1.Bean的Scope

1.Singleton ： 一个Spring容器中只有一个Bean的实例,此为Spring的默认配置,全容器共享一个实例

2.Prototype ： 每次调用新建一个Bean实例

3.Request ： Web项目中,给每一个http request新建一个Bean实例

4.Session ： Web项目中,给每一个http session新建一个Bena实例

5.Globalsession ： 这个只在Portal应用中有用,给每一个global http session新建一个Bean实例

Spring Batch中还有一个Scope是使用@StepScopte

2.Spring EL和资源调用

Spring主要在注解@Value的参数中使用表达式

1. 注入普通字符(@value("普通字符串"))
2. 注入操作系统属性(@Value(#{systemProperties['os.name']}))
3. 注入表达式结果(@Value("#{T(java.lang.Math).random() * 100.0}"))
4. 注入其他bean的属性@Value(#{demoService.another})
5. 注入文件内容@Value("classpath:cn/txx/...../test.txt")
6. 注入网址内容@Value("http://www.baidu.com")
7. 注入配置文件

   需要在类上添加

@PropertySource("classpath:cn/txx/.../test.properties")

   在属性上使用@Value("${book.name}")

   使用@PropertySource指定文件地址,若使用@Value注入,则要配置一个PropertySourcePlaceholderConfigurer的Bean

3.Bean的初始化和销毁

(1) Java配置的方式,使用@Bean的initMethod和destroyMethod(相当于xml配置的init-method和destroy-method)

(2) 注解方式:利用JSR-250的@PostConstruct和@PreDestroy

需要增加jsr250-api.jar的支持

4.Profile

(1) 通过设定Environment的ActiveProfiles来设定当前的context需要使用的配置环境.在开发中使用@Profile注解类或者方法,达到在不同情况下选择实例化不同的Bean

(2) 通过设定jvm的spring.profiles.active参数来设置配置环境

(3) Web项目设置在Servlet的context parameter中

Servlet2.5以及以下

在dispatcher中初始化参数spring.profiles.active

servlet3.0及以上

实现WebApplicationInitializer接口

container.setInitParameter("spring.profiles.default","dev")

5.事件

Spring的事件(Application Event)为Bean与Bean之间的消息通信提供了支持.当一个Bean处理完一个任务后,希望另外一个Bean知道并能做相应的处理,这时我们就需要让另外一个Bean监听当前Bean所发送的事件

Spring的事件需要遵循如下流程

(1) 自定义事件,集成ApplicationEvent(事件类)

(2) 定义事件监听器,实现ApplicationListener(事件监听类)

(3) 使用容器发布事件(事件发布类,使用容器发布事件):applicationContext.publishEvent(new DemoEvent(this.msg))