

dubbo Mock接口

Dubbo Mock接口

说明书
版本 0.1
项目名称：

修订历史

版本号	作者	修订章节	修订原因	修订日期
0.1	甘建新	初稿	初稿	2018-08-23

目 录

- Dubbo Mock接口
- 目 录
- 1 管控台配置方法mock
 - 1.1 环境介绍
 - 1.1.1 开发环境地址
 - 1.1.2 测试1环境
 - 1.1.3 测试2环境
 - 1.1.4 准生产环境
 - 1.2 操作步骤
 - 1.2.1 选择需要Mock的服务
 - 1.2.2 新建动态配置
 - 1.2.3 填写对应的Mock方式
 - 1.2.4 查看动态配置列表
 - 1.2.5 测试Mock结果
- 2 Provider编码方式Mock
- 3 Consumer编码方式Mock
 - 3.1 Stub介绍
 - 3.2 属性参数介绍
 - 3.3 配置文件介绍
 - 3.3.1 配置boolean示例
 - 3.3.2 配置具体类示例
 - 3.4 Stub类介绍
 - 3.4.1 实现方式
 - 3.4.2 代码示例
- 4 参考资料

管控台配置方法mock

以下主要介绍了通过管控台配置提供者Mock的方法；

环境介绍

开发环境地址

<http://192.168.180.88:8080/index.htm#/admin/apps>

测试1环境

<http://10.251.6.100:9003/index.htm#/admin/apps>

测试2环境

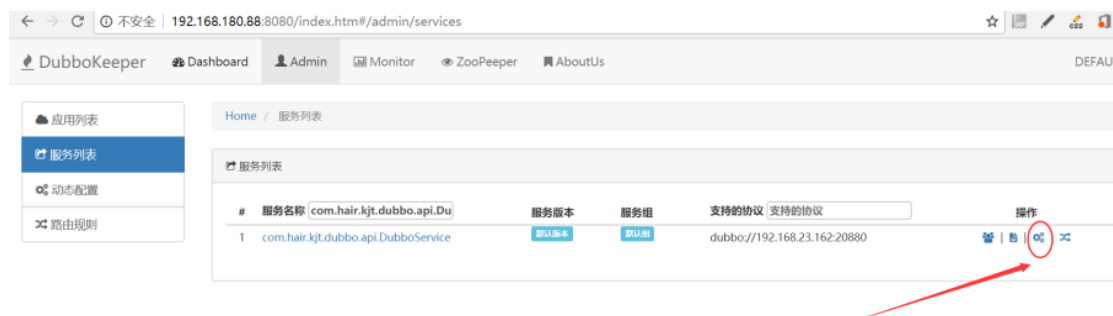
<http://10.252.6.100:9003/index.htm#/admin/apps>

准生产环境

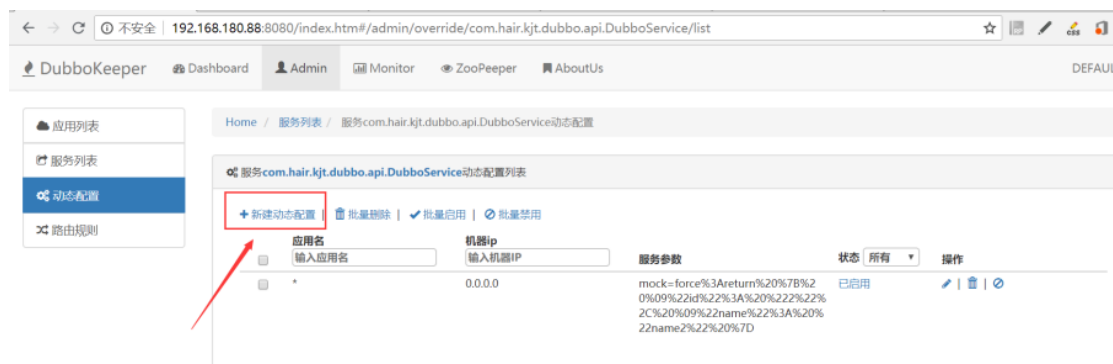
<http://10.255.6.154:9003/index.htm#/admin/apps>

