**Dubbo服务集群容错配置**

1.Dubbo服务集群部署

以简易版支付系统中的部分服务为例：

用户服务：pay-service-user 交易服务：pay-service-trade

2.Dubbo服务集群容错配置--集群容错模式：

标签：<dubbo:service>、<dubbo:reference>、 <dubbo:consumer>、<dubbo:provider>属性：cluster 类型：string

是否必填：可选

缺省值：failover

作用：性能调优

集群方式：可选：failover/failfast/failsafe/failback/forking

兼容性：2.0.5以上版本

1、Failover Cluster ：失败自动切换

失败自动切换，当出现失败，重试其它服务器。(缺省) 通常用于读操作，但重试会带来更长延迟。可通过retries="2"来设置重试次数(不含第一次)。

<dubbo:service retries="2" /> 或：

<dubbo:referenceretries="2" />

或：

<dubbo:reference>

<dubbo:methodname="findFoo" retries="2" /> </dubbo:reference>

2、Failfas tCluster：快速失败

快速失败，只发起一次调用，失败立即报错。

通常用于非幂等性的写操作，比如新增记录。

<dubbo:service cluster="failfast" /> 或：

<dubbo:referencecluster="failfast" />

3、Failsafe Cluster：失败安全

失败安全，出现异常时，直接忽略。

通常用于写入审计日志等操作。

<dubbo:service cluster="failsafe" /> 或：

<dubbo:referencecluster="failsafe" />

4、FailbackCluster：失败自动恢复

失败自动恢复，后台记录失败请求，定时重发。通常用于消息通知操作。

<dubbo:service cluster="failback" /> 或：

<dubbo:referencecluster="failback" />

5、ForkingCluster：并行调用

并行调用多个服务器，只要一个成功即返回。

通常用于实时性要求较高的读操作，但需要浪费更多服务资源。可通过forks="2"来设置最大并行数。

<dubbo:servicecluster=“forking" />

或：

<dubbo:referencecluster=“forking" />

Dubbo服务集群容错配置--集群容错模式：策略成熟度



3.启动时检查

Dubbo缺省会在启动时检查依赖的服务是否可用，不可用时会抛出异常，阻止Spring初始化完成，以便上线时，能及早发现问题，默认check=true。

如果你的Spring容器是懒加载的，或者通过API编程延迟引用服务，请关闭check，否则服务临时不可用时，会抛出异常，拿到null引用，如果check=false，总是会返回引用，当服务恢复时，能自动连上。

可以通过check="false"关闭检查，比如，测试时，有些服务不关心，或者出现了循环依赖，必须有一方先启动。

关闭某个服务的启动时检查：(没有提供者时报错)

<dubbo:reference interface="com.foo.BarService" check="false" />

关闭所有服务的启动时检查：(没有提供者时报错)

<dubbo:consumer check="false" />

关闭注册中心启动时检查：(注册订阅失败时报错)

<dubbo:registry check="false" />

也可以用dubbo.properties配置：

dubbo.properties

dubbo.reference.com.foo.BarService.check=false

dubbo.reference.check=false

dubbo.consumer.check=false

dubbo.registry.check=false

也可以用-D参数：

java -Ddubbo.reference.com.foo.BarService.check=false

java -Ddubbo.reference.check=false

java -Ddubbo.consumer.check=false

java -Ddubbo.registry.check=false

注意区别

dubbo.reference.check=false，强制改变所有reference的check值，就算配置中有声明，也会被覆盖。

dubbo.consumer.check=false，是设置check的缺省值，如果配置中有显式的声明，如：<dubbo:reference check="true"/>，不会受影响。

dubbo.registry.check=false，前面两个都是指订阅成功，但提供者列表是否为空是否报错，如果注册订阅失败时，也允许启动，需使用此选项，将在后台定时重试。

引用缺省是延迟初始化的，只有引用被注入到其它Bean，或被getBean()获取，才会初始化。

如果需要饥饿加载，即没有人引用也立即生成动态代理，可以配置：

<dubbo:reference interface="com.foo.BarService" init="true" />

---------------------

作者：青鸟&飞鱼

来源：CSDN

原文：https://blog.csdn.net/u014401141/article/details/71307820

版权声明：本文为博主原创文章，转载请附上博文链接！