* SpringBoot整个启动流程分为两个步骤：初始化一个SpringApplication对象、执行该对象的run方法。

### SpringApplication初始化

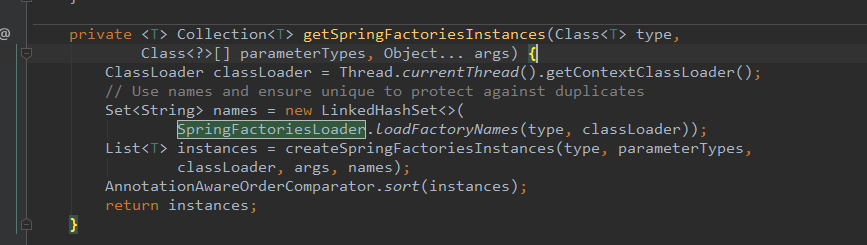
spring.factories定义了springboot应用所需的listener/processor/initializer/error reporters等

### 

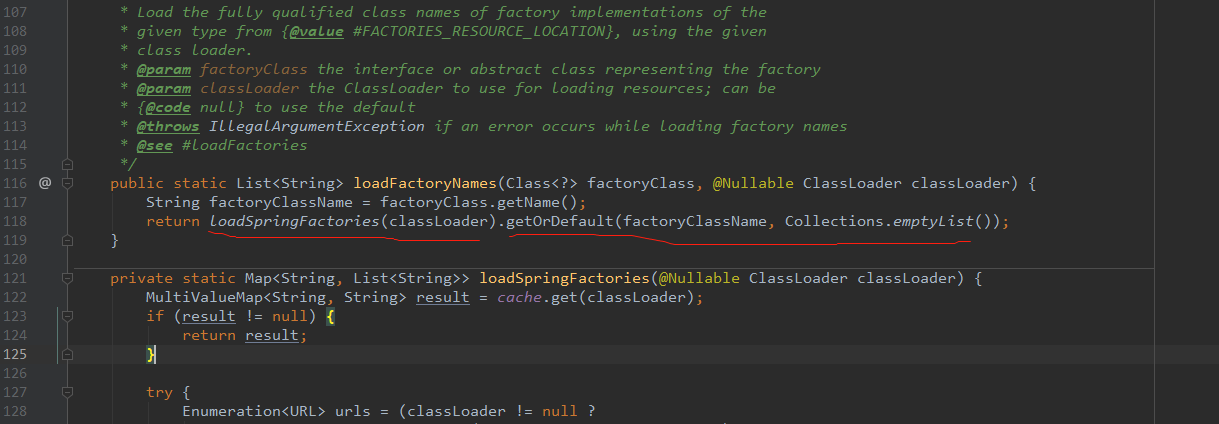
### new SpringApplication() 实例化SpringApplication时会进行系统初始化

