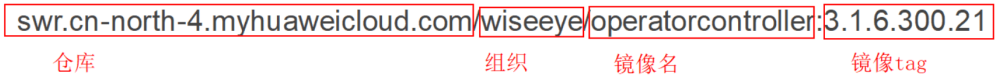
其中package.json里面的内容为playbook。

3.3 镜像包

镜像包，package.json里面的type为docker，一般是以docker镜像的形式存储在镜像仓中。也可以用docker的镜像包打出tar包的形式，同时在tar包同层级目录放置package.json内容，和tar包一起，再打出zip包格式。

容器镜像地址分为四个部分：镜像仓库、组织、镜像名、镜像tag，一个完整的容器镜像地址如图3-5所示。

图 3-5 镜像地址



3.4 SQL 包

SQL包是指包含了通过WiseDBA执行的SQL脚本的zip包，分为DDL和DML两种类型。

文件名

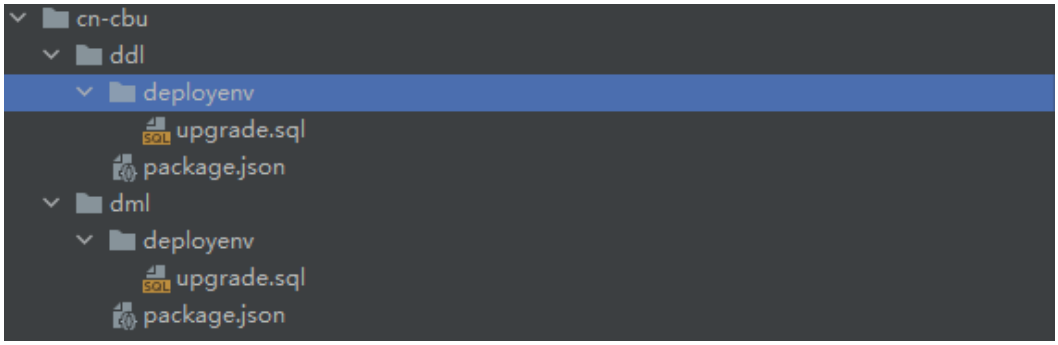
SQL包命名建议：<service_name>-<instance_type>-<site_name>-<type>-<version>.zip

包结构

- 变更多个逻辑库，需要创建多个目录，存放对应逻辑库的sql文件。

- sql文件支持两种命名：upgrade.sql（增量脚本）和rollback.sql（回滚脚本）。
- GeminiDB(for Cassandra)类型数据库sql文件后缀是cql。

图 3-6 SQL 包结构



- DDL的package.json如下所示，主要是写ddl语句。

```
{
  "name": "${service_name}-ddl-sqlchange-cn-cbu",//数据库包的包名，包括站点、业务、服务、实例类型、实例名和包名等信息
  "site_name": "cbu",                          //站点名，中国区为cbu，欧洲区为eu-cbu，亚非拉为aaa-cbu
  "business_name": "${business_name}",         //AppStage业务控制台中业务定义的产品英文名称，查看方式请参考产品管理
  "service_name": "${service_name}",           //AppStage业务控制台中业务定义的服务英文名称，查看方式请参考服务管理
  "instance_name": "${mysql_instance_cn_cbu}",  //WiseDBA中纳管的数据库实例名称
  "instance_type": "GaussDB4MySQL",            //数据库实例类型，支持GaussDB4MySQL/RDS4MySQL/GaussDB4Cassandra/GaussDB4OpenGauss，分别对应WiseDBA中的TaurusDB/RDS for MySQL/GeminiDB(for Cassandra)/GaussDB
  "type": "dbscript_ddl",                      //包类型，ddl语句固定为dbscript_ddl
  "version": "${package_version}"              //数据库包的版本，即包坐标中的version字段，例如：1.0.1
}
```

