

第二节: Dubbo的基本应用与高级应用

课程内容

笔记更新地址：

负载均衡

服务超时

集群容错

服务降级

本地存根

本地伪装

参数回调

异步调用

泛化调用

泛化服务

Dubbo中的REST

管理台

动态配置

服务路由

什么是蓝绿发布、灰度发布

Zookeeper可视化客户端工具

课程内容

1. 负载均衡、集群容错、服务降级
2. 本地存根、本地伪装、参数回调
3. 异步调用、泛化调用、动态配置
4. 管理台、动态配置、服务路由

笔记更新地址：

<https://www.yuque.com/books/share/f2394ae6-381b-4f44-819e-c231b39c1497>（密码：kyys）

《Dubbo笔记》

官网：<http://dubbo.apache.org/zh/docs/v2.7/user/>

demo项目地址：<https://gitee.com/archguide/dubbo-tuling-demo>

clone地址：<https://gitee.com/archguide/dubbo-tuling-demo.git>

管理台github地址：<https://github.com/apache/dubbo-admin>

Dubbo提供了很多功能，这里我们只介绍几种比较重要的，其他功能可以去Dubbo官网上查看。

负载均衡

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/loadbalance/>

如果在消费端和服务端都配置了负载均衡策略，以消费端为准。

这其中比较难理解的就是**最少活跃调用数**是如何进行统计的？

讲道理，最少活跃数应该是在**服务提供者端**进行统计的，服务提供者统计有多少个请求正在执行中。但在Dubbo中，就是**不讲道理**，它是在消费端进行统计的，为什么能在消费端进行统计？

逻辑是这样的：

1. 消费者会缓存所调用服务的所有提供者，比如记为p1、p2、p3三个服务提供者，每个提供者内都有一个属性记为active，默认位0
2. 消费者在调用次服务时，如果负载均衡策略是**leastactive**
3. 消费者端会判断缓存的所有服务提供者的active，选择最小的，如果都相同，则随机
4. 选出某一个服务提供者后，假设位p2，Dubbo就会对p2.active+1
5. 然后真正发出请求调用该服务
6. 消费端收到响应结果后，对p2.active-1
7. 这样就完成了对某个服务提供者当前活跃调用数进行了统计，并且并不影响服务调用的性能

服务超时

在服务提供者和服务消费者上都可以配置服务超时时间，这两者是不一样的。

消费者调用一个服务，分为三步：

1. 消费者发送请求（网络传输）
2. 服务端执行服务
3. 服务端返回响应（网络传输）

如果在服务端和消费端只在**其中一方**配置了timeout，那么没有歧义，表示消费端**调用服务的超时时间**，消费端如果超过时间还没有收到响应结果，则消费端会抛**超时异常**，**但**，服务端不会抛异常，服务端在执行服务后，会检查**执行该服务**的时间，如果超过timeout，则会打印一个**超时日志**。服务会正常的执行完。

如果在服务端和消费端各配了一个timeout，那就比较复杂了，假设

1. 服务执行为5s
2. 消费端timeout=3s
3. 服务端timeout=6s

那么消费端调用服务时，消费端会收到超时异常（因为消费端超时了），服务端一切正常（服务端没有超时）。

集群容错

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/fault-tolerent-strategy/>

集群容错表示：服务消费者在调用某个服务时，这个服务有多个服务提供者，在经过负载均衡后选出其中一个服务提供者之后进行调用，但调用报错后，Dubbo所采取的后续处理策略。

服务降级

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/service-downgrade/>

服务降级表示：服务消费者在调用某个服务提供者时，如果该服务提供者报错了，所采取的措施。

集群容错和服务降级的区别在于：

1. 集群容错是整个集群范围内的容错
2. 服务降级是单个服务提供者的自身容错

本地存根

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/local-stub/>

本地存根，名字很抽象，但实际上不难理解，本地存根就是一段逻辑，这段逻辑是在服务消费端执行的，这段逻辑一般都是由服务提供者提供，服务提供者可以利用这种机制在服务消费者远程调用服务提供者之

