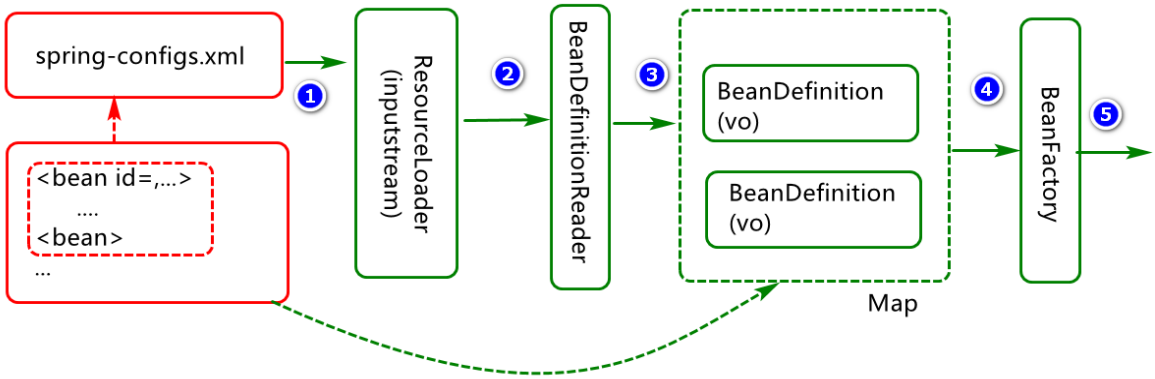
FRAMEWORK

第二章
SPRING 原理剖析与实践

1.Spring 核心模块设计及源码剖析

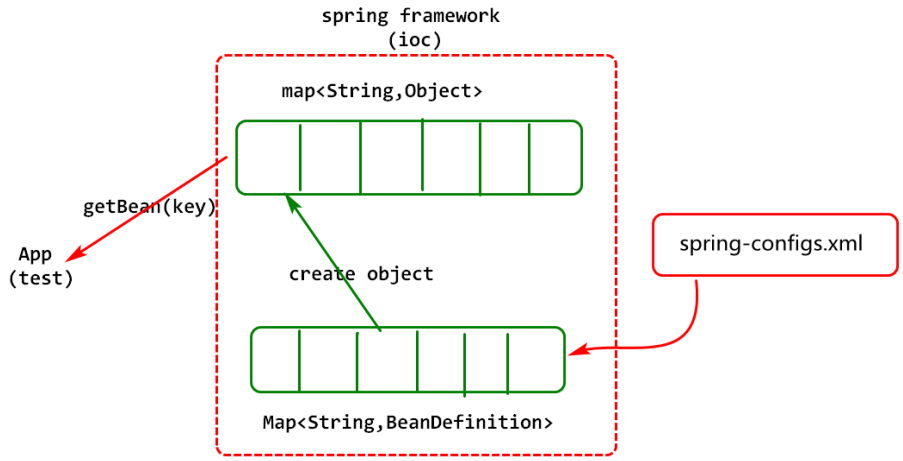
1.1. SPRING IOC 模块加强分析

1.1.1. 容器初始化

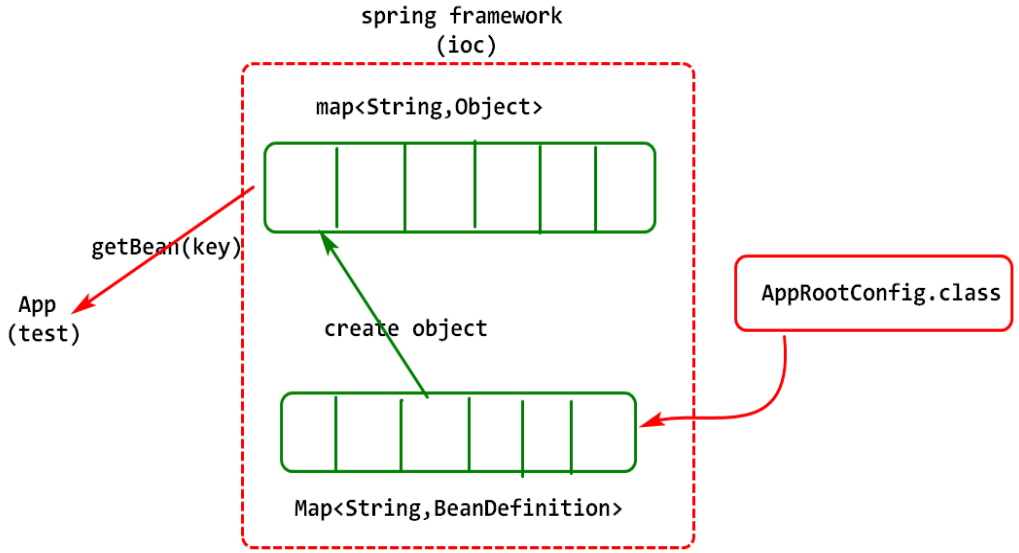


1.1.2. 两大容器对象

基于 xml 方式容器初始化

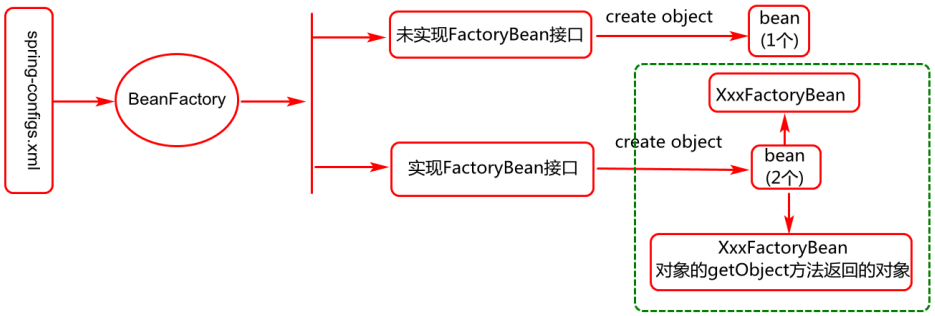