操作步骤

选择需要Mock的服务



新建动态配置



填写对应的Mock方式

服务降级方式选择"屏蔽", 该方式是告诉消费者无须调用提供者, 在返回值部分需要返回对应json值; 如图中填写

```
return {  
  "id": "2",  
  "name": "name2"  
}
```

为com.hair.kjt.dubbo.api.DubboService服务新增动态配置

服务名: com.hair.kjt.dubbo.api.DubboService

消费者应用名: 输入应用名 不填表示对所有消费者生效

绘制定服务提供者配置: 输入地址信息 不填表示对提供者应用的所有机器生效

服务超时时间: 输入超时时间

状态: 禁用

服务端系统参数设置 消费端系统参数设置 服务降级设置 自定义参数

服务降级

所有方法的Mock值: 屏蔽

Mock值: return { "id": "2", 示例: return null/empty/JSON或throw com.foo.BarException

新增方法

保存

查看动态配置列表

Home / 服务列表 / 动态配置概要列表 / 服务com.hair.kjt.dubbo.api.DubboService动态配置

为com.hair.kjt.dubbo.api.DubboService服务动态配置列表

+ 新增动态配置 | 批量删除 | 批量启用 | 批量禁用

应用名	机器ip	服务参数	状态	操作
输入应用名	输入机器IP		所有	
*	0.0.0.0	mock=force%3Areturn%20%7B%20%09%22id%22%3A%20%22%22%2C%20%09%22name%22%3A%20%22name2%22%20%7D	已启用	编辑 删除 禁用

测试Mock结果

Problems Javadoc Declaration Console Progress Servers

```
<terminated> Rerun com.hair.kjt.dubbo.test.ConsumerTest.startConsumer [JUnit] D:\Program Files\Java\jdk1.8.0_131\bin\java  
12:56:27,843 [main-SendThread()] INFO ClientCnxn:1041 - Opening socket connection to server /192.  
12:56:32,390 [main-SendThread(192.168.180.42:2181)] INFO ClientCnxn:949 - Socket connection estab  
12:56:32,396 [main-SendThread(192.168.180.42:2181)] INFO ClientCnxn:738 - Session establishment c  
12:56:32,398 [main-EventThread] INFO ZkClient:449 - zookeeper state changed (SyncConnected)  
12:56:32,791 [main] INFO ConsumerService:60 - consumer:DemoVo[id=2,name=name2]  
12:56:32,795 [Thread-2] INFO GenericApplicationContext:1048 - Closing org.springframework.context  
12:56:32,797 [Thread-2] INFO DefaultListableBeanFactory:444 - Destroying singletons in org.spring  
12:56:32,804 [ZkClient-EventThread-13-192.168.180.42:2181] INFO ZkEventThread:82 - Terminate ZkCl  
12:56:32,821 [DubboShutdownHook] INFO ZooKeeper:538 - Session: 0x165671cd8d00072 closed  
12:56:32,821 [main-EventThread] INFO ClientCnxn:520 - EventThread shut down
```

