Dubbo Spring Boot 设计与实现

Spring Boot 基础

自动装配

核心组件

Spring 3.2 引入 org.springframework.core.io.support.SpringFactoriesLoader，在 Spring Framework 中未被广泛使用，但被 Spring Boot 广泛运用，主要用于自动装配。

核心注解 - @EnableAutoConfiguration

`@org.springframework.boot.autoconfigure.EnableAutoConfiguration` 配置在 `META-INF/spring.factories` 文件中。

传统 Spring 配置类 `@Configuration` 的 Class，需要被动被导入或者被扫描，手动装配。

自动装配本质是配置装配。

辅助 Spring 组件

条件注解 - @Conditional 派生

@ConditionalOnBean

@ConditionalOnClass

...

传统注解

- @Import
- @EnableXXX

Spring 事件 - ApplicationEvent 扩展

外部化配置

服务治理（Actuator）

Dubbo Spring Boot

主要特性

- 支持 Spring Boot 特性
- 兼容 Spring Boot 1.x 和 2.x
  - dubbo-spring-boot-autoconfigure-compatible

Spring Boot 自动装配

META-INF/spring.factories 配置文件

```
org.springframework.boot.autoconfigure.EnableAutoConfiguration=\
org.apache.dubbo.spring.boot.autoconfigure.DubboAutoConfiguration,\
org.apache.dubbo.spring.boot.autoconfigure.DubboRelaxedBindingAutoConfiguration
```

自动装配类

org.apache.dubbo.spring.boot.autoconfigure.DubboAutoConfiguration

运用 Dubbo Spring 实现：

- @EnableDubboConfig - 激活 Dubbo 外部化配置
- org.apache.dubbo.config.spring.beans.factory.annotation.ServiceClassPostProcessor
- org.apache.dubbo.config.spring.context.DubboBootstrapApplicationListener
- org.apache.dubbo.config.spring.context.DubboLifecycleComponentApplicationListener

org.apache.dubbo.spring.boot.autoconfigure.DubboRelaxedBindingAutoConfiguration

兼容 Spring Boot 1.x 松散外部化配置绑定特性

激活条件 - 需要 ClassLoader 能够加载 ”org.springframework.boot.bind.RelaxedPropertyResolver“

org.apache.dubbo.spring.boot.autoconfigure.DubboRelaxedBinding2AutoConfiguration

兼容 Spring Boot 2.x 松散外部化配置绑定特性

激活条件 - 需要 ClassLoader 能够加载 ”org.springframework.boot.context.properties.bind.Binder“

Dubbo Spring 相关

Dubbo 外部化配置前缀：

`dubbo.*`

@org.apache.dubbo.config.spring.context.annotation.EnableDubboConfig

辅助注解：

- @com.alibaba.spring.beans.factory.annotation.EnableConfigurationBeanBindings
- @com.alibaba.spring.beans.factory.annotation.EnableConfigurationBeanBinding

```
# 普通风格
dubbo.config-center.highestPriority  = true

# 标准风格
dubbo.config-center.highest-priority = true

# 环境变量风格
dubbo.config-center.HIGHEST_PRIORITY = true
```

默认情况，仅识别 Spring 标准方式，即 dubbo.config-center.highestPriority

借助于 Spring Boot 1.x 或 2.x 实现

com.alibaba.spring.beans.factory.annotation.EnableConfigurationBeanBinding

底层实现：
com.alibaba.spring.beans.factory.annotation.ConfigurationBeanBindingRegistrar

实际处理实现：
com.alibaba.spring.beans.factory.annotation.ConfigurationBeanBindingPostProcessor

将绑定的配置属性前缀与目标配置 Bean 绑定，比如：

"dubbo.config-center" ->
org.apache.dubbo.config.spring.ConfigCenterBean

com.alibaba.spring.context.config.ConfigurationBeanBinder 实现来解决配置风格不同实现

com.alibaba.spring.context.config.ConfigurationBeanBinder

辅助 Spring 外部化配置组件适配 Spring 、Spring Boot 1.x 以及 Spring Boot 2.x 的实现差异

- Dubbo Spring 实现 - com.alibaba.spring.context.config.DefaultConfigurationBe

anBinder
- Dubbo Spring Boot 1.x 实现 - org.apache.dubbo.spring.boot.autoconfigure.RelaxedDubboConfigBinder
- Dubbo Spring Boot 2.x 实现 - org.apache.dubbo.spring.boot.autoconfigure.BinderDubboConfigBinder

Spring Boot 开发技巧

自动装配开发技巧

1. 尽可能复用 Spring 中的实现，让 Spring Boot 最小化使用
2. 尽可能使用 Spring 内部 API 扩展

作业

1. 将上次 MyBatis @Enable 模块驱动，封装成 Spring Boot Starter 方式

   参考：MyBatis Spring Project 里面会有 Spring Boot 实现