基于注解方式容器初始化：



1.1.3. 两大 Bean 对象

Spring 中两大 Bean 对象类型：



IOC Bean 容器（配置文件，工厂，容器，全局访问点）

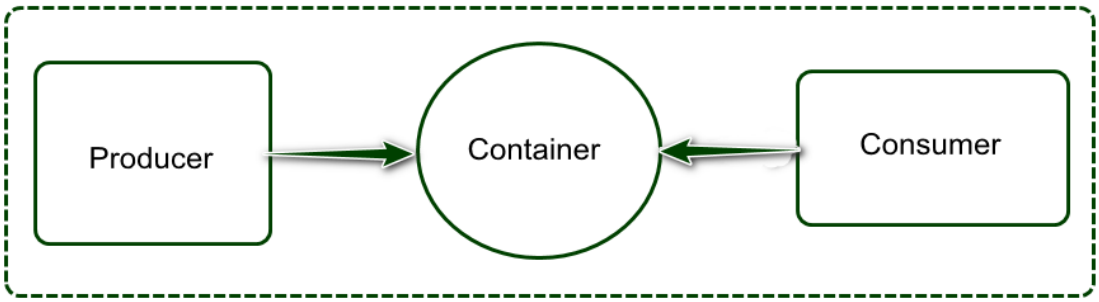
1.1.4. 两大 Bean 的配置方式

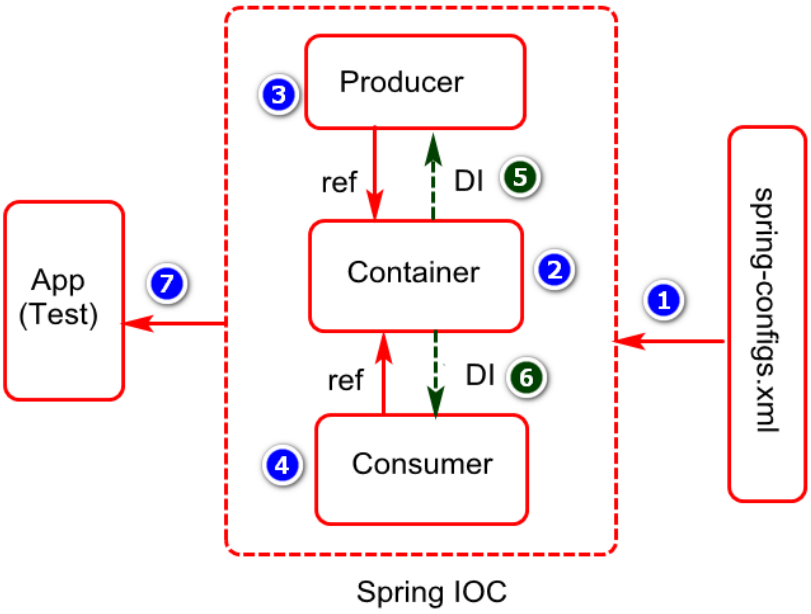
1．基于`

2．基于注解方式:@Controller,@Service,@Respository,@Component,
@Configuration,@Bean,.....

