**春天的杰作Spring源码分析**

主讲：Cat 老师

北京动力节点教育科技有限公司  
2020 • 北京

动力节点•版权所有•禁止传播

## 阅读源码注意事项

1、源码看第一遍基本上都会晕车的，特别是之前没有看过源码的学员；

2、一开始看源码晕车很正常，建议至少看3遍以上，然后可能才感觉不晕车；

3、看源码目的主要了解其底层实现，设计思想，拓展思维，开发时借鉴和参考；

4、不必把每一个细节都弄明白，当今社会追求高效，不必重复造轮子；

5、看源码不必有完美主义，开源代码中也有不尽人意之处；

6、看源码主要掌握核心底层实现及精华之处，在工作中遇到异常能快速定位；

## 阅读源码指导原则

1、寻找入口

2、断点跟踪

3、先粗后细

4、精略结合

5、猜想验证

注：建议先自己看一遍源码，然后再看一遍视频，然后再自己看一轮源码，源码建议看3遍以上；

3年及3年以上工作经验的学员，一定要学会看源码，什么事情开始了就开始了，不开始可能一拖再拖，一直就不会去看源码，强迫自己学会看源码；

## Spring源码分析

官网：<https://spring.io/>

Github：<https://github.com/spring-projects>

当前最新版本：5.2.6.RELEASE

Spring现在相当于是一个统称了，也称为Spring家族；（Spring family）

其下有20几个项目 <https://spring.io/projects> ，Spring Framework 只是其中一个；

2003年2月，春天，为了解决Java编程开发中的一些基础性的、重复的问题而产生的；

Sping是Java编程的春天来了；



Spring轻量级、面向Bean，只需要编写非常普通的Java bean对象；

设计优雅，松耦合，将Java中经典的设计模式运用得淋淋尽致；

Spring是个粘合剂，与主流框架无缝集成；

Java是咖啡，bean翻译为 豆子，是咖啡中的豆子，一个豆子就是一个java对象，Spring bean就是咖啡豆；

Spring的初心：**简化开发**，她始终如一地坚持这一理念；

**Spring对JDK版本要求**

Spring Framework 5.3.x: JDK 8-17 (expected)

Spring Framework 5.2.x: JDK 8-15 (expected)

Spring Framework 5.1.x: JDK 8-12

Spring Framework 5.0.x: JDK 8-10

Spring Framework 4.3.x: JDK 6-8

## Spring源码构建

1）Spring源码采用Gradle工具构建；

2）下载gradle，官网：<https://gradle.org>

3）安装gradle （需要先安装JDK）

Gradle直接解压到一个目录即完成安装；

4）Gradle配置环境变量

配置GRADLE\_HOME 和Path

1. 验证 gradle -v

验证Gradle环境是否正常；

1. 确保本机安装 JDK 8 update 60 or later 并配置环境变量.

其他JDK9、10、11、12、13我没有试过，但是我用JDK14试过，会提示兼容性问题报错；

7）当前我使用的是IDEA 2020.1.2 旗舰版

File -->Setting-->Editor-->File Encodings -->全部改为UTF-8

安装代码统计插件：Statistic

安装英文翻译插件：ECTranslation

8）Spring 5源码下载

[https://github.com/spring-projects](https://github.com/spring-projects/spring-framework/releases)

