在Spring中，@Service默认都是单例的。用了私有全局变量，若不想影响下次请求，就需要用到原型模式，即@Scope(“prototype”)

所谓单例，就是Spring的IOC机制只创建该类的一个实例，每次请求，都会用这同一个实例进行处理，因此若存在全局变量，本次请求的值肯定会影响下一次请求时该变量的值。

原型模式，指的是每次调用时，会重新创建该类的一个实例，比较类似于我们自己自己new的对象实例。

全局变量：

**private** **int** count = 1;

**private** **static** **int** *num* = 1;

**public** **void** testScope() {

System.***out***.println("count:" + count++ + "/t" + "num:" + *num*++);

}

如果是单例，那么每次调用testScope()，count和num都会增加。

如果service层开启@Scope("prototype")，那么有3种启动防止prototype失效方法。

1. Service层添加 @Scope(value = "prototype", proxyMode = ScopedProxyMode.TARGET\_CLASS)
2. Service层添加 @Scope("prototype")，在controller层也要添加此注解
3. Service层添加 @Scope("prototype")，在controller层调用service应用的时候，不用 @autowired 注入，从ApplicationContext获取，ApplicationContext 获取方式这里提供2中方式：
4. 实现ApplicationContextAware接口
5. 自定义监听器ApplicationListener<ApplicationReadyEvent>

启动自定义监听并且注册到容器的方法有两种：

1. 在实现类上加 @Component注解
2. 在启动类里面通过SpringApplication添加监听