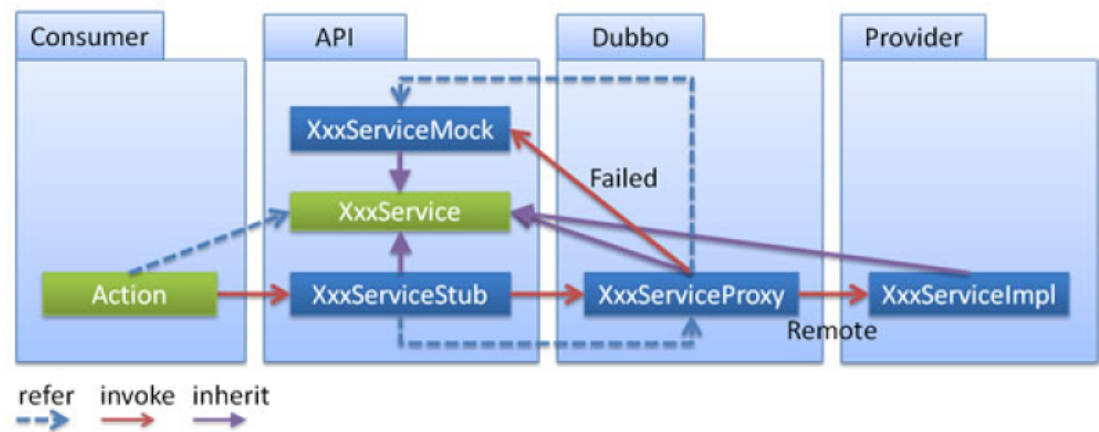
Provider编码方式Mock

忽略；

Consumer编码方式Mock

Stub介绍

当使用rpc时，客户端通常只有接口，但有时客户端也想在客户端执行部分逻辑。
例如：执行ThreadLocal缓存，验证参数，在调用失败时返回模拟数据等。
要解决此问题，您可以在API中配置存根，以便在客户端生成代理实例时，它通过构造函数将代理传递给Stub，然后您可以在存根实现代码中实现您的逻辑。



属性参数介绍

<dubbo:reference>	stub	class/boolean	服务接口客户端本地代理类名，用于在客户端执行本地逻辑，如本地缓存等，该本地代理类的构造
<dubbo:reference>	mock	class/boolean	服务接口调用失败Mock实现类名，该Mock类必须有一个无参构造函数，与Local的区别在于，Lo

配置文件介绍

在消费接口上加上stub属性；

配置boolean示例

```
<dubbo:reference id="dubboService" interface="com.hair.kjt.dubbo.api.IDubboService" stub="true"/>
```

默认找interface接口名+Stub类；

配置具体类示例

```
<dubbo:reference id="dubboService" interface="com.hair.kjt.dubbo.api.IDubboService" stub="com.hair.kjt.dubbo.stub.DubboServiceStub"/>
```

Stub类介绍

实现方式

- 继承对应接口类，并实现需要处理的业务逻辑；

- Stub必须有一个可以传入代理的构造函数；

代码示例

```
package com.hair.kjt.dubbo.stub;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import com.hair.kjt.dubbo.api.IDubboService;
import com.hair.kjt.dubbo.domain.DemoVo;

/**
 *
 * • TODO
 *
 * • @author ganjianxin
 * • @version $Id: DubboServiceStub.java, v 0.1 2018年8月24日 下午4:30:04 ganjianxin
 * • Exp $
 */
public class DubboServiceStub implements IDubboService {

    /** */
    Logger logger = LoggerFactory.getLogger(DubboServiceStub.class);
    /** */
    private IDubboService dubboService;

    /**
    •
    */
    public DubboServiceStub() {
        // TODO Auto-generated constructor stub
    }

    /**
    • @param dubboService
    */
    public DubboServiceStub(IDubboService dubboService) {
        // TODO Auto-generated constructor stub
        this.dubboService = dubboService;
    }

    /**
    • @see com.hair.kjt.dubbo.api.IDubboService#printWord(com.hair.kjt.dubbo.domain.DemoVo)
    */
    @Override
    public DemoVo printWord(DemoVo demo) throws InterruptedException {
        // TODO Auto-generated method stub
        demo.setName("stub");
        logger.info("stub:{}", demo);
        return demo;
    }

    public IDubboService getDubboService() {
        return dubboService;
    }

    public void setDubboService(IDubboService dubboService) {
        this.dubboService = dubboService;
    }

}
```

参考资料

<http://dubbo.apache.org/en-us/docs/user/demos/local-stub.html>
<http://dubbo.apache.org/en-us/docs/user/demos/local-mock.html>