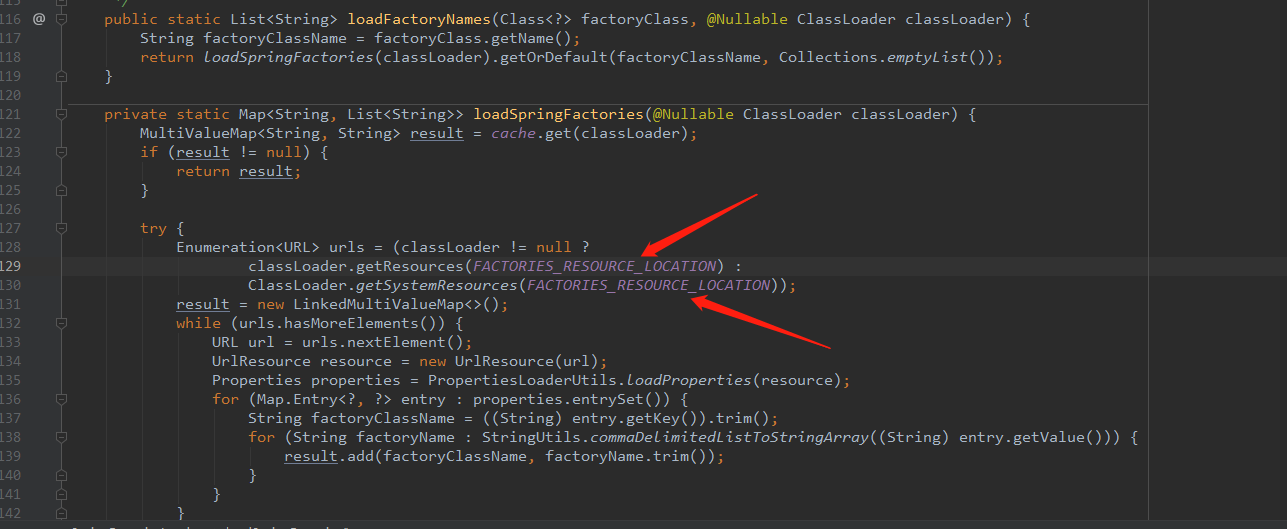
### 

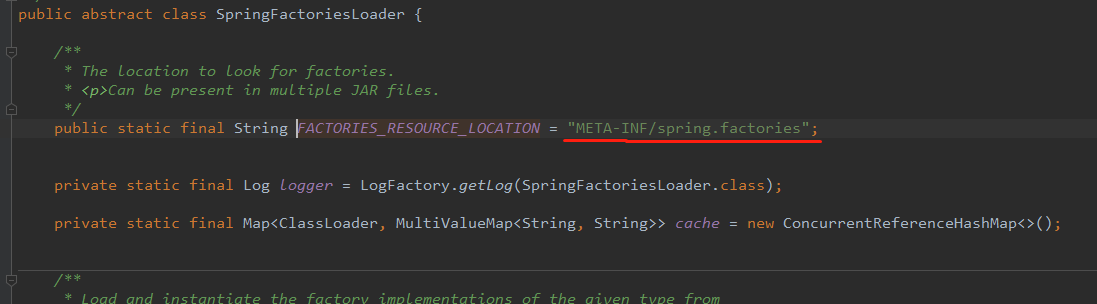
过程a是判断是否是web应用  
过程b就是获取spring.factories文件中ApplicationListener和ApplicationContextInitializer的相应实现类，具体是通过SpringFactoriesLoader类来实现的。