9）将下载下来的spring 5源码压缩包解压到本地某个目录下；

10）将解压后的spring 5源码导入到idea中；

具体导入到Idea中的步骤如下：

根据解压的spring5源码下import-into-idea.md文件说明来操作；

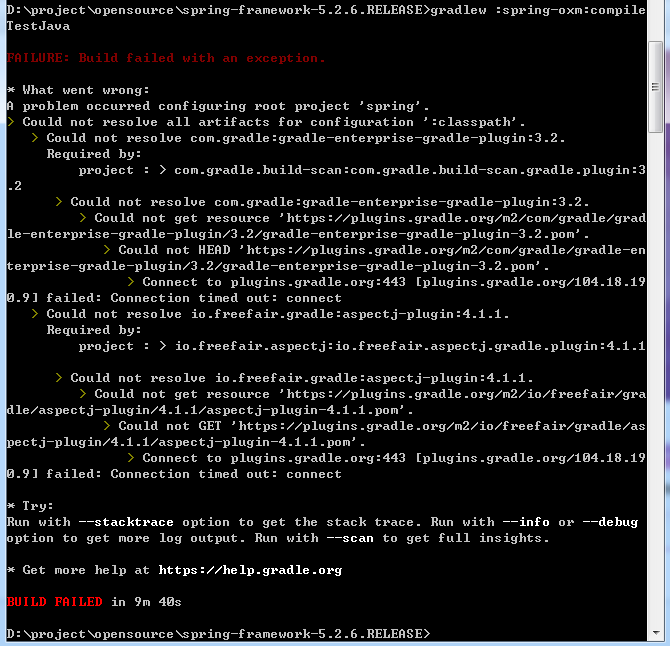
import-into-idea.md文件：

（1）切换到D:\project\opensource\spring-framework-5.2.6.RELEASE 目录下；

（2）执行 gradlew :spring-oxm:compileTestJava

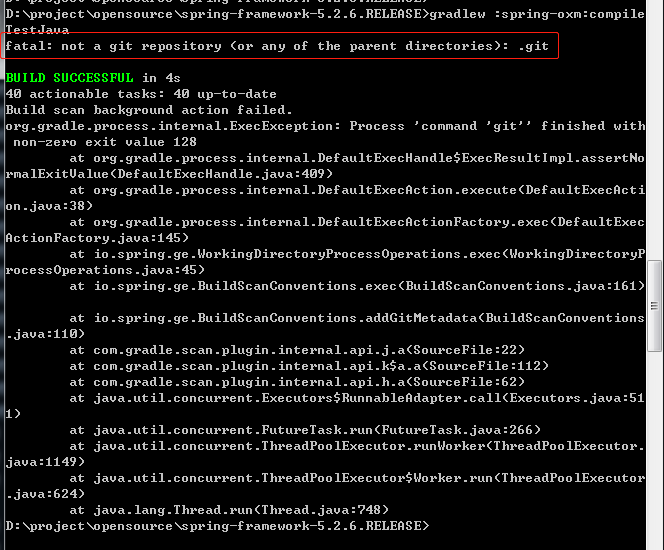
它会开始下载gradle-5.6.4-bin.zip, 默认是下载到  
C:\Users\Administrator\.gradle\wrapper\dists\gradle-5.6.4-bin\目录；

当没有使用vpn的时候，可能构建会因为网络原因构建失败，主要是连不上<https://plugins.gradle.org>到这里下载一些依赖的插件失败；如下图：



上面的错误是因为在这个过程中需要连接gradle官网下载一些gradle的插件，如果插件下载失败，就会导致构建失败；

当我们解决了网络连接问题后，构建会出现下面的错误，如下图所示：



出现上面这个错误，那么需要在   
D:\project\opensource\spring-framework-5.2.6.RELEASE 目录下使用git创建git仓库；

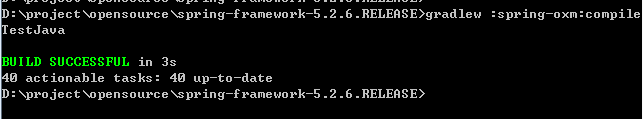
切换到D:\project\opensource\spring-framework-5.2.6.RELEASE 目录；

创建.git目录： git init

将spring源码目录加入暂存区： git add \*

提交spring源码到本地Git仓库： git commit -m‘spring’

