# 第十章：Spring 配置元信息

小马哥 • mercyblitz

# Spring 配置元信息

1. Spring 配置元信息
2. Spring Bean 配置元信息
3. Spring Bean 属性元信息
4. Spring 容器配置元信息
5. 基于XML文件装载Spring Bean配置元信息
6. 基于Properties文件装载Spring Bean配置元信息
7. 基于Java注解装载Spring Bean配置元信息
8. Spring Bean 配置元信息底层实现
9. 基于XML文件装载 Spring IoC 容器配置元信息
10. 基于Java注解装载 Spring IoC 容器配置元信息

# Spring 配置元信息

11. 基于 Extensible XML authoring 扩展Spring XML元素
12. Extensible XML authoring扩展原理
13. 基于Properties文件装载外部化配置
14. 基于YAML文件装载外部化配置
15. 面试题

# Spring 配置元信息

- 配置元信息
  - Spring Bean 配置元信息 - BeanDefinition
  - Spring Bean 属性元信息 - PropertyValues
  - Spring 容器配置元信息
  - Spring 外部化配置元信息 - PropertySource
  - Spring Profile 元信息 - @Profile

# Spring Bean 配置元信息

- Bean 配置元信息 - BeanDefinition
  - GenericBeanDefinition：通用型 BeanDefinition
  - RootBeanDefinition：无 Parent 的 BeanDefinition 或者合并后 BeanDefinition
  - AnnotatedBeanDefinition：注解标注的 BeanDefinition

# Spring Bean 属性元信息

- Bean 属性元信息 - PropertyValues
    - 可修改实现 - MutablePropertyValues
    - 元素成员 - PropertyValue
- Bean 属性上下文存储 - AttributeAccessor
- Bean 元信息元素 - BeanMetadataElement

# Spring 容器配置元信息

- Spring XML 配置元信息 - beans 元素相关

| beans 元素属性 | 默认值 | 使用场景 |
| --- | --- | --- |
| profile | null（留空） | Spring Profiles 配置值 |
| default-lazy-init | default | 当 outter beans “default-lazy-init” 属性存在时，继承该值，否则为“false” |
| default-merge | default | 当 outter beans “default-merge” 属性存在时，继承该值，否则为“false” |
| default-autowire | default | 当 outter beans “default-autowire” 属性存在时，继承该值，否则为“no” |
| default-autowire-candidates | null（留空） | 默认 Spring Beans 名称 pattern |
| default-init-method | null（留空） | 默认 Spring Beans 自定义初始化方法 |
| default-destroy-method | null（留空） | 默认 Spring Beans 自定义销毁方法 |

# Spring 容器配置元信息

- Spring XML 配置元信息 - 应用上下文相关

| XML 元素 | 使用场景 |
| --- | --- |
| <context:annotation-config /> | 激活 Spring 注解驱动 |
| <context:component-scan /> | Spring @Component 以及自定义注解扫描 |
| <context:load-time-weaver /> | 激活 Spring LoadTimeWeaver |
| <context:mbean-export /> | 暴露 Spring Beans 作为 JMX Beans |
| <context:mbean-server /> | 将当前平台作为 MBeanServer |
| <context:property-placeholder /> | 加载外部化配置资源作为 Spring 属性配置 |
| <context:property-override /> | 利用外部化配置资源覆盖 Spring 属性值 |

# 基于 XML 资源装载 Spring Bean 配置元信息

- Spring Bean 配置元信息

| XML 元素 | 使用场景 |
| --- | --- |
| <beans:beans /> | 单 XML 资源下的多个 Spring Beans 配置 |
| <beans:bean /> | 单个 Spring Bean 定义（BeanDefinition）配置 |
| <beans:alias /> | 为 Spring Bean 定义（BeanDefinition）映射别名 |
| <beans:import /> | 加载外部 Spring XML 配置资源 |

底层实现 - XmlBeanDefinitionReader

# 基于 Properties 资源装载 Spring Bean 配置元信息

- Spring Bean 配置元信息

| Properties 属性名 | 使用场景 |
| --- | --- |
| (class) | Bean 类全称限定名 |
| (abstract) | 是否为抽象的 BeanDefinition |
| (parent) | 指定 parent BeanDefinition 名称 |
| (lazy-init) | 是否为延迟初始化 |

底层实现 - PropertiesBeanDefinitionReader

# 基于 Properties 资源装载 Spring Bean 配置元信息

- Spring Bean 配置元信息

| Properties 属性名 | 使用场景 |
| --- | --- |
| (ref) | 引用其他 Bean 的名称 |
| (scope) | 设置 Bean 的 scope 属性 |
| ${n} | n 表示第 n+1 个构造器参数 |
| | |

底层实现 - PropertiesBeanDefinitionReader

# 基于 Java 注解装载 Spring Bean 配置元信息

- Spring 模式注解

| Spring 注解 | 场景说明 | 起始版本 |
| --- | --- | --- |
| @Repository | 数据仓储模式注解 | 2.0 |
| @Component | 通用组件模式注解 | 2.5 |
| @Service | 服务模式注解 | 2.5 |
| @Controller | Web 控制器模式注解 | 2.5 |
| @Configuration | 配置类模式注解 | 3.0 |

