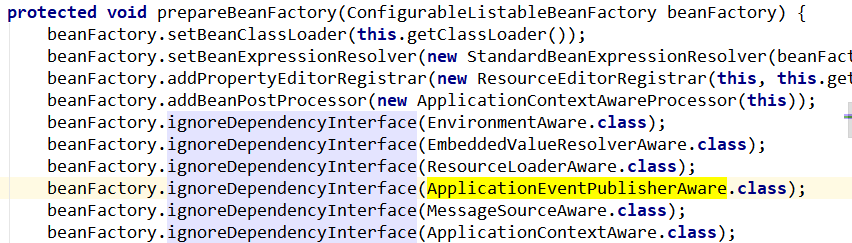
### Spring注册监听事件：

第一步创建event事件并继承ApplicationEvent类。

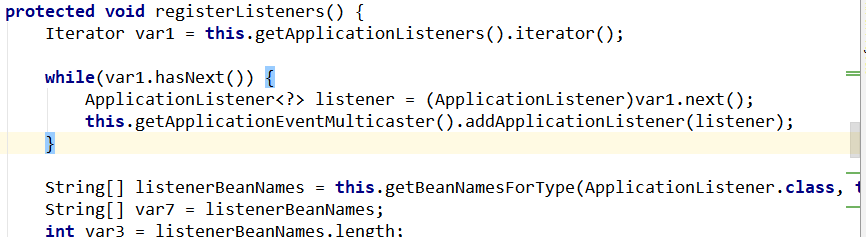
第二步创建listener监听类并继承ApplicationListener类，注意这里的ApplicationListener类的泛型应该是ApplicationEvent类。

第三步创建aware类并继承ApplicationEventPublisherAware类,这个类的setApplicationEventPublisher(ApplicationEventPublisher applicationEventPublisher)会将ApplicationContext设置到这个类的变量中。以便后面发布事件。



上图中的ignoreDependencyInterface()将接口的Class对象放入set中，以方便后续操作。猜测，在某个地方将执行实现该接口的类的set方法。

注意：ApplicationContext继承了ApplicationEventPublisher，所以直接设置的ApplicationContext。在refresh()方法中的registerListener()方法中会注册所有的监听器。



ApplicationEventPublisher的publish方法会根据事件类型取出对应的监听器并执行onApplicationEvent()方法。

参考网站：<https://www.cnblogs.com/liuruowang/p/4542739.html>

### Spring的依赖注入：

@Autowired 注解可以对成员变量、构造方法、方法进行标注，来完成自动装配工作；放在成员变量上，也可以放在成员变量的set()方法上，**也可以放在任意方法上表示自动执行当前方法，如果方法有参数，会在IOC容器中自动寻找同类型参数为其传值。**

**参考网站：<https://blog.csdn.net/GavinLi2588/article/details/103483634>**

### Java配置和注解配置：

1）涉及到全局配置的，例如数据库相关配置、MVC相关配置等，就用JAVA配置的方式； //@Confirguration 相当于spring的配置文件XML @Bean 用到方法上，表示当前方法的返回值是一个bean

2）涉及到业务配置的，就使用注解方式； //注解配置： @Service @Component @Repository @Controller

### Spring中的jdbcTemplate的使用：

参考网站：<https://www.iteye.com/blog/alex4java-2294833>

1. 用户发送请求至前端控制器DispatcherServlet；  
   （2） DispatcherServlet收到请求后，调用HandlerMapping处理器映射器，请求获取Handle；  
   （3）处理器映射器根据请求url找到具体的处理器，生成处理器对象及处理器拦截器(如果有则生成)一并返回给DispatcherServlet；  
   （4）DispatcherServlet 调用 HandlerAdapter处理器适配器；  
   （5）HandlerAdapter 经过适配调用 具体处理器(Handler，也叫后端控制器)；  
   （6）Handler执行完成返回ModelAndView；
2. 在执行之前还会执行一些拦截方法，自定义拦截器需要实现HandlerInterceptor接https://blog.csdn.net/qq\_40323256/article/details/89875160  
   （7）HandlerAdapter将Handler执行结果ModelAndView返回给DispatcherServlet；
3. 前面部分代码会放到try代码块中，出现异常会调用异常处理器。https://blog.csdn.net/qq\_29914837/article/details/82697089

https://blog.csdn.net/yb223731/article/details/104977453/  
（8）DispatcherServlet将ModelAndView传给ViewResolver视图解析器进行解析；  
（9）ViewResolver解析后返回具体View；  
（10）DispatcherServlet对View进行渲染视图（即将模型数据填充至视图中）  
（11）DispatcherServlet响应用户