## Spring讲解

阅读开发者文档

三种方式

方法一Xml配置

Bean标签



构造方法注入



属性注入

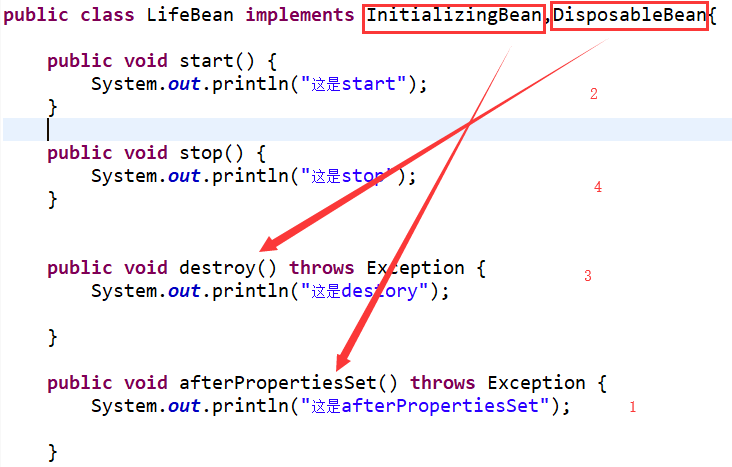


两种方式的依赖注入，构造器是先执行的后执行的是set，尽量少用构造器，在初始化的时候就会启动构造器，避免在初始化的时候进行依赖注入。Set方式在容器一开始的时候不会实例化，

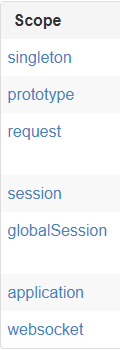
生命周期



这种写法看起来也许很吊，但是会造成耦合，并不推荐



Bean的作用范围



ApplicationContextAware

bean可以通过ApplicationContextAware以编程方式操作创建它们的方法，或者通过将引用转换为此接口的已知子类（例如ConfigurableApplicationContext，公开其他功能）来操作。有时这种能力很有用; 但是，通常你应该避免它，因为它将代码耦合到Spring并且不遵循Inversion of Control样式，[1见开发者文档]

@Required 注释适用于bean属性setter方法但必须填充，否则会报错，并且@option注解可以不填充。Aware 接口是容器提供访问资源的渠道Bean 父类容器的扩展

方法二注解

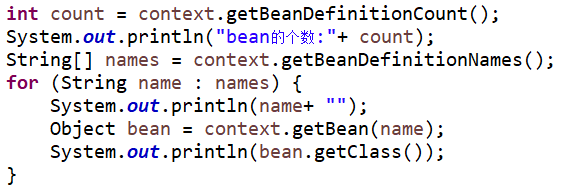
相当于在spring容器中写入一个bean标签

自动装配

按文件目录获取配置文件



打印容器中bean的个数



@Autowired注解可以应用于构造器，set方法，私有属性。

@Primary注解，Autowired按类型加载时如果有多个相同类型，会优先primary

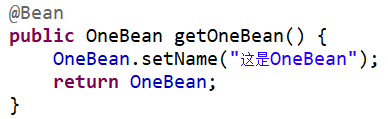
@Resource=@Autowired+@Qualifer

@postConstruct初始化时预先填充

@PreDestory销毁时行动

方法三配置类



设置bean标签

容器一

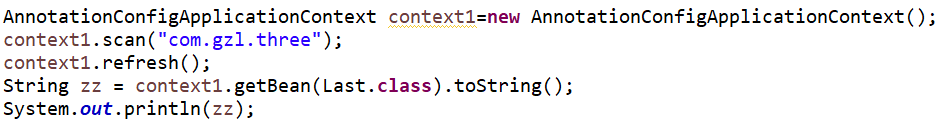


容器二



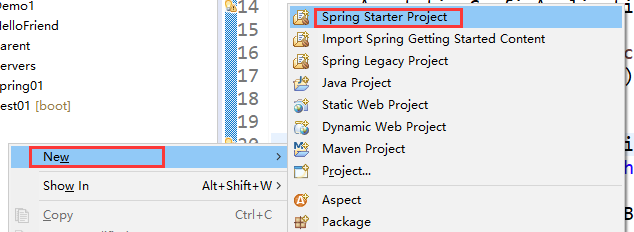
输出：

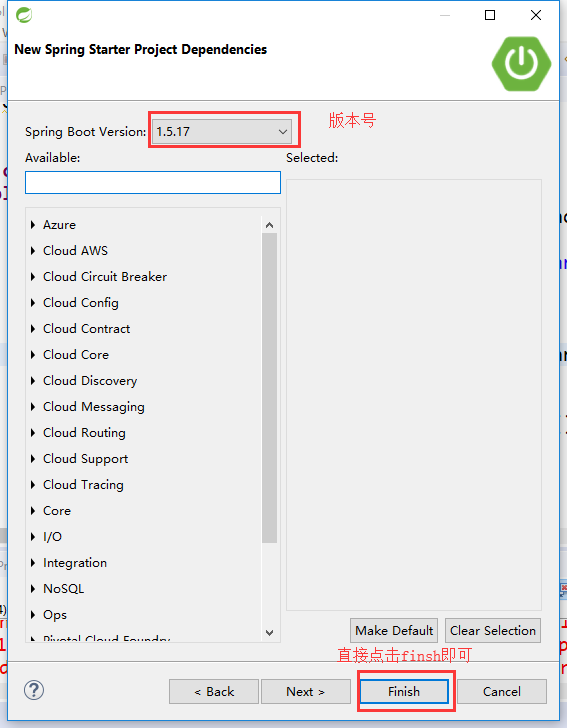
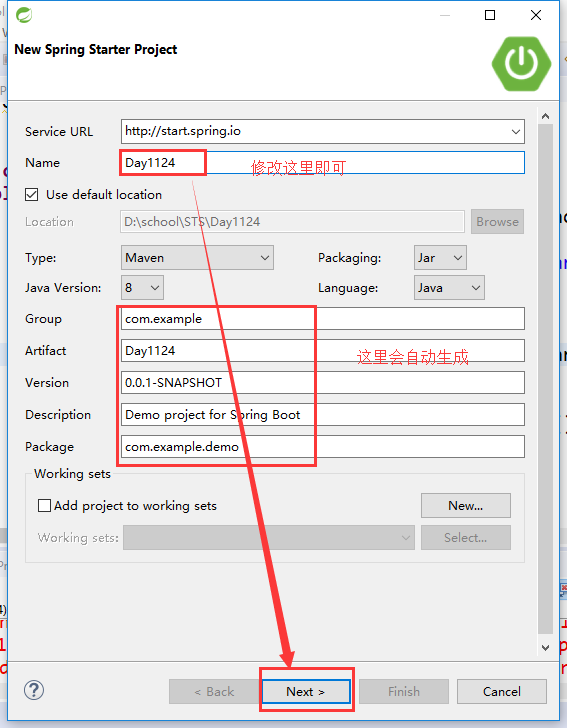