内容来源于《Spring Boot 编程思想（核心篇）》 - “Spring 核心注解场景分类” 章节

# 基于 Java 注解装载 Spring Bean 配置元信息

- Spring Bean 定义注解

| Spring 注解 | 场景说明 | 起始版本 |
| --- | --- | --- |
| @Bean | 替换 XML 元素 <bean> | 3.0 |
| @DependsOn | 替代 XML 属性 <bean depends-on="..."/> | 3.0 |
| @Lazy | 替代 XML 属性 <bean lazy-init="true\|falses" /> | 3.0 |
| @Primary | 替换 XML 元素 <bean primary="true\|false" /> | 3.0 |
| @Role | 替换 XML 元素 <bean role="..." /> | 3.1 |
| @Lookup | 替代 XML 属性 <bean lookup-method="..."> | 4.1 |

内容来源于《Spring Boot 编程思想（核心篇）》 - “Spring 核心注解场景分类” 章节

# 基于 Java 注解装载 Spring Bean 配置元信息

- Spring Bean 依赖注入注解

| Spring 注解 | 场景说明 | 起始版本 |
| --- | --- | --- |
| @Autowired | Bean 依赖注入，支持多种依赖查找方式 | 2.5 |
| @Qualifier | 细粒度的 @Autowired 依赖查找 | 2.5 |

| Java 注解 | 场景说明 | 起始版本 |
| --- | --- | --- |
| @Resource | 类似于 @Autowired | 2.5 |
| @Inject | 类似于 @Autowired | 2.5 |

内容来源于《Spring Boot 编程思想（核心篇）》 - “Spring 核心注解场景分类” 章节

# 基于 Java 注解装载 Spring Bean 配置元信息

- Spring Bean 条件装配注解

| Spring 注解 | 场景说明 | 起始版本 |
| --- | --- | --- |
| @Profile | 配置化条件装配 | 3.1 |
| @Conditional | 编程条件装配 | 4.0 |

内容来源于《Spring Boot 编程思想（核心篇）》 - “Spring 核心注解场景分类” 章节

# 基于 Java 注解装载 Spring Bean 配置元信息

- Spring Bean 生命周期回调注解

| Spring 注解 | 场景说明 | 起始版本 |
| --- | --- | --- |
| @PostConstruct | 替换 XML 元素 <bean init-method="..." /> 或 InitializingBean | 2.5 |
| @PreDestroy | 替换 XML 元素 <bean destroy-method="..." /> 或 DisposableBean | 2.5 |

内容来源于《Spring Boot 编程思想（核心篇）》 - “Spring 核心注解场景分类” 章节

# Spring Bean 配置元信息底层实现

- Spring BeanDefinition 解析与注册

| 实现场景 | 实现类 | 起始版本 |
| --- | --- | --- |
| XML 资源 | XmlBeanDefinitionReader | 1.0 |
| Properties 资源 | PropertiesBeanDefinitionReader | 1.0 |
| Java 注解 | AnnotatedBeanDefinitionReader | 3.0 |

# Spring Bean 配置元信息底层实现

- Spring XML 资源 BeanDefinition 解析与注册
  - 核心 API - XmlBeanDefinitionReader
    - 资源 - Resource
    - 底层 - BeanDefinitionDocumentReader
      - XML 解析 - Java DOM Level 3 API
      - BeanDefinition 解析 - BeanDefinitionParserDelegate
      - BeanDefinition 注册 - BeanDefinitionRegistry

# Spring Bean 配置元信息底层实现

- Spring Properties 资源 BeanDefinition 解析与注册
  - 核心 API - PropertiesBeanDefinitionReader
    - 资源
      - 字节流 - Resource
      - 字符流 - EncodedResouce
    - 底层
      - 存储 - java.util.Properties
      - BeanDefinition 解析 - API 内部实现
      - BeanDefinition 注册 - BeanDefinitionRegistry

# Spring Bean 配置元信息底层实现

- Spring Java 注册 BeanDefinition 解析与注册
  - 核心 API - AnnotatedBeanDefinitionReader
    - 资源
      - 类对象 - java.lang.Class
    - 底层
      - 条件评估 - ConditionEvaluator
      - Bean 范围解析 - ScopeMetadataResolver
      - BeanDefinition 解析 - 内部 API 实现
      - BeanDefinition 处理 - AnnotationConfigUtils.processCommonDefinitionAnnotations
      - BeanDefinition 注册 - BeanDefinitionRegistry