- DML的package.json如下所示，主要是写dml语句。

```
{
  "name": "${service_name}-dml-sqlchange-cn-cbu",//数据库包的包名，包括站点、业务、服务、实例类型、实例名和包名等信息
  "site_name": "cbu",                          //站点名，中国区为cbu，欧洲区为eu-cbu，亚非拉为aaa-cbu
  "business_name": "${business_name}",         //AppStage业务控制台中业务定义的产品英文名称，查看方式请参考产品管理
  "service_name": "${service_name}",           //AppStage业务控制台中业务定义的服务英文名称，查看方式请参考服务管理
  "instance_name": "${mysql_instance_cn_cbu}",  //WiseDBA中纳管的数据库实例名称
  "instance_type": "GaussDB4MySQL",            //数据库实例类型，支持GaussDB4MySQL/RDS4MySQL/GaussDB4Cassandra/GaussDB4OpenGauss，分别对应WiseDBA中的TaurusDB/RDS for MySQL/GaussDB(for Cassandra)/GaussDB
  "type": "dbscript_dml",                      //包类型，dml语句固定为dbscript_dml
  "version": "${package_version}"              //数据库包的版本，即包坐标中的version字段，例如：1.0.1
}
```

3.5 IaC 3.0 包

IaC3.0推荐以服务为粒度，一个服务的IaC代码打包为一个zip包进行版本发布。

文件名

- 文件名后缀只支持zip。
- 文件名只允许包含英文、数字、“-”、“_”、“（）”、“.”、空格，最大长度不超过200。

以上符号是英文符号，不支持中文符号。文件名不合规时，会导致发布电子流失败，并且只能重走电子流。

包结构

IaC3.0 包有IaC Spec和IaC Patch两种类型，具体包结构介绍请参见**IaC Spec包典型目录结构**和**IaC Patch包典型目录结构**。

3.6 Terraform 包

Terraform包的规范请参考**Terraform**。

3.7 TF 模板包

运维中心集成华为云资源编排服务RFS，用于管理系统资源及服务资源。RFS主要包含模板和资源栈两部分，TF模板是用来创建、更新资源栈的脚本。

研发包结构

Service下所有的main.tf文件中的Resource实例不能有重叠，否则会导致资源被覆盖或删除。

图 3-7 研发包结构

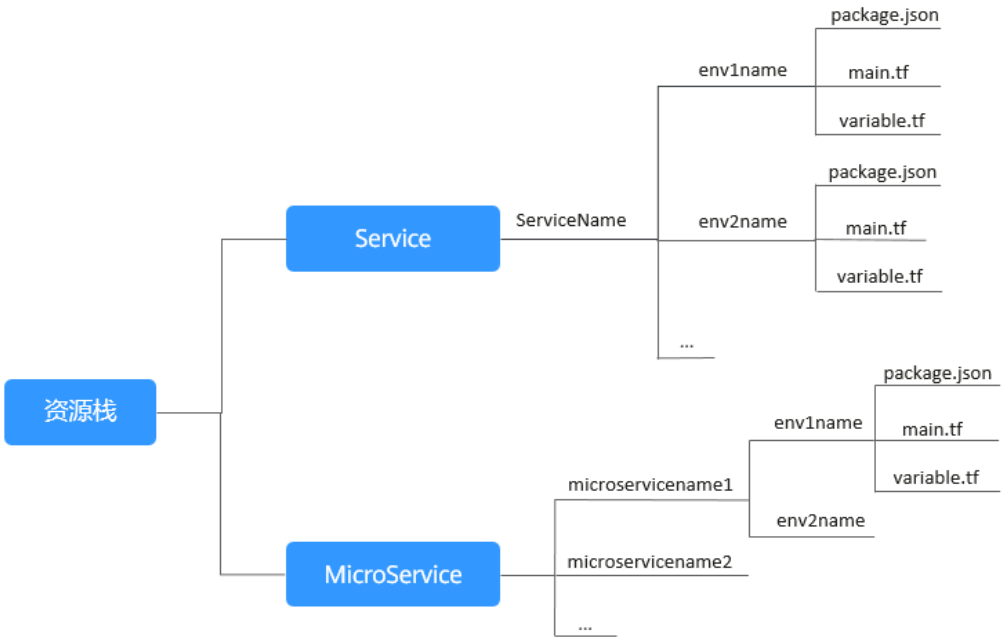


表 3-2 研发包结构介绍

文件名	说明
Service（推荐方式）	按云服务维度的资源模板，整个云服务涉及资源的生命周期都在同一个RFS资源模板中管理。

文件名	说明
MicroService	按微服务维度的资源模板管理，当一个云服务的每个微服务有单独的资源时使用。
envXname	云服务/微服务部署环境，每个环境的RFS包资源都是与环境配套，每个环境都有单独的资源模板。