## SpringBoot初识

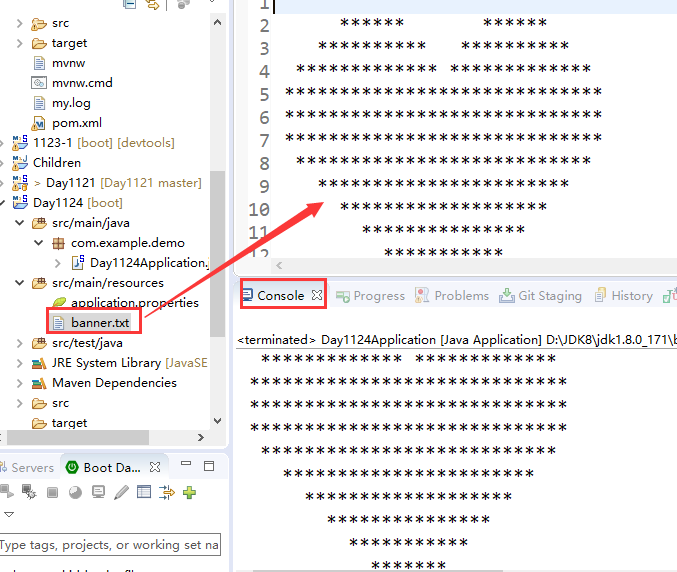
1. 建立项目





注意：application.java扫描本目录及其子目录下的其他文件所有其他文件应在其包下建立。

欢迎横幅



关于@SpringBootApplication

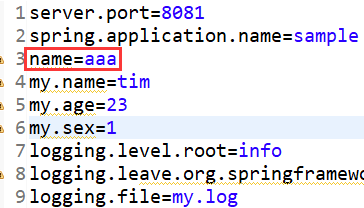


使用springb-boot-devtools可以实现热部署

使用ApplicationRunner或CommandLineRunner



通过配置文件为属性赋值



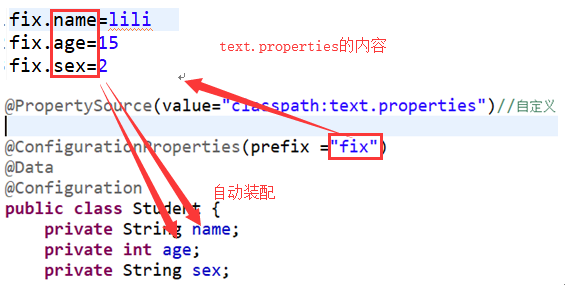


.yml配置高于.propertise

关于占位符的更多扩展：

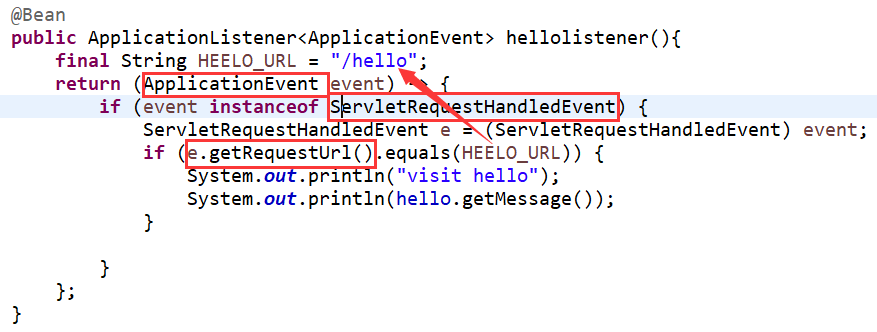


@ConfigurationPropertie



逻辑建立地址

监听器demo，感觉非常有用



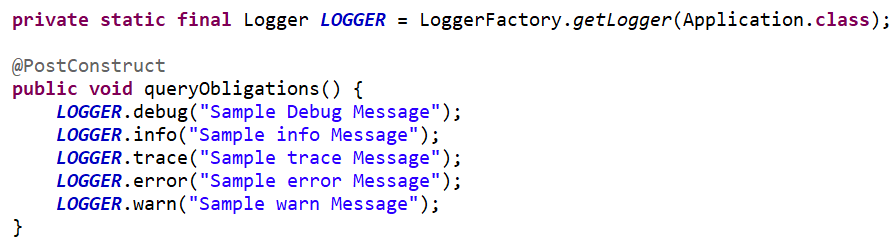
## springboot深入





如果没有active，会执行默认配置文件，但现在执行的是hello配置文件

配置日志



在配置文件中：