# 基于 XML 资源装载 Spring IoC 容器配置元信息

- Spring IoC 容器相关 XML 配置

| 命名空间 | 所属模块 | Schema 资源 URL |
| --- | --- | --- |
| beans | spring-beans | https://www.springframework.org/schema/beans/spring-beans.xsd |
| context | spring-context | https://www.springframework.org/schema/context/spring-context.xsd |
| aop | spring-aop | https://www.springframework.org/schema/aop/spring-aop.xsd |
| tx | spring-tx | https://www.springframework.org/schema/tx/spring-tx.xsd |
| util | spring-beans | https://www.springframework.org/schema/util/spring-util.xsd |
| tool | spring-beans | https://www.springframework.org/schema/tool/spring-tool.xsd |

# 基于 Java 注解装载 Spring IoC 容器配置元信息

- Spring IoC 容器装配注解

| Spring 注解 | 场景说明 | 起始版本 |
| --- | --- | --- |
| @ImportResource | 替换 XML 元素 <import> | 3.0 |
| @Import | 导入 Configuration Class | 3.0 |
| @ComponentScan | 扫描指定 package 下标注 Spring 模式注解的类 | 3.1 |

内容来源于《Spring Boot 编程思想（核心篇）》 - “Spring 核心注解场景分类” 章节

# 基于 Java 注解装载 Spring IoC 容器配置元信息

- Spring IoC 配属属性注解

| Spring 注解 | 场景说明 | 起始版本 |
| --- | --- | --- |
| @PropertySource | 配置属性抽象 PropertySource 注解 | 3.1 |
| @PropertySources | @PropertySource 集合注解 | 4.0 |

内容来源于《Spring Boot 编程思想（核心篇）》 - “Spring 核心注解场景分类” 章节

# 基于 Extensible XML authoring 扩展 Spring XML 元素

- Spring XML 扩展
  - 编写 XML Schema 文件：定义 XML 结构
  - 自定义 NamespaceHandler 实现：命名空间绑定
  - 自定义 BeanDefinitionParser 实现：XML 元素与 BeanDefinition 解析
  - 注册 XML 扩展：命名空间与 XML Schema 映射

# Extensible XML authoring 扩展原理

- 触发时机
    - AbstractApplicationContext#obtainFreshBeanFactory
        - AbstractRefreshableApplicationContext#refreshBeanFactory
            - AbstractXmlApplicationContext#loadBeanDefinitions
                - ...
                    - XmlBeanDefinitionReader#doLoadBeanDefinitions
                        - ...
                            - BeanDefinitionParserDelegate#parseCustomElement

# Extensible XML authoring 扩展原理

- 核心流程
  - BeanDefinitionParserDelegate#parseCustomElement(org.w3c.dom.Element, BeanDefinition)
    - 获取 namespace
    - 通过 namespace 解析 NamespaceHandler
    - 构造 ParserContext
    - 解析元素，获取  BeanDefinintion

# 基于 Properties 资源装载外部化配置

- 注解驱动
  - @org.springframework.context.annotation.PropertySource
  - @org.springframework.context.annotation.PropertySources
- API 编程
  - org.springframework.core.env.PropertySource
  - org.springframework.core.env.PropertySources

详细讨论将在《Spring Environment 抽象》 中展开

# 基于 YAML 资源装载外部化配置

- API 编程
  - org.springframework.beans.factory.config.YamlProcessor
    - org.springframework.beans.factory.config.YamlMapFactoryBean
    - org.springframework.beans.factory.config.YamlPropertiesFactoryBean

# 面试题


**沙雕面试题** - Spring 内建 XML Schema 常见有哪些?

答:

| 命名空间 | 所属模块 | Schema 资源 URL |
| --- | --- | --- |
| beans | spring-beans | https://www.springframework.org/schema/beans/spring-beans.xsd |
| context | spring-context | https://www.springframework.org/schema/context/spring-context.xsd |
| aop | spring-aop | https://www.springframework.org/schema/aop/spring-aop.xsd |
| tx | spring-tx | https://www.springframework.org/schema/tx/spring-tx.xsd |
| util | spring-beans | https://www.springframework.org/schema/util/spring-util.xsd |
| tool | spring-beans | https://www.springframework.org/schema/tool/spring-tool.xsd |

# 面试题

## 996 面试题 - Spring配置元信息具体有哪些？

答：

- Bean 配置元信息：通过媒介（如 XML、Proeprties 等），解析 BeanDefinition
- IoC 容器配置元信息：通过媒介（如 XML、Proeprties 等），控制 IoC 容器行为，比如注解驱动、AOP 等
- 外部化配置：通过资源抽象（如 Proeprties、YAML 等），控制 PropertySource
- Spring Profile：通过外部化配置，提供条件分支流程

# 面试题

**劝退面试题** - Extensible XML authoring 的缺点？

答：

- 高复杂度：开发人员需要熟悉 XML Schema，spring.handlers，spring.schemas 以及 Spring API
- 嵌套元素支持较弱：通常需要使用方法递归或者其嵌套解析的方式处理嵌套（子）元素
- XML 处理性能较差：Spring XML 基于 DOM Level 3 API 实现，该 API 便于理解，然而性能较差
- XML 框架移植性差：很难适配高性能和便利性的 XML 框架，如 JAXB