

# Spring源码解析

Spring、SpringMVC、SpringBoot、案例解析



班主任：孙艺萌



辅导老师：李小奎



主讲老师：雷丰阳

- 1 Spring 源码篇
- 2 SpringMVC 源码篇
- 3 SpringBoot 源码篇
- 4 面试问题解答与案例分析篇

## Spring 源码篇

✎ 核心注解

✎ 整体架构

✎ 源码解析

## 核心注解

注解	功能
@Bean	容器中注册组件
@Primary	同类组件如果有多个，标注主组件
@DependsOn	组件之间声明依赖关系
@Lazy	组件懒加载（最后使用的时候才创建）
@Scope	声明组件的作用范围(SCOPE_PROTOTYPE,SCOPE_SINGLETON)
@Configuration	声明这是一个配置类，替换以前配置文件
@Component	@Controller、@Service、@Repository
@Indexed	加速注解，所有标注了 @Indexed 的组件，直接会启动快速加载
@Order	数字越小优先级越高，越先工作
@ComponentScan	包扫描
@Conditional	条件注入
@Import	导入第三方jar包中的组件，或定制批量导入组件逻辑



注解	功能
@ImportResource	导入以前的xml配置文件，让其生效
@Profile	基于多环境激活
@PropertySource	外部properties配置文件和JavaBean进行绑定.结合ConfigurationProperties
@PropertySources	@PropertySource组合注解
@Autowired	自动装配
@Qualifier	精确指定
@Value	取值、计算机环境变量、JVM系统。xxxx。@Value("\${xx}")
@Lookup	单例组件依赖非单例组件，非单例组件获取需要使用方法

注：@Indexed 需要引入依赖

```
<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context-indexer</artifactId>
    <optional>true</optional>
</dependency>
```