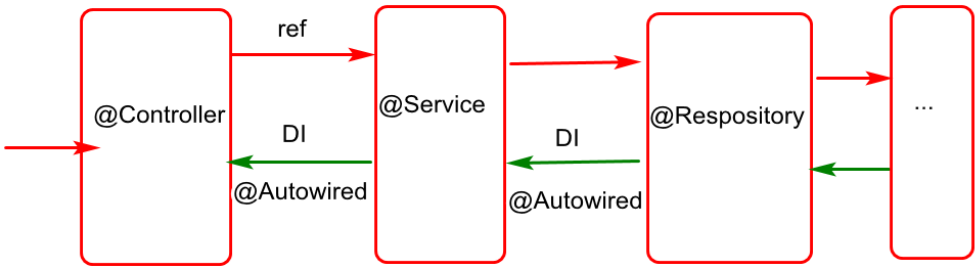
1.1.5. IOC 与 DI 应用分析

生产者消费者模式:





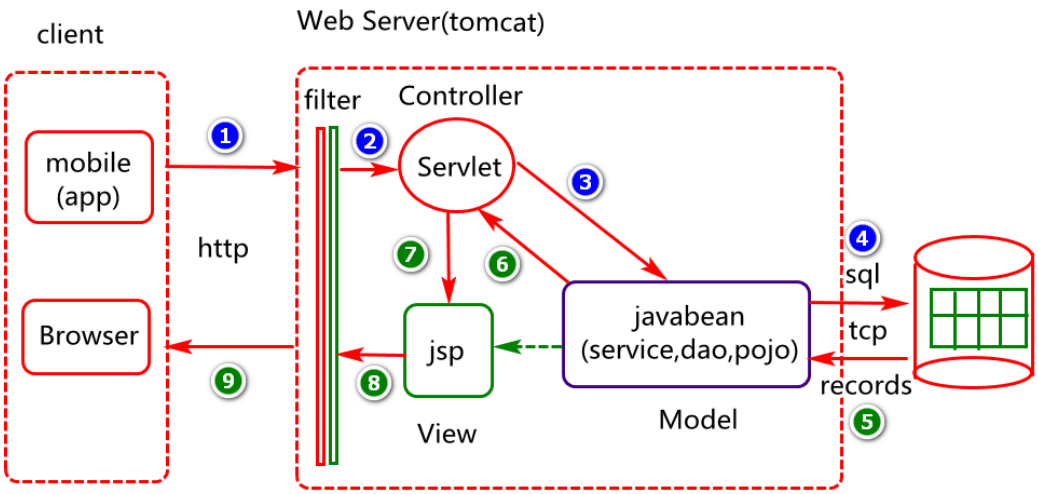
项目分层架构中的对象设计



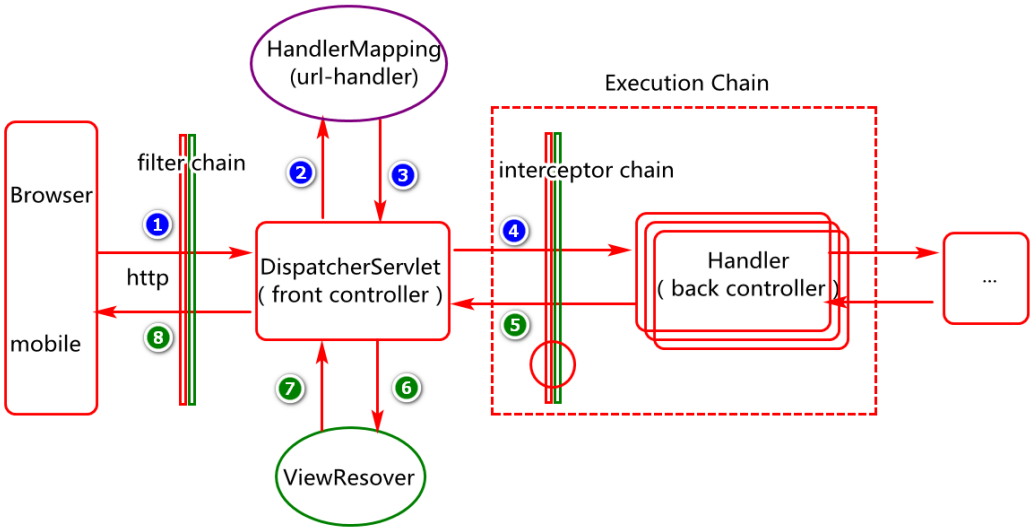
1.2. SPRING MVC 模块加强分析

1.2.1. MVC 核心组件及流程分析

MVC 设计思想基本实现：（jsp+servlet+javabean）

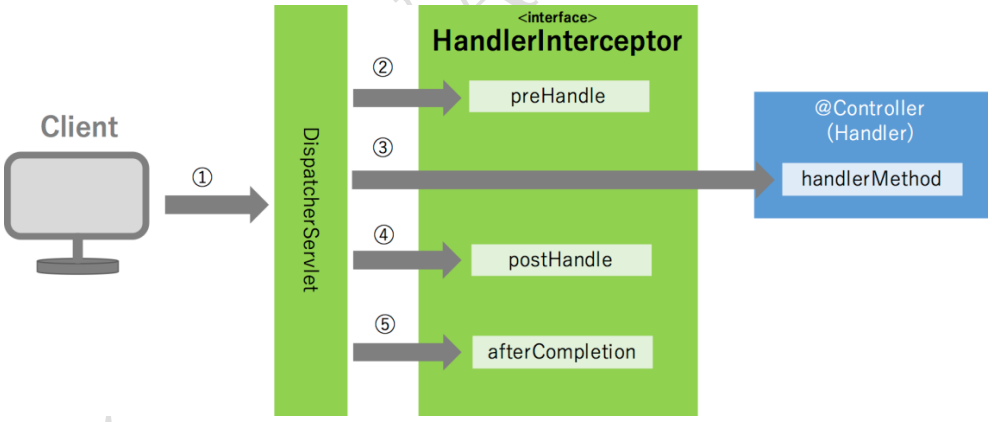


SPRING MVC 核心模块对象分析：（五大核心对象）

