# [motan-1：优雅关闭](http://confluence.coohua.com/pages/viewpage.action?pageId=6004345)

原创；微信公众号：千里行走；

受限图片大小限制，有些图片不是很清晰，可以到微信公众号查看；

目录

(1).官方demo代码

(2).我们的代码

(3).问题与解决

(4).源码解析

(5).参考资料

正文

## **(1).官方demo代码**

//Motan支持在Consul、ZooKeeper集群环境下优雅的关闭节点，当需要关闭或重启节点时，可以先将待上线节点从集群中摘除，避免直接关闭影响正常请求。

//待关闭节点需要调用以下代码，建议通过servlet或业务的管理模块进行该调用。

MotanSwitcherUtil.setSwitcherValue(MotanConstants.REGISTRY\_HEARTBEAT\_SWITCHER, false)

# **(2).我们的代码**

### web应用

public class ContextListener implements ServletContextListener {

@Override

public void contextInitialized(ServletContextEvent servletContextEvent) {

// 启用 Motan

MotanSwitcherUtil.setSwitcherValue(MotanConstants.REGISTRY\_HEARTBEAT\_SWITCHER, true);

}

@Override

public void contextDestroyed(ServletContextEvent servletContextEvent) {

}

}

### spring-boot应用

@SpringBootApplicationpublic class AccountApplication {

public static void main(String[] args){

SpringApplication.run(AccountApplication.class, args);

MotanSwitcherUtil.setSwitcherValue(MotanConstants.REGISTRY\_HEARTBEAT\_SWITCHER, true);

}

}

**(3).问题与解决**

### 1.问题

在web项目中没有在contextDestroyed的监听中调用motan的销毁反注册钩子，导致motan服务已经下线，但是zk中依然保留服务的注册信息，客户端不能及时感知到服务的下线，导致下线后依然打过来大量的请求，造成客户端阻塞或者报错。

**2.解决方案**

### 针对web程序

在ServletContextListener中contextDestroyed方法中显示调用motan的反注册

@Overridepublic void contextDestroyed(ServletContextEvent servletContextEvent) {

MotanSwitcherUtil.setSwitcherValue(MotanConstants.REGISTRY\_HEARTBEAT\_SWITCHER, false);

TimeUnit.SECONDS.sleep(1);

}

### 针对springboot/普通 java应用

@SpringBootApplicationpublic class AccountApplication {

public static void main(String[] args){

SpringApplication.run(AccountApplication.class, args);

Runtime.getRuntime().addShutdownHook(new Thread() {

public void run() {

try {

MotanSwitcherUtil.setSwitcherValue(MotanConstants.REGISTRY\_HEARTBEAT\_SWITCHER, false);

TimeUnit.SECONDS.sleep(1);

logger.info("motan service has shutdown gracefully!");

} catch (Exception e) {

logger.error("error occured during montan service shutdown! pls check! ", e);

}

}

});

}

}

1. **.源码解析**

### 1.开关及监听

官方推荐我们优雅关闭是推荐这句代码

MotanSwitcherUtil.setSwitcherValue(MotanConstants.REGISTRY\_HEARTBEAT\_SWITCHER, false);

我们来看下Motan向MotanConstants.REGISTRY\_HEARTBEAT\_SWITCHER这个开关项注册了什么监听：

public AbstractRegistry(URL url) {

this.registryUrl = url.createCopy();

// register a heartbeat switcher to perceive service state change and change available state

MotanSwitcherUtil.initSwitcher(MotanConstants.REGISTRY\_HEARTBEAT\_SWITCHER, false);

MotanSwitcherUtil.registerSwitcherListener(MotanConstants.REGISTRY\_HEARTBEAT\_SWITCHER, new SwitcherListener() {

@Override

public void onValueChanged(String key, Boolean value) {

if (key != null && value != null) {

if (value) {

available(null);

} else {

unavailable(null);

}

}

}

});

}

可见在Registry的抽象类中注册了这个开关的监听，当开关值变化后处理相应的动作，如果关闭，则触发unavailable方法，这个方法会注销所有的服务（在registry中删除所有的节点）。

这里我们以zookeeper的实现为例：

@Overrideprotected void doUnavailable(URL url) {

try{

serverLock.lock();

if (url == null) {

availableServices.removeAll(getRegisteredServiceUrls());

for (URL u : getRegisteredServiceUrls()) {

removeNode(u, ZkNodeType.AVAILABLE\_SERVER);

removeNode(u, ZkNodeType.UNAVAILABLE\_SERVER);

createNode(u, ZkNodeType.UNAVAILABLE\_SERVER);

}

} else {

availableServices.remove(url);

removeNode(url, ZkNodeType.AVAILABLE\_SERVER);

removeNode(url, ZkNodeType.UNAVAILABLE\_SERVER);

createNode(url, ZkNodeType.UNAVAILABLE\_SERVER);

}

} finally {

serverLock.unlock();

}

}

监听触发后会注销所有的服务。

1. **.参考资料**
2. **motan source code：**

**https://github.com/weibocom/motan**