spring 轻量化、扩展性强、易用、易维护、侵入性较弱、模块化的代码框架。

主要是解决管理对象生命周期的问题，以及这些对象之间的关系管理。

spring是一个总的项目，它包含很多模块。我们熟知的依赖倒置、bean、aop、orm、jdbc、web等核心特性属于spring framework模块，一个模块包含多个jar。对于spring-web,如果不需要web功能，就不要加入依赖，只加入核心的spring-context/core/beans就行。

跟安全有关的属于security模块，其依赖的组为org.springframework. security,熟知的jar有spring-security-web/config/taglibs等。

跟数据库有关的属于data模块，其依赖的组为org.springframework. data,熟知的jar有spring-data-commons/jpa/mongodb等。

以spring的主要目标来看，它解决了目标的创建问题，也就是设计模式的创建型模式全部都得以实现——工厂方法、抽象工厂、建造者、原型、单例。

spring-aop实现了设计模式的装饰者、代理模式。它的代码实现了更多的模式，比如核心的AbstractApplicationContext类，主方法refresh是一个算法，它依次调用了不同的模板方法，这些方法在子类中得以实现/重写，因此属于模板方法模式。这个类内部又包含了一系列的接口，如Environment、MessageSource，然后自身暴露了这些接口的方法，默认实现是直接调用自身对应的对象的方法，因此属于代理模式。它的内部还维护了一个ApplicationListener的Set集合，用于事件的发布，监听者会感知到事件的发生，感知到的事件被事件对象封装，这里同时用了观察者模式和命令模式。Spring代码内还大量地使用了策略模式，在web的DispatherServlet中有个Properties的defaultStrategies属性，读取类路径的某个properties文件，里面包含了策略接口：默认策略实现类的对应关系。比如里面包含了HandlerMapping的默认实现类BeanNameUrlHandlerMapping（将请求的url映射到/开始的名字的bean，也支持\*匹配路径，）,DefaultAnnotationHanlderMapping（用来将url映射到被Controller或者RequestMapping等标注的类或方法上）。ViewResolver的默认实现类InternalResourceViewResolver。

1、@Autowired与@Resource都可以用来装配bean. 都可以写在字段上,或写在setter方法上。   
2、@Autowired默认按类型装配（这个注解是属业spring的），默认情况下必须要求依赖对象必须存在，如果要允许null 值，可以设置它的required属性为false，如：@Autowired(required=false) ，如果我们想使用名称装配可以结合@Qualifier注解进行使用，如下：

**Java代码  [收藏代码](javascript:void())**

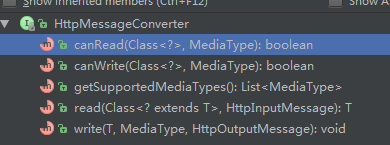
1. @Autowired() @Qualifier("baseDao")
2. **private** BaseDao baseDao;

 3、@Resource（这个注解属于J2EE的），默认安照名称进行装配，名称可以通过name属性进行指定，   
如果没有指定name属性，当注解写在字段上时，默认取字段名进行按照名称查找，如果注解写在setter方法上默认取属性名进行装配。 当找不到与名称匹配的bean时才按照类型进行装配。但是需要注意的是，如果name属性一旦指定，就只会按照名称进行装配。

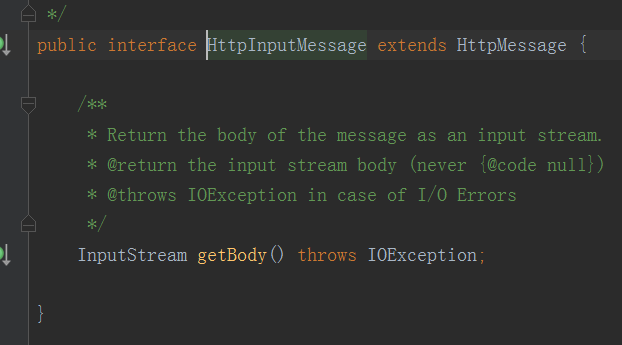
**Java代码  [收藏代码](javascript:void())**

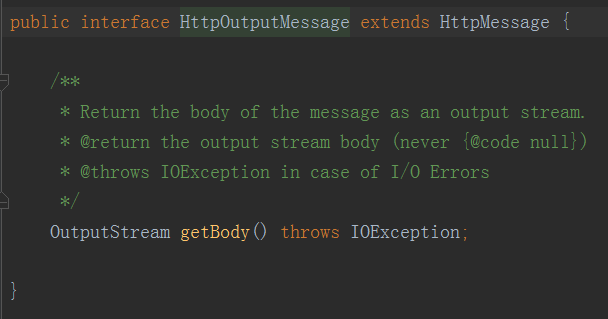
1. @Resource(name="baseDao")
2. **private** BaseDao baseDao;