不会的小伙伴，请去 <https://www.bilibili.com/video/BV1gW411W7wy?p=1>，观看 1-25 集即可

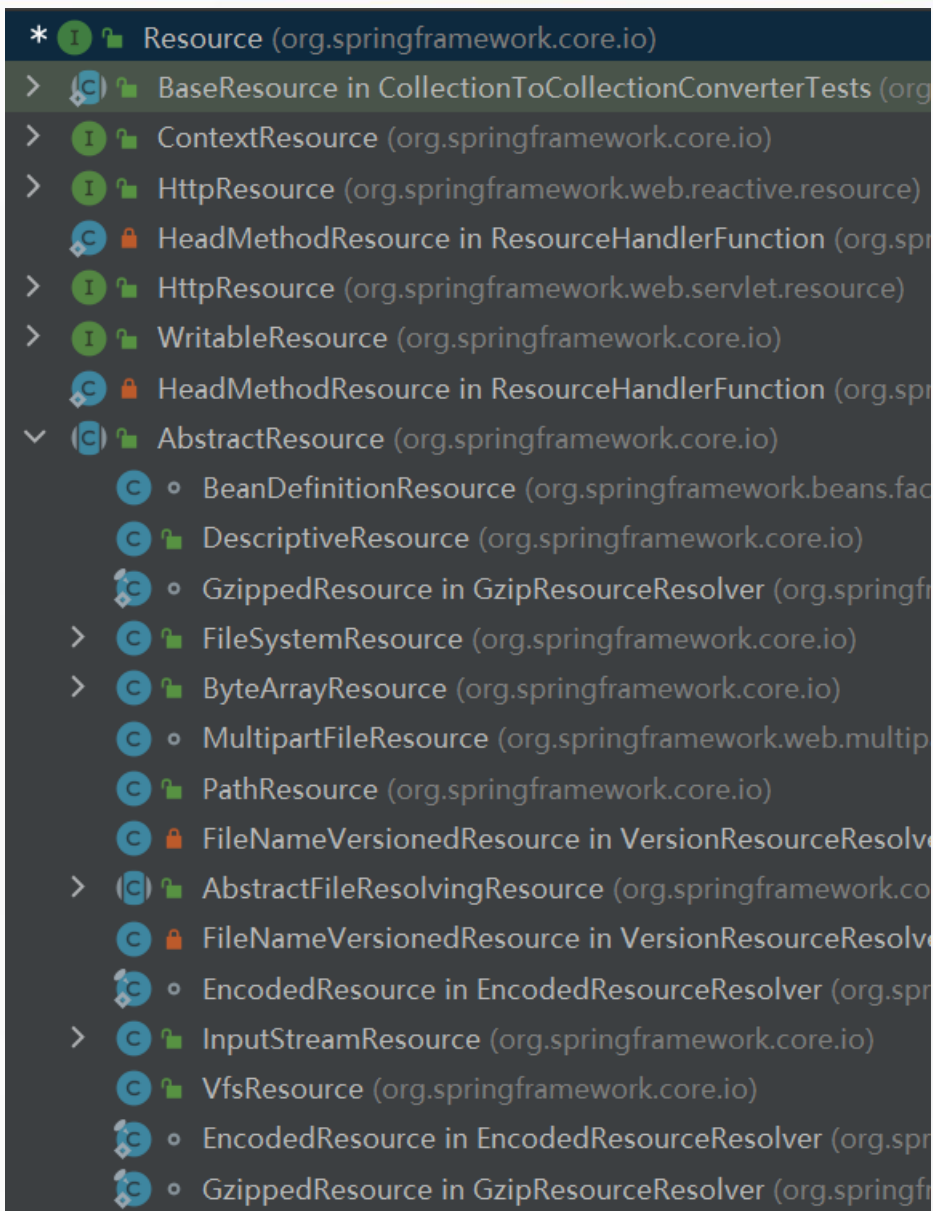
## 整体架构

- 基础接口
  - Resource+ResourceLoader
  - BeanFactory
  - BeanDefinition
  - BeanDefinitionReader
  - BeanDefinitionRegistry
  - SingletonBeanRegistry
  - ApplicationContext
  - Aware
- 生命周期-后置处理器
  - BeanFactoryPostProcessor
  - InitializingBean
  - BeanPostProcessor
  - SmartInitializingSingleton