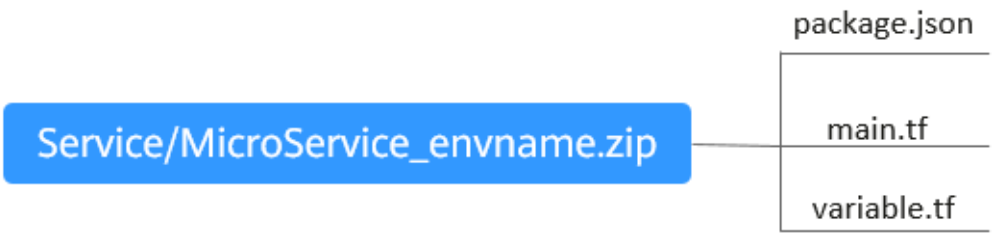
package.json	固定名称与格式，发布包版本信息，类型，软件包名称，版本。
	{
	"type": "rfstemplate", // 固定
	"name": "templatename", // RFS模板名称，服务下唯一
	"envid": "envid1", // 云服务的环境ID
	"version": "1.0.0", // RFS模板版本号
	"maintf": "main.tf", // 实际创建资源的TF文件
	"variablestf": "variable.tf" // 存储参数变量的TF文件
	}

main.tf	RFS实际使用的模板文件，通过此文件来管理华为云资源。当前支持的华为云资源有CCE、DCS、DNS、ECS、Kafka、RDS及VPC，模板文件样例分别参考**CCE资源RFS模板文件**、**DCS资源RFS模板文件**、**DNS资源RFS模板文件**、**ECS资源RFS模板文件**、**Kafka资源RFS模板文件**、**RDS资源RFS模板文件**及**VPC资源RFS模板文件**。 详细的开发规范请参考**Terraform文档**。
variable.tf	RFS资源模板中涉及到的变量值，可能多个云服务变量名称不一致，提取出来单独设置。样例如下： CCE-Name-1234: "CCE-Cluster-ERS" 建议敏感数据不在此文件设置，敏感数据在界面手动输入。

发布包结构

发布包作为package发布，以服务或微服务加上环境名称命名为发布包。

图 3-8 发布包结构



CCE 资源 RFS 模板文件

```
{
  "terraform": {
    "required_providers": {
```

```
        "huaweicloud": {
            "source": "huawei.com/provider/huaweicloud",
            "version": "1.56.0"
        }
    }
},
"provider": {
    "huaweicloud": {
        "auth_url": "https://iam.cn-north-4.myhuaweicloud.com/v3",
        "insecure": true,
        "region": "cn-north-4"
    }
},
"resource": {
    "huaweicloud_cce_cluster": {
        "cce-cluster-yfclf": {
            "vpc_id": "18b117f8-****-****-****-6d022db472a1",
            "name": "cce-cluster-rfs-001",
            "cluster_version": "v1.27",
            "charging_mode": "postPaid",
            "flavor_id": "cce.s2.small",
            "container_network_type": "vpc-router",
            "enterprise_project_id": "2191bb05-****-****-****-96f098494b8d",
            "subnet_id": "bef6af2f-****-****-****-9e78ef03eb6a"
        }
    },
    "huaweicloud_cce_node": {
        "cce-node-pmqnn": {
            "name": "cce-node-rfs-001",
            "charging_mode": "postPaid",
            "flavor_id": "c7.large.2",
            "os": "Huawei Cloud EulerOS 2.0",
            "runtime": "containerd",
            "root_volume": {
                "size": 50,
                "volumetype": "SAS"
            },
            "password": "${var.CCE-Node-Password-u9fi}",
            "cluster_id": "d6eb9020-****-****-****-0255ac1000ac",
            "availability_zone": "cn-north-4c",
            "data_volumes": [{
                "volumetype": "SAS",
                "size": 100
            }],
            "storage": {
                "selectors": [{
                    "name": "cceUse",
                    "type": "evs",
                    "match_label_count": 1,
                    "match_label_size": 100,
                    "match_label_volume_type": "SAS"
                }],
                "groups": [{
                    "name": "vgpaas",
                    "cce_managed": true,
                    "selector_names": ["cceUse"],
                    "virtual_spaces": [{
                        "name": "runtime",
                        "size": "90%",
                        "runtime_lv_type": "linear"
                    }, {
                        "name": "kubernetes",
                        "size": "10%",
                        "lvm_lv_type": "linear"
                    }]
                }]
            },
            "depends_on": ["huaweicloud_cce_cluster.cce-cluster-yfclf"],
            "subnet_id": "bef6af2f-****-****-****-9e78ef03eb6a"
```



```
            }
        }
    },
    "variable": {
        "CCE-Node-Password-u9fi": {
            "description": "Password for cce-node-pmqnn",
            "type": "string",
            "sensitive": true,
            "nullable": false,
            "default": ""
        }
    }
}
```