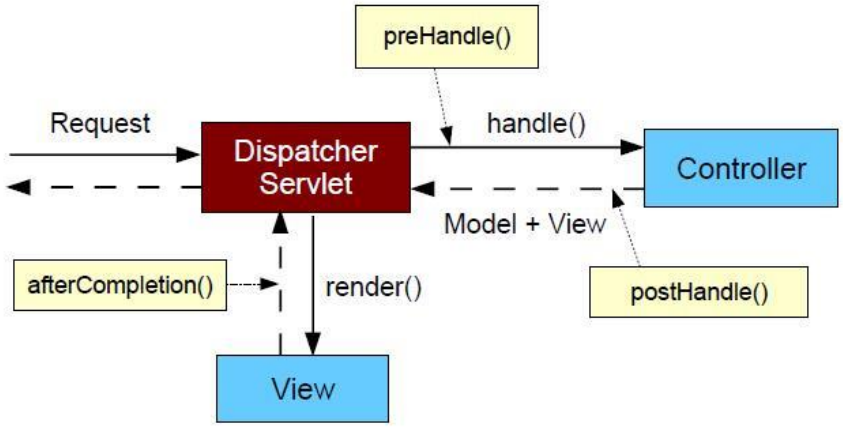


1.2.2. Spring MVC 中的拦截器

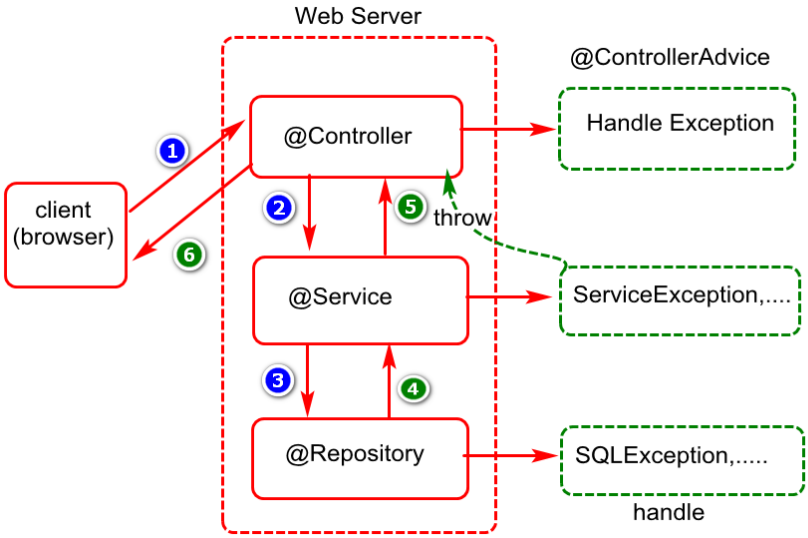
拦截器（Interceptor）应用原理及场景分析：



拦截器方法细节应用强化分析：



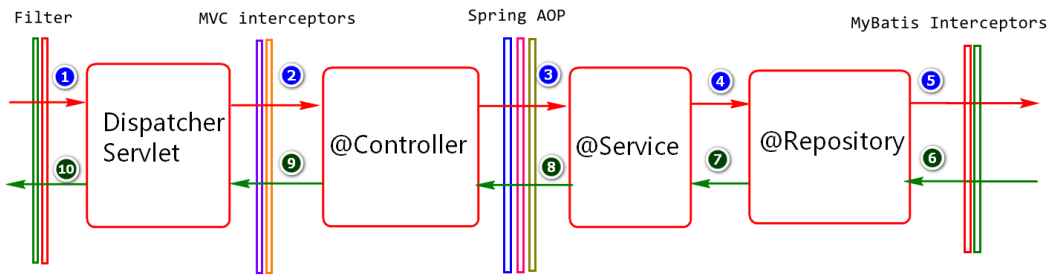
1.2.3. Spring MVC 异常处理增强分析。



1.3. SPRING AOP 模块加强分析

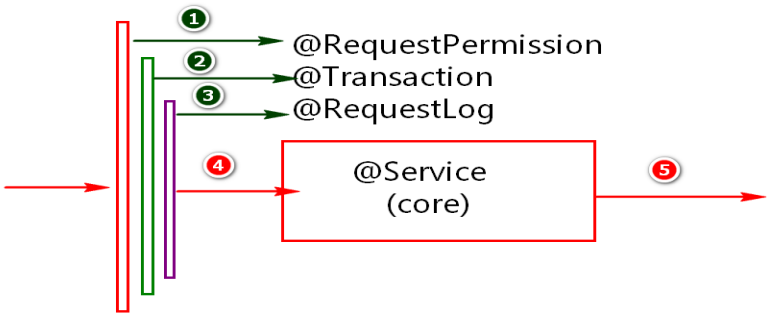
1.3.1. AOP 角色定位分析

Spring AOP 在我们项目中的定位:



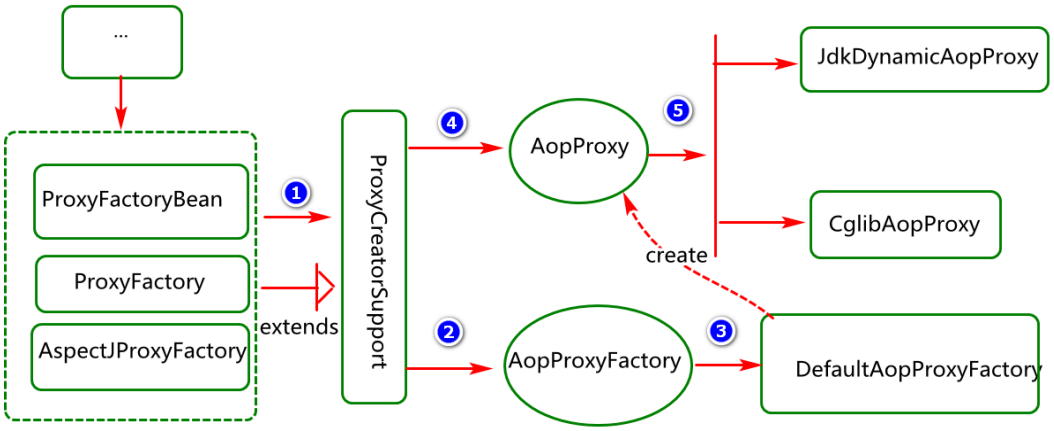
1.3.2. AOP 场景应用分析

Spring AOP 在我们项目中要实现的功能

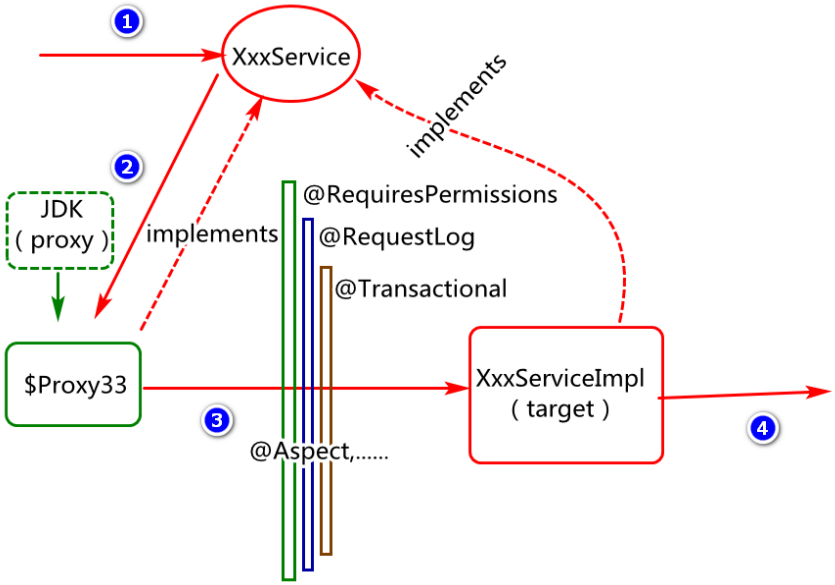


1.3.3. AOP 应用原理分析

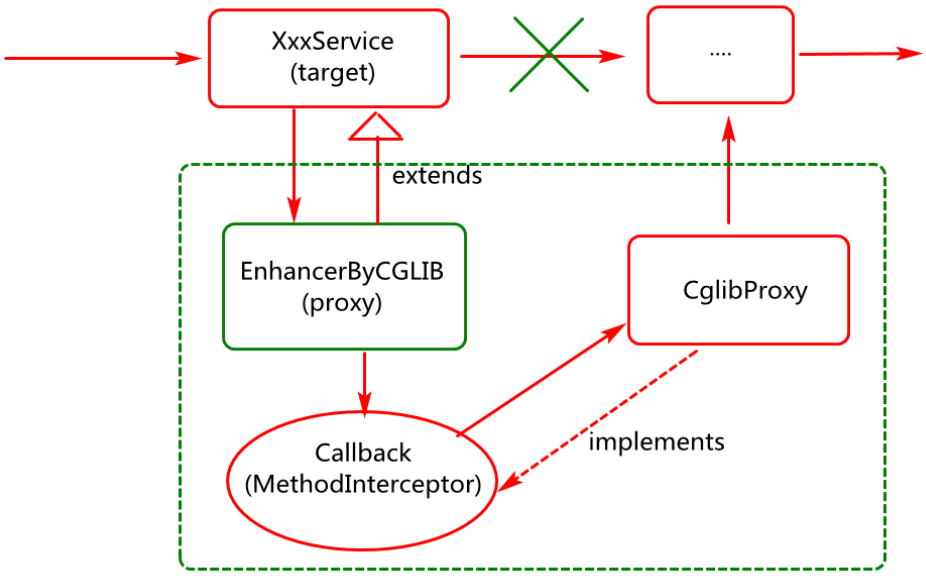