画一张整体架构工作图

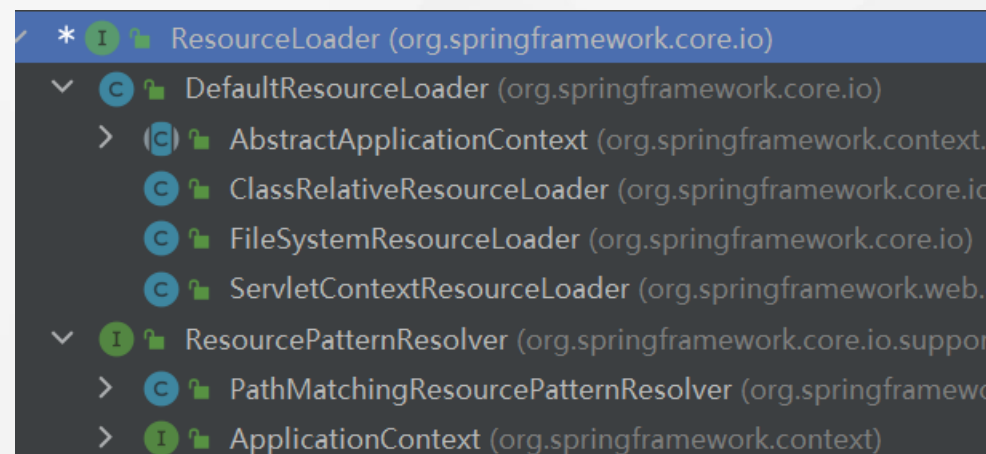
# 核心组件接口分析-Resource



AbstractApplicationContext  
环境类引用loader策略

资源加载策略  
resource loader

资源加载策略  
实现



策略模式

```
ResourceLoader {  
    Resource getResource(String location);  
}
```

### - HierarchicalBeanFactory: 定义父子工厂（父子容器）

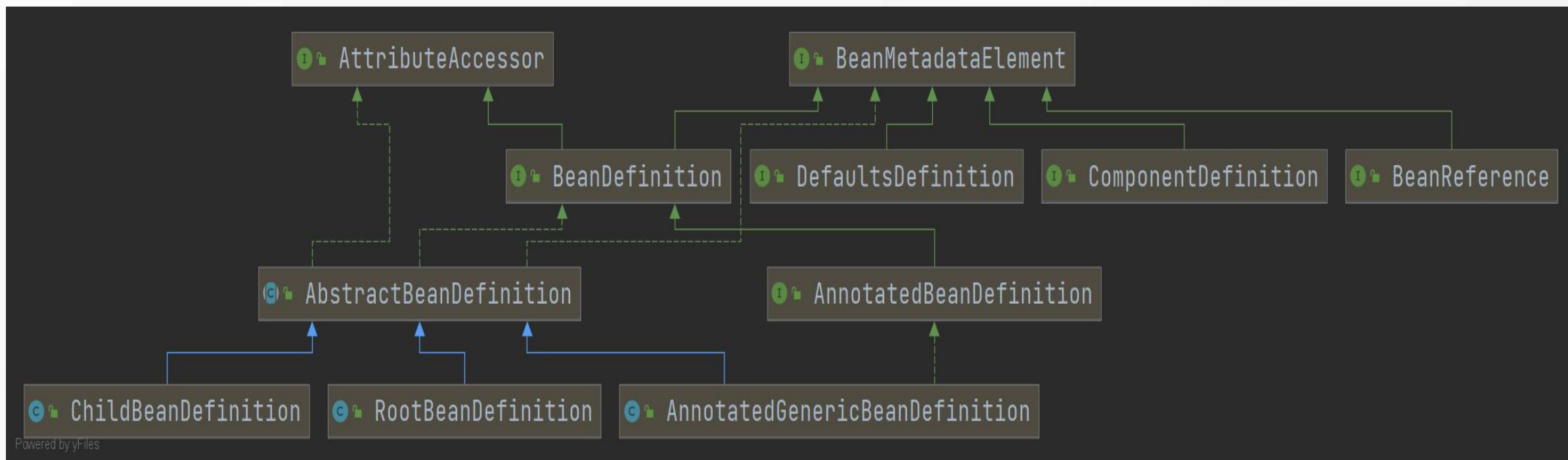
- ListableBeanFacotory: 的实现是DefaultListableBeanFactory, 保存了ioc容器中的核心信息

- AutowireCapableBeanFactory: 提供自动装配能力

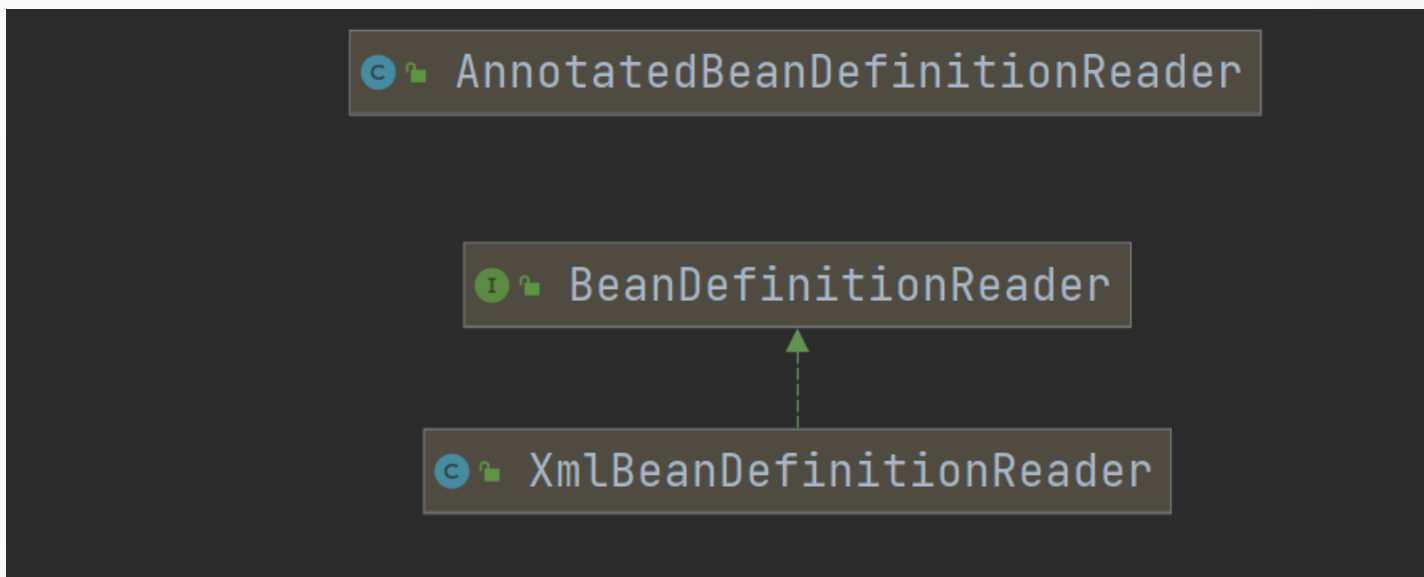
AnnotationApplicationContext组合了档案馆，他有自动装配能力。