DCS 资源 RFS 模板文件

```
{
    "terraform": {
        "required_providers": {
            "huaweicloud": {
                "source": "huawei.com/provider/huaweicloud",
                "version": "1.56.0"
            }
        }
    },
    "provider": {
        "huaweicloud": {
            "auth_url": "https://iam.cn-north-4.myhuaweicloud.com/v3",
            "insecure": true,
            "region": "cn-north-4"
        }
    },
    "resource": {
        "huaweicloud_dcs_instance": {
            "dcs-cx83b": {
                "charging_mode": "postPaid",
                "availability_zones": ["cn-north-4a", "cn-north-4a"],
                "vpc_id": "18b117f8-****-****-****-6d022db472a1",
                "subnet_id": "bef6af2f-****-****-****-9e78ef03eb6a",
                "maintain_begin": "18:00:00",
                "maintain_end": "22:00:00",
                "engine_version": "5.0",
                "capacity": 1,
                "flavor": "redis.ha.xu1.large.r2.1",
                "enterprise_project_id": "2191bb05-****-****-****-96f098494b8d",
                "password": "${var.DCS-Password-qvwu}",
                "name": "dcs-rfs-001",
                "engine": "Redis"
            }
        }
    },
    "variable": {
        "DCS-Password-qvwu": {
            "description": "dcs password for dcs-cx83b",
            "type": "string",
            "sensitive": true,
            "nullable": true,
            "default": null
        }
    }
}
```

DNS 资源 RFS 模板文件

```
{
    "terraform": {
        "required_providers": {
            "huaweicloud": {
                "source": "huawei.com/provider/huaweicloud",
```



```
            "version": "1.56.0"
        }
    }
  },
  "provider": {
    "huaweicloud": {
      "auth_url": "https://iam.cn-north-4.myhuaweicloud.com/v3",
      "insecure": true,
      "region": "cn-north-4"
    }
  },
  "resource": {
    "huaweicloud_dns_zone": {
      "dns-zone-iz7r1": {
        "enterprise_project_id": "2191bb05-****-****-****-96f098494b8d",
        "zone_type": "private",
        "router": {
          "router_id": "c6131e37-****-****-****-5fffa75982f3"
        },
        "name": "exampleninenine.com"
      }
    }
  }
}
```


ECS 资源 RFS 模板文件


```
{
  "terraform": {
    "required_providers": {
      "huaweicloud": {
        "source": "huawei.com/provider/huaweicloud",
        "version": "1.56.0"
      }
    }
  },
  "provider": {
    "huaweicloud": {
      "auth_url": "https://iam.cn-north-4.myhuaweicloud.com/v3",
      "insecure": true,
      "region": "cn-north-4"
    }
  },
  "resource": {
    "huaweicloud_compute_instance": {
      "ecs-qh7h5": {
        "name": "ecs-rfs-001",
        "charging_mode": "postPaid",
        "admin_pass": "${var.ECS-Password-ia2c}",
        "flavor_id": "s6.small.1",
        "system_disk_type": "SAS",
        "availability_zone": "cn-north-4a",
        "network": {
          "uuid": "bef6af2f-****-****-****-9e78ef03eb6a"
        },
        "security_group_ids": ["986d4460-****-****-****-f5f237df42c0"],
        "enterprise_project_id": "2191bb05-****-****-****-96f098494b8d",
        "system_disk_size": 40,
        "image_id": "86405805-****-****-****-09f30b497c98"
      }
    }
  },
  "variable": {
    "ECS-Password-ia2c": {
      "description": "Ecs password for ecs-qh7h5",
      "type": "string",
      "sensitive": true,
      "nullable": true,
      "default": null
```



```
    }
  }
}
```