前或之后再做一些其他事情，比如结果缓存，请求参数验证等等。

本地伪装

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/local-mock/>

本地伪装就是Mock，Dubbo中Mock的功能相对于本地存根更简单一点，Mock其实就是Dubbo中的服务容错的解决方案。

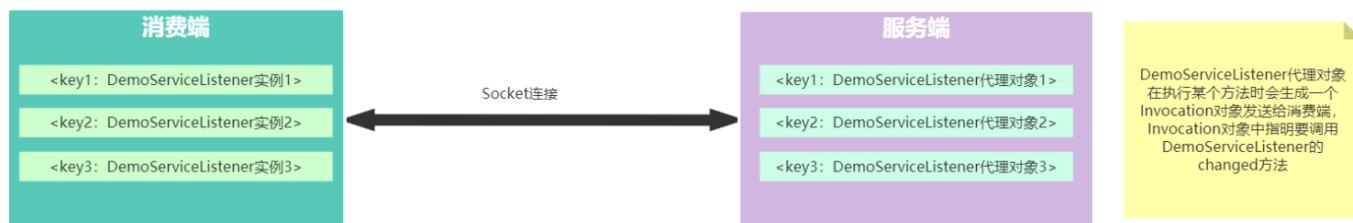
参数回调

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/callback-parameter/>

官网上的Demo其实太复杂，可以看课上的Demo更为简单。

首先，如果当前服务支持参数回调，意思就是：对于某个服务接口中的某个方法，如果想支持消费者在调用这个方法时能设置回调逻辑，那么该方法就需要提供一个入参用来表示回调逻辑。

因为Dubbo协议是基于长连接的，所以消费端在两次调用同一个方法时想指定不同的回调逻辑，那么就需要在调用时在指定一定key进行区分。



异步调用

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/async-call/>

理解起来比较容易，主要要理解CompletableFuture，如果不理解，就直接把它理解为Future

其他异步调用方式：<https://mp.weixin.qq.com/s/U3eyBUy6HBVy-xRw3LGBRQ>

泛化调用

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/generic-reference/>

泛化调用可以用来做服务测试。

在Dubbo中，如果某个服务想要支持泛化调用，就可以将该服务的generic属性设置为true，那对于服务消费者来说，就可以不用依赖该服务的接口，直接利用GenericService接口来进行服务调用。

泛化服务

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/generic-service/>

实现了GenericService接口的就是泛化服务

Dubbo中的REST

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/rest/>

注意Dubbo的REST也是Dubbo所支持的一种协议。

当我们用Dubbo提供了一个服务后，如果消费者没有使用Dubbo也想调用服务，那么这个时候我们就可以让我们的服务支持REST协议，这样消费者就可以通过REST形式调用我们的服务了。

管理台

github地址：<https://github.com/apache/dubbo-admin>

动态配置

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/config-rule/>

注意动态配置修改的是服务**参数**，并不能修改服务的协议、IP、PORT、VERSION、GROUP，因为这5个信息是服务的标识信息，是服务的身份证号，是不能修改的。

服务路由

官网地址：<http://dubbo.apache.org/zh/docs/v2.7/user/examples/routing-rule/>

什么是蓝绿发布、灰度发布


<https://zhuanlan.zhihu.com/p/42671353>

Zookeeper可视化客户端工具

Zookeeper可视化客户端：

 [ZoolInspector.zip](#)

解压后运行：

讲课PPT > zookeeper > ZoolInspector > build >				
名称	修改日期	类型	大小	
classes	2020/3/29 20:28	文件夹		
config	2020/3/29 20:28	文件夹		
icons	2020/3/29 20:28	文件夹		
lib	2020/3/29 20:28	文件夹		
licences	2020/3/29 20:28	文件夹		
test	2020/3/29 20:28	文件夹		
 zookeeper-dev-ZoolInspector.jar	2010/2/21 17:50	Executable Jar File		