

# 第六章：Spring IoC 注入

小马哥 • mercyblitz

# Spring IoC 依赖注入

---

1. 依赖注入的模式和类型
2. 自动绑定 (Autowiring)
3. 自动绑定 (Autowiring) 模式
4. 自动绑定 (Autowiring) 限制和不足
5. Setter 方法依赖注入
6. 构造器依赖注入
7. 字段注入
8. 方法注入
9. 回调注入
10. 依赖注入类型选择



# Spring IoC 依赖注入

---

- 11. 基础类型注入
- 12. 集合类型注入
- 13. 限定注入
- 14. 延迟依赖注入
- 15. 依赖处理过程
- 16. @Autowired 注入原理
- 17. JSR-330 @Inject 注入原理
- 18. Java通用注解注入原理
- 19. 自定义依赖注入注解
- 20. 面试题精选



# 依赖注入的模式和类型

- 手动模式 - 配置或者编程的方式，提前安排注入规则
  - XML 资源配置元信息
  - Java 注解配置元信息
  - API 配置元信息
- 自动模式 - 实现方提供依赖自动关联的方式，按照内建的注入规则
  - Autowiring（自动绑定）

# 依赖注入的模式和类型

- 依赖注入类型

依赖注入类型	配置元数据举例
Setter 方法	<code>&lt;proeprty name=" user" ref=" userBean" /&gt;</code>
构造器	<code>&lt;constructor-arg name="user" ref="userBean" /&gt;</code>
字段	<code>@Autowired User user;</code>
方法	<code>@Autowired public void user(User user) { ... }</code>
接口回调	<code>class MyBean implements BeanFactoryAware { ... }</code>

# 自动绑定 (Autowiring)

- 官方说明

The Spring container can **autowire** relationships between collaborating beans. You can let Spring resolve collaborators (other beans) automatically for your bean by inspecting the contents of the `ApplicationContext`.

- 优点

- Autowiring can significantly reduce the need to **specify** properties or constructor arguments.
- Autowiring can **update** a configuration as your objects evolve.

# 自动绑定（Autowiring）模式

- Autowiring modes

模式	说明
no	默认值，未激活 Autowiring，需要手动指定依赖注入对象
byName	根据被注入属性的名称作为 Bean 名称进行依赖查找，并将对象设置到该属性
byType	根据被注入属性的类型作为依赖类型进行查找，并将对象设置到该属性
constructor	特殊 byType 类型，用于构造器参数

参考枚举：`org.springframework.beans.factory.annotation.Autowired`

# 自动绑定（Autowiring）限制和不足

- 官方说明

Limitations and Disadvantages of Autowiring 小节

链接: <https://docs.spring.io/spring/docs/5.2.2.RELEASE/spring-framework-reference/core.html#beans-autowired-exceptions>



# Setter 方法注入

- 实现方法
  - 手动模式
    - XML 资源配置元信息
    - Java 注解配置元信息
    - API 配置元信息
  - 自动模式
    - byName
    - byType

# 构造器注入

- 实现方法
  - 手动模式
    - XML 资源配置元信息
    - Java 注解配置元信息
    - API 配置元信息
  - 自动模式
    - constructor

# 字段注入

- 实现方法
  - 手动模式
    - Java 注解配置元信息
      - @Autowired
      - @Resource
      - @Inject（可选）

# 方法注入

- 实现方法
  - 手动模式
    - Java 注解配置元信息
      - @Autowired
      - @Resource
      - @Inject（可选）
      - @Bean

# 接口回调注入

- Aware 系列接口回调
  - 自动模式

内建接口	说明
BeanFactoryAware	获取 IoC 容器 - BeanFactory
ApplicationContextAware	获取 Spring 应用上下文 - ApplicationContext 对象
EnvironmentAware	获取 Environment 对象
ResourceLoaderAware	获取资源加载器 对象 - ResourceLoader
BeanClassLoaderAware	获取加载当前 Bean Class 的 ClassLoader
BeanNameAware	获取当前 Bean 的名称

# 接口回调注入

- Aware 系列接口回调（续）
  - 自动模式

内建接口	说明
MessageSourceAware	获取 MessageSource 对象，用于 Spring 国际化
ApplicationEventPublisherAware	获取 ApplicationEventPublishAware 对象，用于 Spring 事件
EmbeddedValueResolverAware	获取 StringValueResolver 对象，用于占位符处理

# 依赖注入类型选择

- 注入选型
  - 低依赖：构造器注入
  - 多依赖：Setter 方法注入
  - 便利性：字段注入
  - 声明类：方法注入

# 基础类型注入

- 基础类型
  - 原生类型 (Primitive) : boolean、byte、char、short、int、float、long、double
  - 标量类型 (Scalar) : Number、Character、Boolean、Enum、Locale、Charset、Currency、Properties、UUID
  - 常规类型 (General) : Object、String、TimeZone、Calendar、Optional 等
  - Spring 类型: Resource、InputSource、Formatter 等



# 集合类型注入

- 集合类型
  - 数组类型（Array）：原生类型、标量类型、常规类型、Spring 类型
  - 集合类型（Collection）
    - Collection: List、Set（SortedSet、NavigableSet、EnumSet）
    - Map: Properties

# 限定注入

- 使用注解 `@Qualifier` 限定
  - 通过 Bean 名称限定
  - 通过分组限定
- 基于注解 `@Qualifier` 扩展限定
  - 自定义注解 - 如 Spring Cloud `@LoadBalanced`

# 延迟依赖注入

- 使用 API `ObjectFactory` 延迟注入
  - 单一类型
  - 集合类型
- 使用 API `ObjectProvider` 延迟注入（推荐）
  - 单一类型
  - 集合类型

# 依赖处理过程

- 基础知识
  - 入口 - `DefaultListableBeanFactory#resolveDependency`
  - 依赖描述符 - `DependencyDescriptor`
  - 自定绑定候选对象处理器 - `AutowireCandidateResolver`

# @Autowired 注入

- @Autowired 注入规则
  - 非静态字段
  - 非静态方法
  - 构造器

# @Autowired 注入

- @Autowired 注入过程
  - 元信息解析
  - 依赖查找
  - 依赖注入（字段、方法）

## @Inject 注入

- @Inject 注入过程

- 如果"果"t 果"果"果"t 果"果"果"t m eeeeeee eee

# Java通用注解注入原理

- CommonAnnotationBeanPostProcessor
  - 注入注解
    - javax.xml.ws.WebServiceRef
    - javax.ejb.EJB
    - javax.annotation.Resource
  - 生命周期注解
    - javax.annotation.PostConstruct
    - javax.annotation.PreDestroy



# 自定义依赖注入注解

- 基于 `AutowiredAnnotationBeanPostProcessor` 实现
- 自定义实现
  - 生命周期处理
    - `InstantiationAwareBeanPostProcessor`
    - `MergedBeanDefinitionPostProcessor`
  - 元数据
    - `InjectedElement`
    - `InjectionMetadata`

# 面试题

沙雕面试题 – 有多少种依赖注入的方式？

答：构造器注入  
Setter 注入  
字段注入  
方法注入  
接口回调注入



# 面试题

996 面试题 – 你偏好构造器注入还是 Setter 注入？



答：两种依赖注入的方式均可使用，如果是必须依赖的话，那么推荐使用构造器注入，Setter 注入用于可选依赖。

# 面试题

## 劝退面试题 – Spring



答：答案将《Spring IoC依赖来源》章节中继续讨论。