Kafka 资源 RFS 模板文件

```
{
  "terraform": {
    "required_providers": {
      "huaweicloud": {
        "source": "huawei.com/provider/huaweicloud",
        "version": "1.56.0"
      }
    }
  },
  "provider": {
    "huaweicloud": {
      "auth_url": "https://iam.cn-north-4.myhuaweicloud.com/v3",
      "insecure": true,
      "region": "cn-north-4"
    }
  },
  "resource": {
    "huaweicloud_dms_kafka_instance": {
      "kafka-gq8ef": {
        "manager_password": "${var.Kafka-Manager-Password-nsbl}",
        "name": "kafka-rfs-001",
        "charging_mode": "postPaid",
        "manager_user": "kafka-manager-rfs",
        "flavor_id": "s6.2u4g.cluster.small",
        "engine_version": "3.x",
        "broker_num": 3,
        "storage_spec_code": "dms.physical.storage.high.v2",
        "availability_zones": ["cn-north-4a"],
        "storage_space": 300,
        "vpc_id": "18b117f8-****-****-****-6d022db472a1",
        "network_id": "bef6af2f-****-****-****-9e78ef03eb6a",
        "security_group_id": "32ed0723-****-****-****-7c0fb748d436",
        "enterprise_project_id": "2191bb05-****-****-****-96f098494b8d",
        "retention_policy": "time_base"
      }
    }
  },
  "variable": {
    "Kafka-Manager-Password-nsbl": {
      "description": "Manager Password for kafka-gq8ef",
      "type": "string",
      "sensitive": true,
      "nullable": false,
      "default": ""
    }
  }
}
```

RDS 资源 RFS 模板文件

```
{
  "terraform": {
    "required_providers": {
      "huaweicloud": {
        "source": "huawei.com/provider/huaweicloud",
        "version": "1.56.0"
      }
    }
  },
  "provider": {
    "huaweicloud": {
      "auth_url": "https://iam.cn-north-4.myhuaweicloud.com/v3",
      "insecure": true,
```



```
        "region": "cn-north-4"
      }
    },
    "resource": {
      "huaweicloud_rds_instance": {
        "rds-instance-yaqsb": {
          "name": "rds-rfs-001",
          "charging_mode": "postPaid",
          "db": {
            "type": "MySQL",
            "version": "8.0",
            "password": "${var.RDS-Password-u6z0}"
          },
          "ha_replication_mode": "async",
          "volume": {
            "size": 40,
            "type": "ULTRAHIGH"
          },
          "vpc_id": "18b117f8-****-****-****-6d022db472a1",
          "subnet_id": "bef6af2f-****-****-****-9e78ef03eb6a",
          "security_group_id": "32ed0723-****-****-****-7c0fb748d436",
          "enterprise_project_id": "2191bb05-****-****-****-96f098494b8d",
          "availability_zone": ["cn-north-4c", "cn-north-4c"],
          "flavor": "rds.mysql.c6.large.2.ha"
        }
      }
    },
    "variable": {
      "RDS-Password-u6z0": {
        "description": "Password for rds-instance-yaqsb",
        "type": "string",
        "sensitive": true,
        "nullable": false,
        "default": ""
      }
    }
}
```

VPC 资源 RFS 模板文件

```
{
    "terraform": {
      "required_providers": {
        "huaweicloud": {
          "source": "huawei.com/provider/huaweicloud",
          "version": "1.56.0"
        }
      }
    },
    "provider": {
      "huaweicloud": {
        "auth_url": "https://iam.cn-north-4.myhuaweicloud.com/v3",
        "insecure": true,
        "region": "cn-north-4"
      }
    },
    "resource": {
      "huaweicloud_vpc": {
        "vpc-krkup": {
          "name": "vpc-rfs-001",
          "cidr": "192.168.0.0/16",
          "enterprise_project_id": "2191bb05-****-****-****-96f098494b8d"
        }
      },
      "huaweicloud_vpc_subnet": {
        "vpc-subnet-ppnwk": {
          "name": "subnet-rfs-001",
          "cidr": "192.168.3.0/24",
          "gateway_ip": "192.168.3.1",
```



```
            "vpc_id": "c6131e37-****-****-****-5fffa75982f3",
            "depends_on": ["huaweicloud_vpc.vpc-krkup"]
        }
    },
    "huaweicloud_vpcep_endpoint": {
        "vpcep_endpoint-4epnv": {
            "service_id": "ebc591db-****-****-****-15354c9bef25",
            "network_id": "${huaweicloud_vpc_subnet.vpc-subnet-ppnwk.id}",
            "vpc_id": "${huaweicloud_vpc_subnet.vpc-subnet-ppnwk.vpc_id}"
        }
    },
    "huaweicloud_networking_secgroup": {
        "sg-rmo7v": {
            "name": "sg-rfs-all-deny",
            "enterprise_project_id": "2191bb05-****-****-****-96f098494b8d",
            "description": "通用Web服务器，默认放通22、3389、80、443端口和ICMP协议。 适用于需要远程
登录、公网ping及用于网站服务的云服务器场景。",
            "depends_on": ["huaweicloud_vpc.vpc-krkup"]
        }
    },
    "huaweicloud_networking_secgroup_rule": {
        "sg-rule-d28sj": {
            "action": "allow",
            "direction": "ingress",
            "ethertype": "IPv4",
            "protocol": "icmp",
            "remote_ip_prefix": "0.0.0.0/0",
            "priority": 1,
            "security_group_id": "${huaweicloud_networking_secgroup.sg-rmo7v.id}"
        }
    }
  }
}
```