- ApplicationContext和BeanFactory什么区别?

# 核心组件接口分析-BeanDefinition



# 核心组件接口分析-BeanDefinitionReader

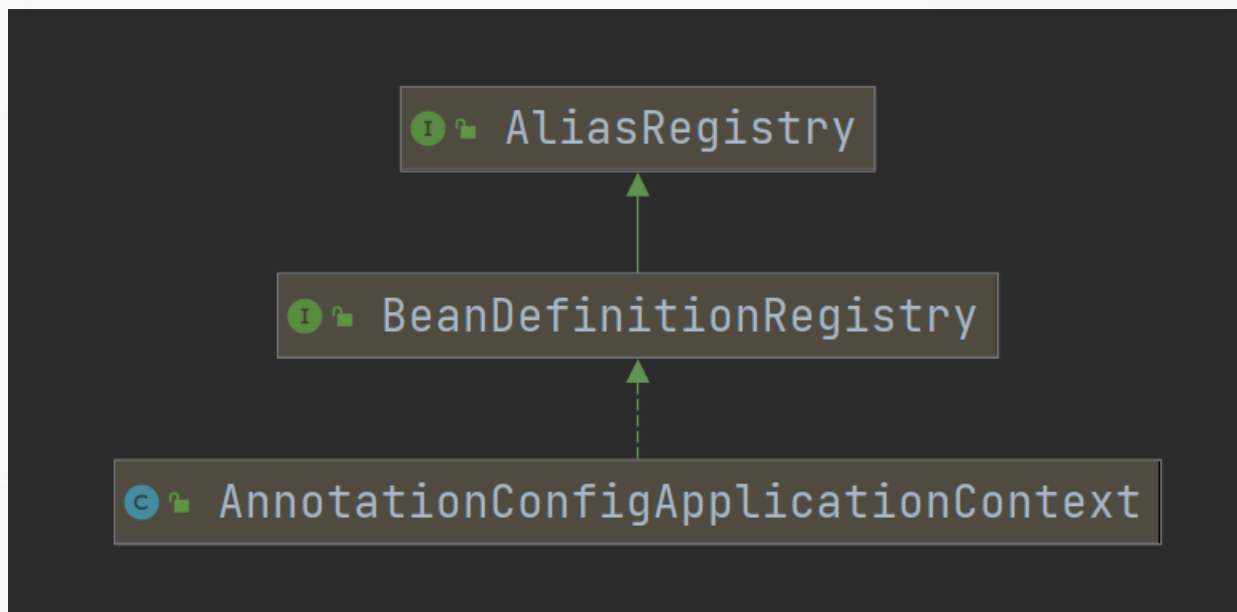


这是什么呢？

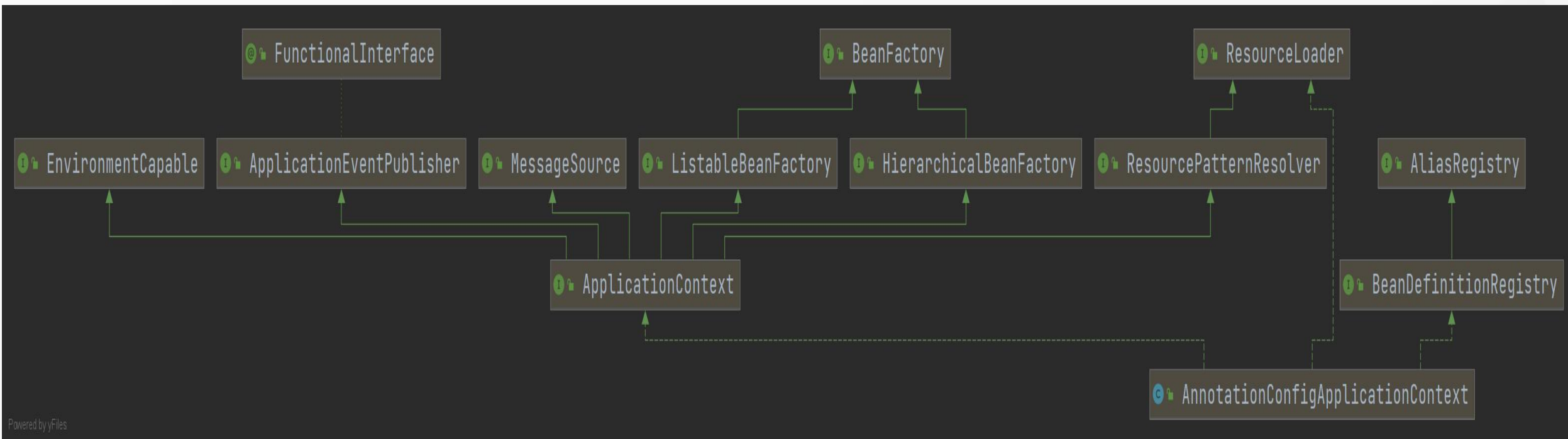
`BeanDefinitionParser`



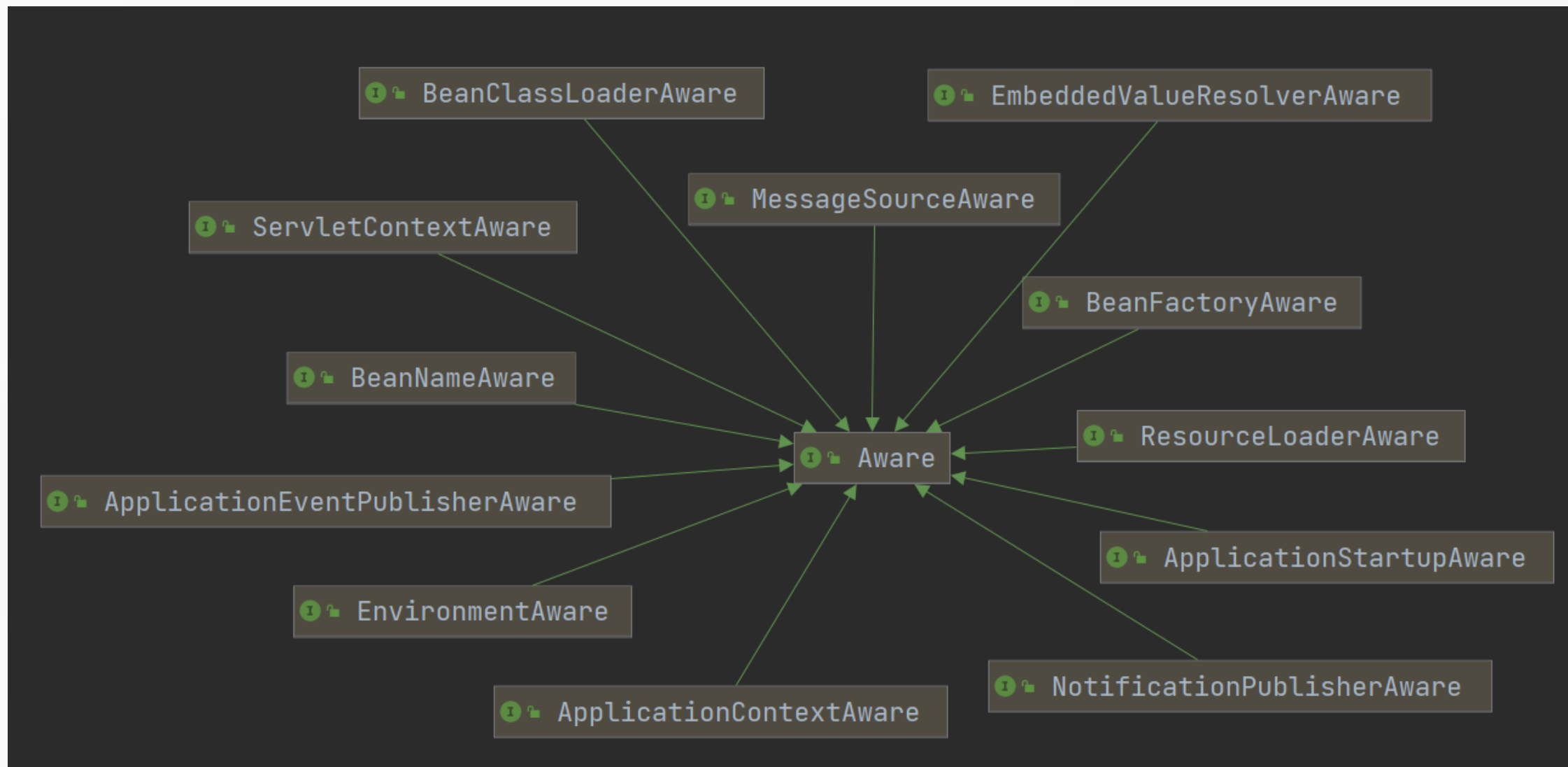