AOP 底层代理创建过程分析



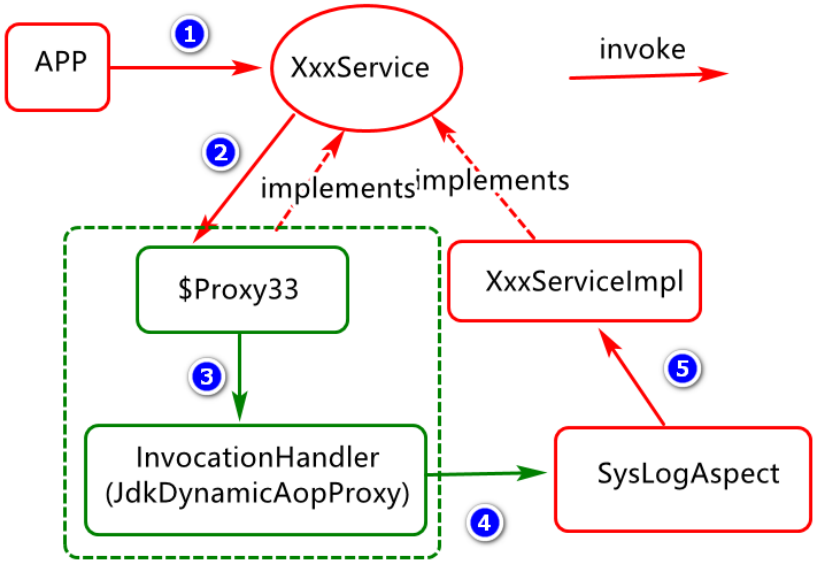
AOP 底层之动态代理应用实现分析与实现：



AOP 代理之 CGLIB 代理应用实现及分析：



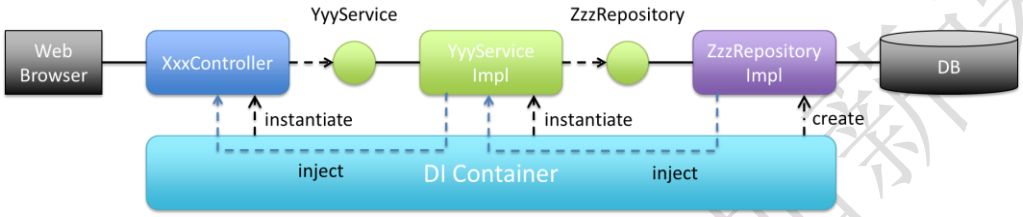
AOP 日志抓取功能实现及增强分析:



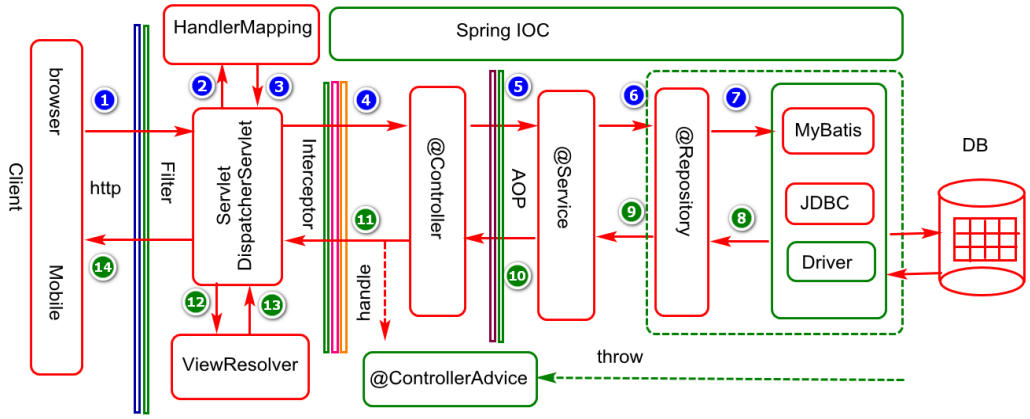
2.Spring 框架模块综合应用增强分析

2.1. Spring 核心模块综合定位分析

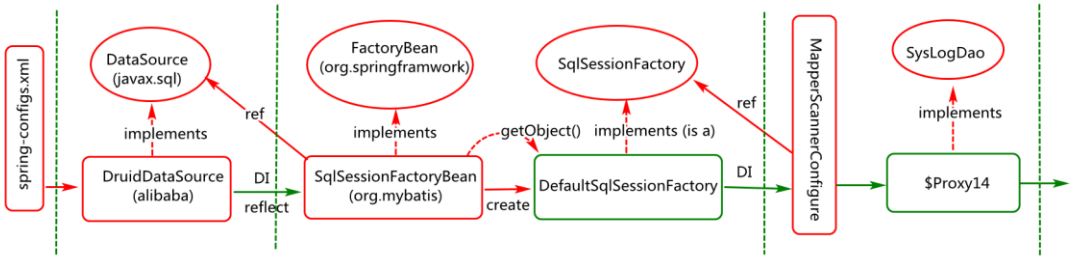
项目中对象角色定位及依赖分析：概要分析



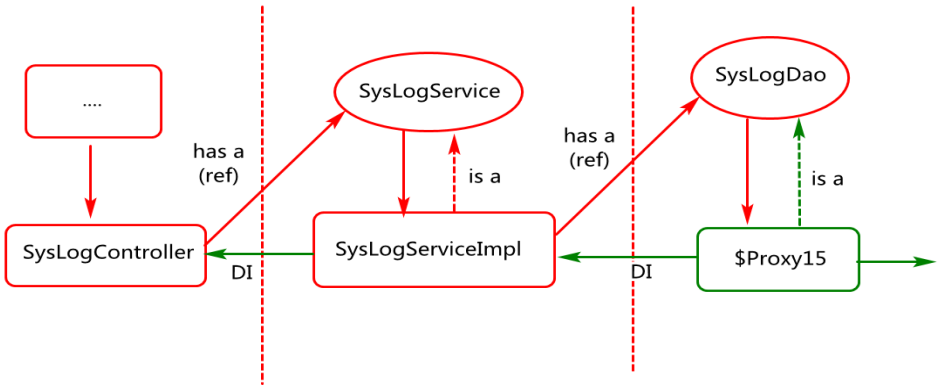
项目中对象角色定位及依赖分析：细节分析



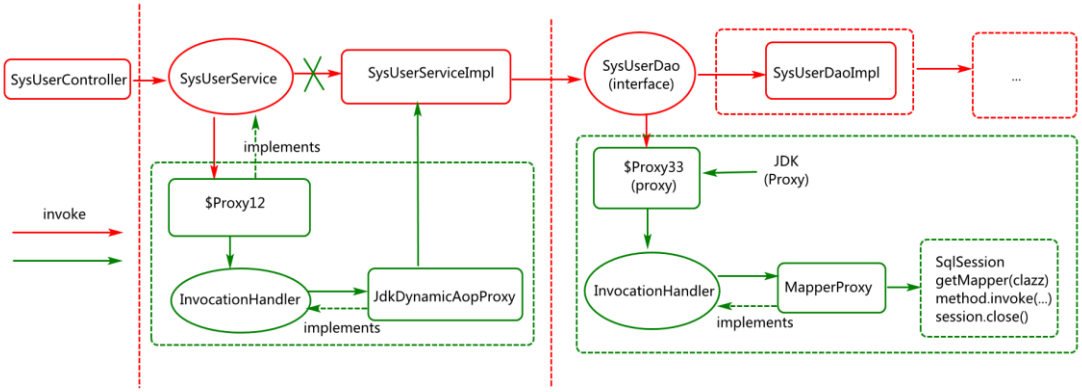
项目中资源整合分析：强化 IOC 设计



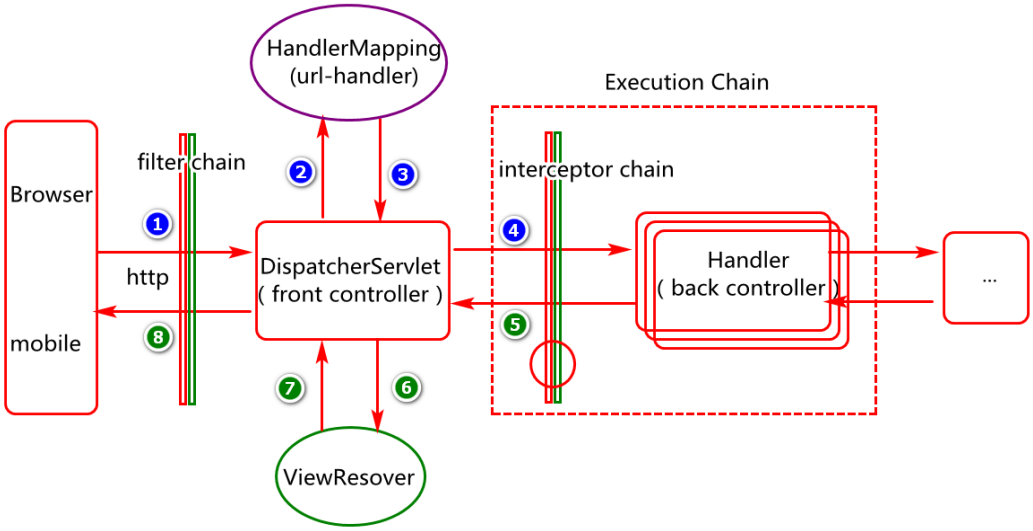
例如日志模块业务访问分析：



项目中资源整合分析：强化 AOP 设计

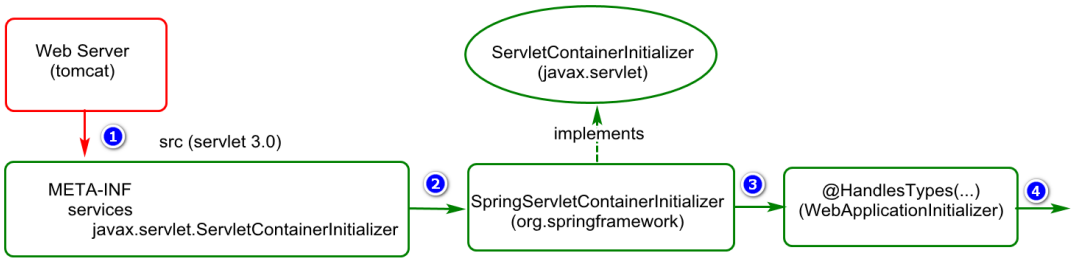


项目中资源整合分析：强化 MVC 设计



2.2. Spring 框架中 SPI 思想的应用实现

SPI（Service Provider Interface）:是一种以接口方式提供服务的标准。



3.Spring 面试问题分析及强化

3.1. 原理设计分析相关

1. 谈谈你对 Spring IOC 设计思想的理解?