我喜欢用 @Resource注解在字段上，且这个注解是属于J2EE的，减少了与spring的耦合。最重要的这样代码看起就比较优雅。









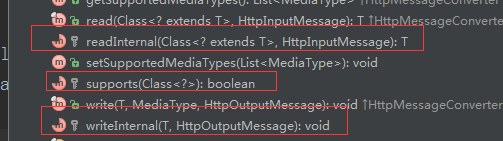
HttpMessageConverter的

基本抽象实现类

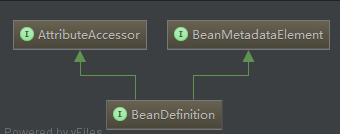
AbstractHttpMessageConverter

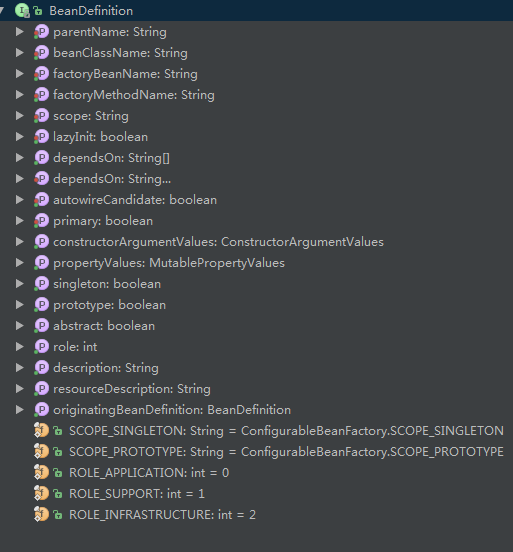


其内部有一个list存放支持的MIME

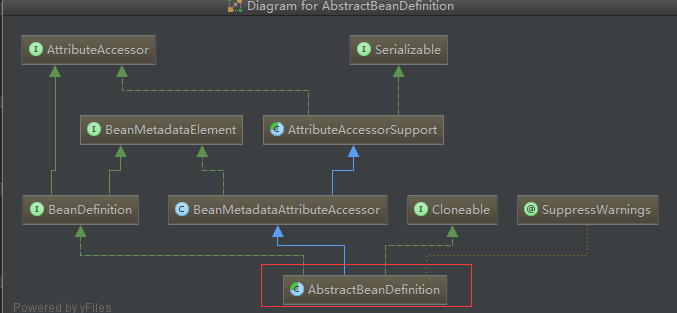
子类只需要实现这3个方法

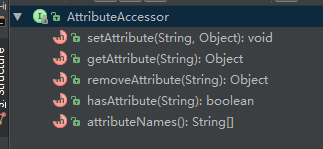
Spring中的BeanDefinition

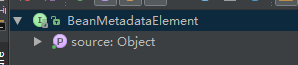




它的抽象实现类

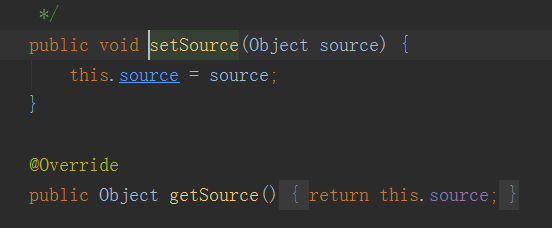




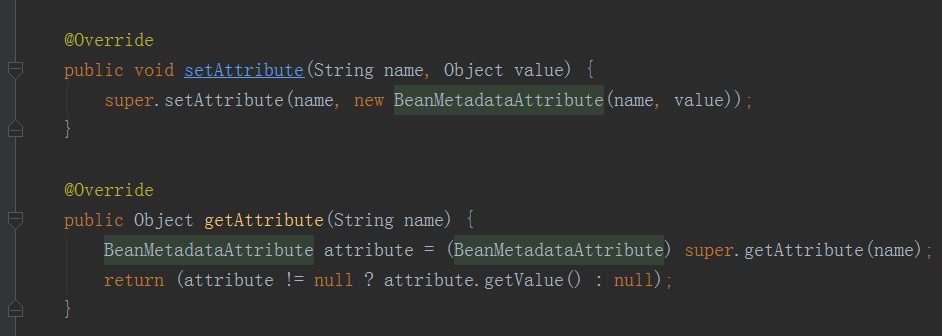


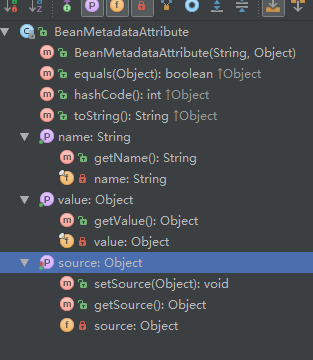
其中不过是一个含有LinkedHashMap<String,Object>的属性提取/存储器

在上面class之上多了实现



以及重写了属性设置/提取方法：





然后就是抽象类，父类，



它是GenericBeanDefinition，RootBeanDefinition，ChildBeanDefinition的公共父类

实现了大部分方法，拥有大部分private property

没有实现set/getParentName

GenericBeanDefinition，主要实现了set/getParentName，主要用来处理普通用户可见的bean定义