4 附录

4.1 使用 configparser 工具优化代码

configparser为自定义参数解析工具，通过NUWA部署时，解析参数模板，将模板中的参数变量，替换为实际的配置项值。

准备工作

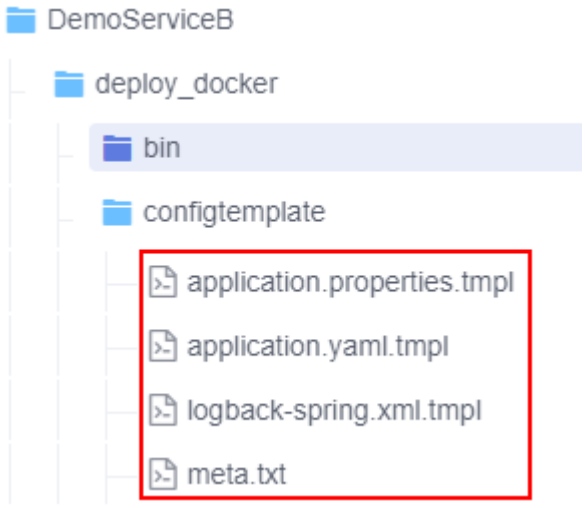
1. 下载configparser工具，在tools文件夹中获取configparser工具。
2. 将本地的configparser工具复制粘贴至项目的bin目录下。

使用 configparser 工具

步骤1 在Dockerfile文件中，为configparser文件添加可执行权限。

步骤2 定义业务配置项模板文件和meta.txt，样例如图1 业务配置项模板所示。

图 4-1 业务配置项模板



步骤3 在业务配置项模板文件（.tmpl文件）中，使用 {{参数名称}} 格式定义需要动态替换的参数，样例如图4-2所示。

图 4-2 模板参数定义

```
sts.server.domain={{sts.server.domain}}
sts.config.path={{sts.config.path}}
```

步骤4 在meta.txt文件中定义需要替换的业务配置项模板文件。如图4-3所示。

图 4-3 替换业务配置项模板文件

```
./application.properties.tmpl|../service/config/application.properties
./application.yaml.tmpl|../service/config/application.yaml
./logback-spring.xml.tmpl|../service/config/logback-spring.xml
```

- application.properties.tmpl为配置文件模板，application.properties为目标配置文件。
- 配置中指定的文件路径是相对于meta.txt文件的路径。

步骤5 启动业务进程之前，在启动脚本中调用configparser工具，进行参数替换。使用方式如下：

```
/opt/huawei/app/bin/configparser -meta /opt/huawei/app/configtemplate/meta.txt -log configparser.log -mode front -tempPath /opt/huawei/app/configtemplate/config-temp
```

- 使用绝对路径的方式调用configparser工具，/opt/huawei/app/bin/为容器启动时的绝对路径。
- -meta：指定meta.txt文件，/opt/huawei/app/configtemplate/为容器启动时meta.txt文件的绝对路径。
- -log：存放configparser工具的运行日志。
- -mode front：固定使用此参数值。
- -tempPath：工具运行过程中生成临时文件的路径。

  /opt/huawei/app/configtemplate/为容器启动时的绝对路径，必须保证此目录路径存在。config-temp文件夹可以不存在，会自动创建，运行结束后会清理此路径。 如果不配置，默认使用/opt/huawei/app/nuwa/config-temp。

----结束