   IOC（Inversion of Control）是一种控制反转思想，目标是科学管理对象资源，降低对象耦合，提高对象应用的灵活性。Spring 框架的核心就是 IOC 思想的实现，Spring 其它所有模块都基于 IOC，并且基于 IOC 实现资源整合，例如 mybatis，hibernate，shiro，中间件等。

2. 谈谈你对 Spring MVC 设计思想的理解?

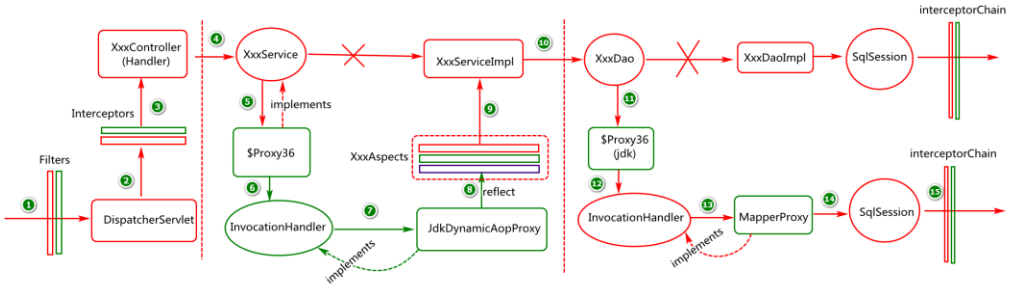
   MVC 是一种分层设计思想，目标是将复杂应用系统按照分层处理方式进行逐步设计和规划。并且通过这种设计实现系统业务的分而治之，从而降低程序的复杂度，提高程序的可维护性。

3. 谈谈你对 Spring AOP 设计思想的理解?

   AOP 是一种动态切入思想，目标是基于 OCP 原则，采用代理方式控制对象并

提供其功能扩展，例如项目中的事务控制，权限控制，缓存处理，日志处理等。

4．如何理解项目中的过滤器，拦截器以及 AOP 的定位?



3.2. 设计模式实现相关

1．简单工厂模式:BeanFactory
2．工厂方法模式:ProxyFactoryBean
3．抽象工厂模式:ClientHttpRequest
4．单例模式:Singleton
5．代理模式：Aop Proxy
6．策略模式：AOP 代理策略，SimpleInstantiationStrategy
7．适配器模式:AdvisorAdapter
8．模版方法模式:JdbcTemplate
9．命令模式：DispatcherServlet
10...