具体的SpringFactoriesLoader.loadFactoryNames

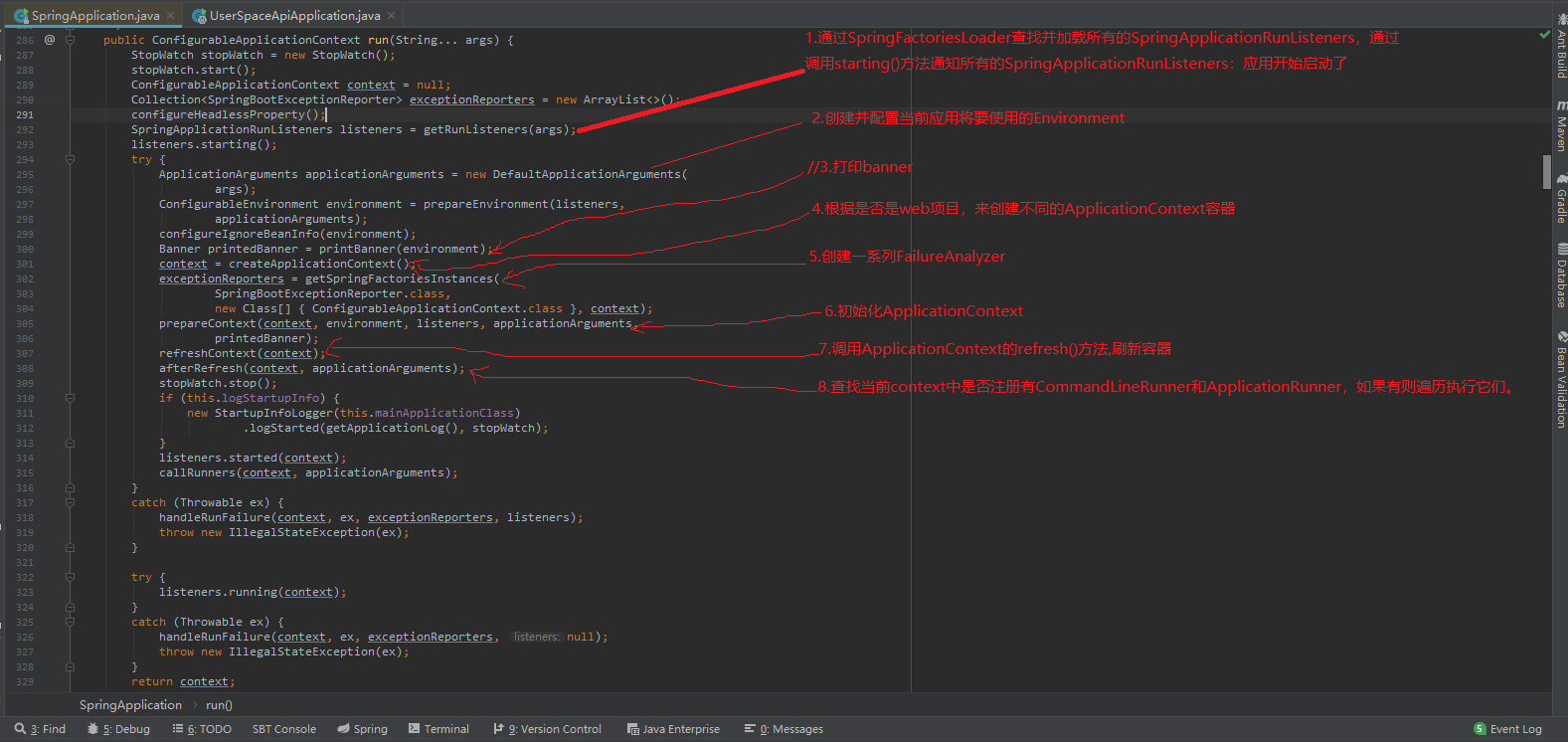






### Spring Boot启动流程

整个启动流程都封装在SpringApplication.run方法中，本质上其实就是在spring的基础之上做了封装，做了大量的扩张。

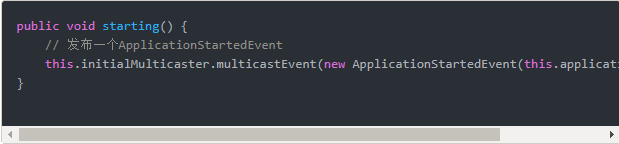


1.通过SpringFactoriesLoader查找并加载所有SpringApplicationRunListeners，通过调用starting()方法通知所有的SpringApplicationRunListeners：应用开始启动了。（SpringApplicationRunListeners其本质上就是一个事件发布者，它在SpringBoot应用启动的不同时间点发布不同应用事件类型(ApplicationEvent)，如果有哪些事件监听者(ApplicationListener)对这些事件感兴趣，则可以接收并且处理）

看下SpringApplicationRunListeners源码：



SpringApplicationRunListener只有一个实现类：EventPublishingRunListener。①处的代码只会获取到一个EventPublishingRunListener的实例，我们来看看starting()方法的内容：



2.创建并配置当前应用将要使用的Environment，Environment用于描述应用程序当前的运行环境，其抽象了两个方面的内容：配置文件(profile)和属性(properties)，不同的环境(eg：生产环境、预发布环境)可以使用不同的配置文件，而属性则可以从配置文件、环境变量、命令行参数等来源获取。因此，当Environment准备好后，在整个应用的任何时候，都可以从Environment中获取资源。

(1)判断Environment是否存在，不存在就创建（如果是web项目就创建StandardServletEnvironment，否则创建StandardEnvironment）

(2)配置Environment：配置profile以及properties

(3)调用SpringApplicationRunListener的environmentPrepared()方法，通知事件监听者：应用的Environment已经准备好

3.打印banner（可以自定义）

4.根据是否是web项目，来创建不同的ApplicationContext容器

5.创建一系列FailureAnalyzer，创建流程依然是通SpringFactoriesLoader获取到所有实现FailureAnalyzer接口的class，然后在创建对应的实例。FailureAnalyzer用于分析故障并提供相关诊断信息。

6.初始化ApplicationContex

 将准备好的Environment设置给ApplicationContext

 遍历调用所有的ApplicationContextInitializer的initialize()方法来对已经创建好的ApplicationContext进行进一步的处理

 调用SpringApplicationRunListener的contextPrepared()方法，通知所有的监听者：ApplicationContext已经准备完毕

 将所有的bean加载到容器中

 调用SpringApplicationRunListener的contextLoaded()方法，通知所有的监听者：ApplicationContext已经装载完毕

7.调用ApplicationContext的refresh()方法,刷新容器

这里的刷新和spring中刷新原理类似，这里重点关注invokeBeanFactoryPostProcessors(beanFactory);方法，主要完成获取到所有的BeanFactoryPostProcessor来对容器做一些额外的操作，通过源可以进入到PostProcessorRegistrationDelegate类的invokeBeanFactoryPostProcessors()方法，会获取类型为BeanDefinitionRegistryPostProcessor的beanorg.springframework.context.annotation.internalConfigurationAnnotationProcessor，对应的Class为ConfigurationClassPostProcessor。ConfigurationClassPostProcessor用于解析处理各种注解，包括：@Configuration、@ComponentScan、@Import、@PropertySource、@ImportResource、@Bean。当处理@import注解的时候，就会调用EnableAutoConfigurationImportSelector.selectImports()来完成自动配置功能

8.查找当前context中是否注册有CommandLineRunner和ApplicationRunner，如果有则遍历执行它们。