# 核心组件接口分析-BeanDefinitionRegistry



# 核心组件接口分析-ApplicationContext

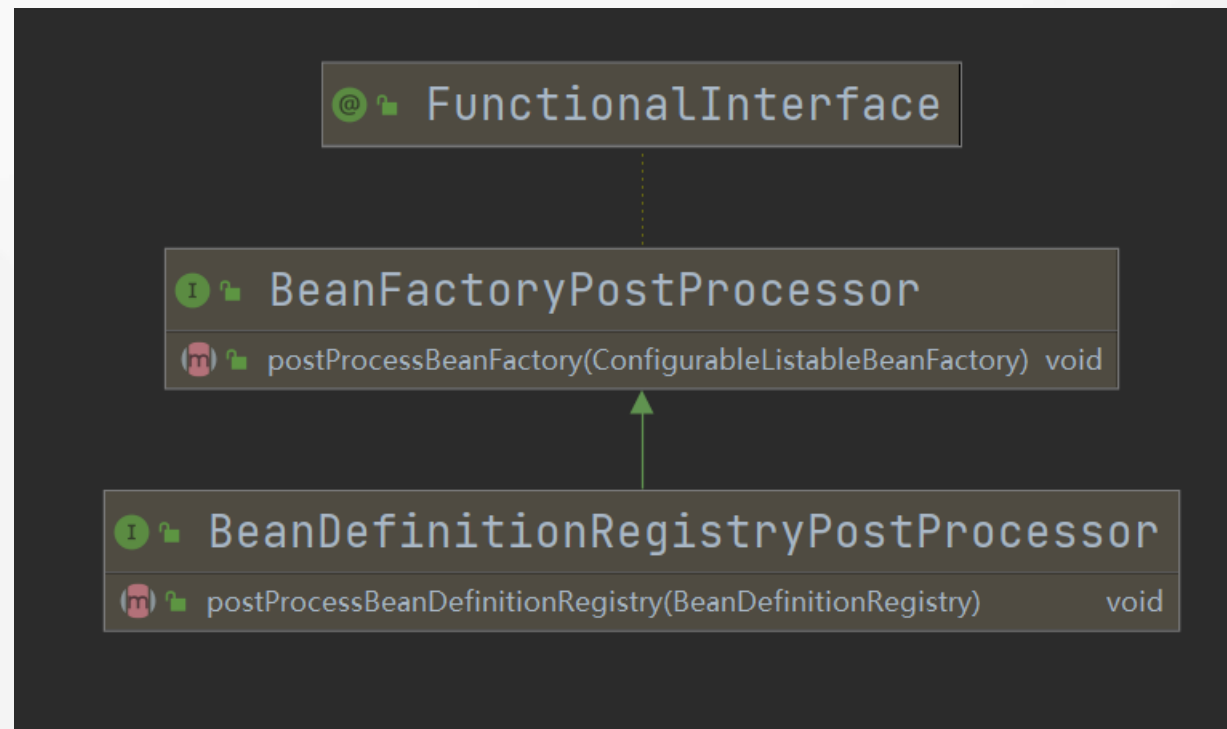


- 1、ioc事件派发器
- 2、国际化解析
- 3、**bean工厂**功能---自动装配被组合进来的
- 4、资源解析功能





# 生命周期后置处理-BeanFactoryPostProcessor


后置增强BeanFactory  
BeanFactoryPostProcessor



# 生命周期后置处理-InitializingBean、DisposableBean

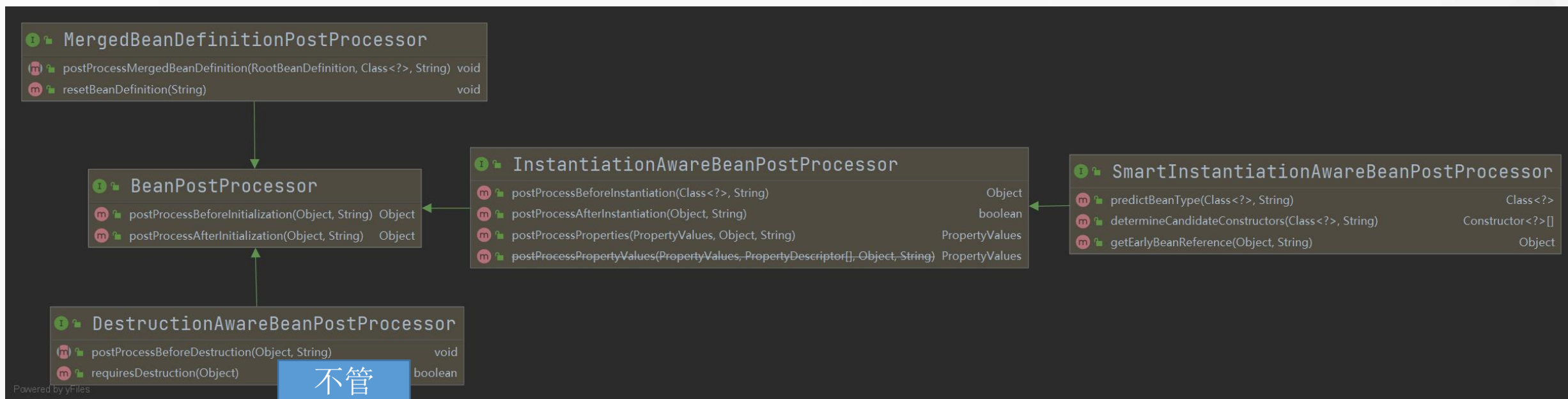
 InitializingBean

 afterPropertiesSet() void

 DisposableBean

 destroy() void

# 生命周期后置处理-BeanPostProcessor



在一个Bean初始化前后进行功能增强。

**BeanPostProcessor**: 后置增强普通的Bean组件

**BeanFactoryPostProcessor**: 后置增强BeanFactory















## SpringMVC 源码篇

✎ 请求流程

✎ 九大组件

✎ 关键流程

## SpringBoot 源码篇

✎ 配置原理

✎ 启动原理

✎ 整合原理

## 面试问题解答与案例分析篇

# 1、IOC容器是什么？DI是什么？

---



## 2、BeanFactory和ApplicationContext?

### 3、Spring怎么解决循环依赖问题？

---

## 4、Ribbon如何实现负载均衡？

---

5、Eureka工作原理，服务宕机后，什么时候剔除？  
影响其他客户端端Ribbon调用吗？Ribbon清单什么时候更新？

## 6、分布式Session一致解决，了解SpringSession吗？

谢谢观看