

# 第十九章：Spring Environment 抽象

小马哥 · mercyblitz



扫码试看/订阅

《小马哥讲 Spring 核心编程思想》视频课程

# Spring Environment 抽象

---

1. 理解 Spring Environment 抽象
2. Spring Environment 接口使用场景
3. Environment 占位符处理
4. 理解条件配置 Spring Profiles
5. Spring 4 重构 @Profile
6. 依赖注入 Environment
7. 依赖查找 Environment
8. 依赖注入 @Value
9. Spring 类型转换在 Environment 中的运用
10. Spring 类型转换在 @Value 中的运用



# Spring Environment 抽象

---

- 11. Spring 配置属性源 PropertySource
- 12. Spring 內建的配置属性源
- 13. 基于注解扩展 Spring 配置属性源
- 14. 基于 API 扩展 Spring 配置属性源
- 15. 课外资料
- 16. 面试题精选



# 理解 Spring Environment 抽象

- 统一的 Spring 配置属性管理

Spring Framework 3.1 开始引入 Environment 抽象，它统一 Spring 配置属性的存储，包括占位符处理和类型转换，不仅完整地替换 PropertyPlaceholderConfigurer，而且还支持更丰富的配置属性源（PropertySource）

- 条件化 Spring Bean 装配管理

通过 Environment Profiles 信息，帮助 Spring 容器提供条件化地装配 Bean

## Spring Environment 接口使用场景

- 用于属性占位符处理
- 用于转换 Spring 配置属性类型
- 用于存储 Spring 配置属性源（PropertySource）
- 用于 Profiles 状态的维护

# Environment 占位符处理

- Spring 3.1 前占位符处理
  - 组件: `org.springframework.beans.factory.config.PropertyPlaceholderConfigurer`
  - 接口: `org.springframework.util.StringValueResolver`
- Spring 3.1 + 占位符处理
  - 组件: `org.springframework.context.support.PropertySourcesPlaceholderConfigurer`
  - 接口: `org.springframework.beans.factory.config.EmbeddedValueResolver`

# 理解条件配置 Spring Profiles

- Spring 3.1 条件配置
  - API: `org.springframework.core.env.ConfigurableEnvironment`
    - 修改: `addActiveProfile(String)`、`setActiveProfiles(String...)` 和 `setDefaultProfiles(String...)`
    - 获取: `getActiveProfiles()` 和 `getDefaultProfiles()`
    - 匹配: `#acceptsProfiles(String...)` 和 `acceptsProfiles(Profiles)`
  - 注解: `@org.springframework.context.annotation.Profile`



## Spring 4 重构 @Profile

- 基于 Spring 4 `org.springframework.context.annotation.Condition` 接口实现
  - `org.springframework.context.annotation.ProfileCondition`

# 依赖注入 Environment

- 直接依赖注入
  - 通过 EnvironmentAware 接口回调
  - 通过 @Autowired 注入 Environment
- 间接依赖注入
  - 通过 ApplicationContextAware 接口回调
  - 通过 @Autowired 注入 ApplicationContext

## 依赖查找 Environment

- 直接依赖查找
  - 通过 `org.springframework.context.ConfigurableApplicationContext#ENVIRONMENT_BEAN_NAME`
- 间接依赖查找
  - 通过 `org.springframework.context.ConfigurableApplicationContext#getEnvironment`

## 依赖注入 @Value

- 通过注入 @Value

- 实现 -

`org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor`

# Spring 类型转换在 Environment 中的运用

- Environment 底层实现
  - 底层实现 - `org.springframework.core.env.PropertySourcesPropertyResolver`
    - 核心方法 - `convertValueIfNecessary(Object, Class)`
  - 底层服务 - `org.springframework.core.convert.ConversionService`
    - 默认实现 - `org.springframework.core.convert.support.DefaultConversionService`

# Spring 类型转换在 @Value 中的运用

- @Value 底层实现
  - 底层实现 - org.springframework.beans.factory.annotation.AutowiredAnnotationBeanPostProcessor
    - org.springframework.beans.factory.support.DefaultListableBeanFactory#doResolveDependency
  - 底层服务 - org.springframework.beans.TypeConverter
    - 默认实现 - org.springframework.beans.TypeConverterDelegate
      - java.beans.PropertyEditor
      - org.springframework.core.convert.ConversionService

# Spring 配置属性源 PropertySource

- API
  - 单配置属性源 - `org.springframework.core.env.PropertySource`
  - 多配置属性源 - `org.springframework.core.env.PropertySources`
- 注解
  - 单配置属性源 - `@org.springframework.context.annotation.PropertySource`
  - 多配置属性源 - `@org.springframework.context.annotation.PropertySources`
- 关联
  - 存储对象 - `org.springframework.core.env.MutablePropertySources`
  - 关联方法 - `org.springframework.core.env.ConfigurableEnvironment#getPropertySources()`

## Spring 內建的配置属性源

- 內建 PropertySource

PropertySource 类型	说明
org.springframework.core.env.CommandLinePropertySource	命令行配置属性源
org.springframework.jndi.JndiPropertySource	JNDI 配置属性源
org.springframework.core.env.PropertiesPropertySource	Properties 配置属性源
org.springframework.web.context.support.ServletConfigPropertySource	Servlet 配置属性源
org.springframework.web.context.support.ServletContextPropertySource	ServletContext 配置属性源
org.springframework.core.env.SystemEnvironmentPropertySource	环境变量配置属性源
.....	



## 基于注解扩展 Spring 配置属性源

- @org.springframework.context.annotation.PropertySource 实现原理
  - 入口 - org.springframework.context.annotation.ConfigurationClassParser#doProcessConfigurationClass
    - org.springframework.context.annotation.ConfigurationClassParser#processPropertySource
  - 4.3 新增语义
    - 配置属性字符编码 - encoding
    - org.springframework.core.io.support.PropertySourceFactory
  - 适配对象 - org.springframework.core.env.CompositePropertySource

## 基于 API 扩展 Spring 配置属性源

- Spring 应用上下文启动前装配 PropertySource
- Spring 应用上下文启动后装配 PropertySource

## 课外资料

- Spring 4.1 测试配置属性源 - @TestPropertySource

## 面试题

### 沙雕面试题 - 简单介绍 Spring Environment 接口?

答:

- 核心接口 - `org.springframework.core.env.Environment`
- 父接口 - `org.springframework.core.env.PropertyResolver`
- 可配置接口 - `org.springframework.core.env.ConfigurableEnvironment`
- 职责:
  - 管理 Spring 配置属性源
  - 管理 Profiles



我真的没笑

## 面试题

996 面试题 - 如何控制 PropertySource 的优先级?

答：代码演示

EnvironmentPropertySourceChangeDemo



## 面试题

**劝退面试题** - Environment 完整的生命周期是怎样的？



答：答案下章揭晓



扫码试看/订阅

《小马哥讲 Spring 核心编程思想》视频课程