最后再构建就会构建成功，如下图所示：

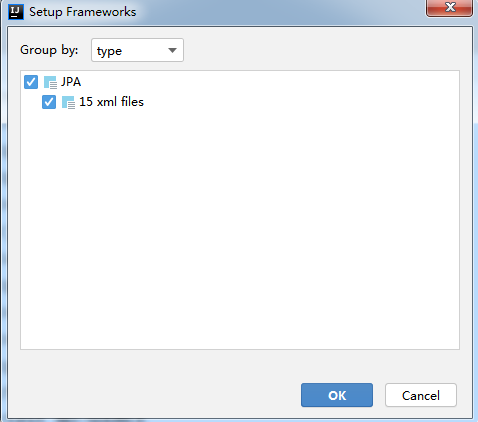


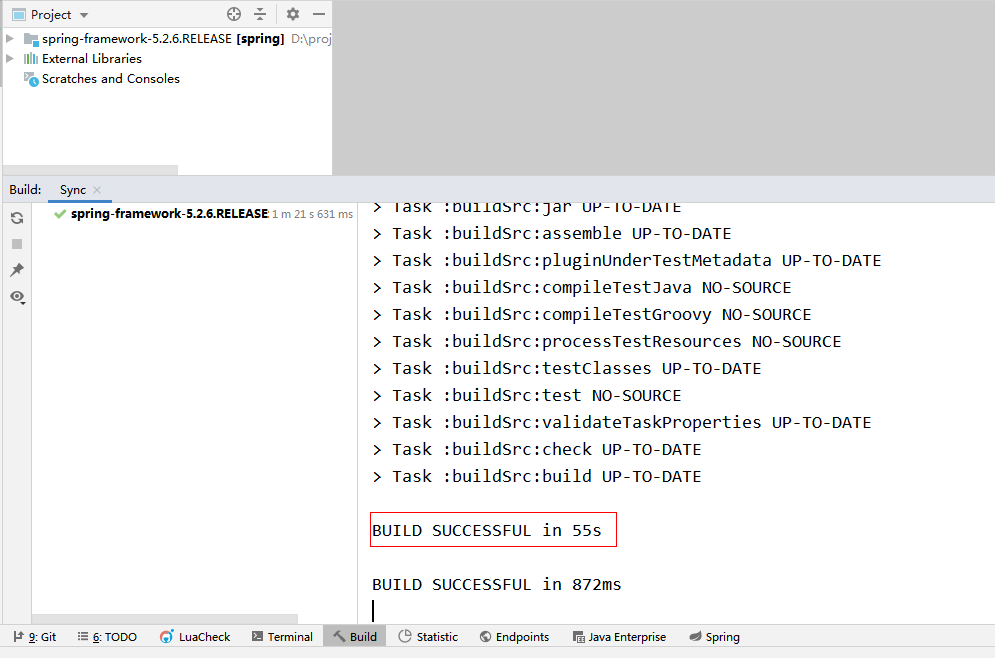
构建成功之后，我们将代码导入到Idea中；

File -> New -> Project from Existing Sources -> Navigate to directory -> Select build.gradle

导入到idea中，idea构建大概需要一会儿，请耐心等待；

然后会弹出配置web、jpa属性：





至此我们的spring源码环境就准备OK了，可以开始着手阅读源码了；

## 验证Spring源码环境是否正常

创建一个spring-z-ioc模块，编写spring代码并运行，验证spring源码环境构建是否正常；

加入spring的依赖，完整的依赖如下：

dependencies **{** testCompile group: 'junit', name: 'junit', version: '4.12'  
 compile (project(":spring-aop"))  
 *//compile (project(":spring-aspects"))* compile (project(":spring-beans"))  
 compile (project(":spring-context"))  
 compile (project(":spring-context-indexer"))  
 compile (project(":spring-context-support"))  
 compile (project(":spring-core"))  
 compile (project(":spring-expression"))  
 compile (project(":spring-instrument"))  
 compile (project(":spring-jcl"))  
 compile (project(":spring-jdbc"))  
 compile (project(":spring-jms"))  
 compile (project(":spring-messaging"))  
 compile (project(":spring-orm"))  
 compile (project(":spring-oxm"))  
 compile (project(":spring-test"))  
 compile (project(":spring-tx"))  
 compile (project(":spring-web"))  
 compile (project(":spring-webmvc"))  
 compile (project(":spring-webflux"))  
 compile (project(":spring-websocket"))  
**}**

最后运行spring-z-ioc的类测试一下spring环境是否正常？

一般情况下，第一次运行会比较慢，后面的运行会变快；

注：Spring-aspect工程里面的类不要去打开，打开后某些类可能会报错，重启一下idea即可恢复正常；

## 源码分析

1. ApplicationContext context = new AnnotationConfigApplicationContext(SpringConfig.class);

2、new AnnotationConfigApplicationContext(SpringConfig.class);--> {

2.1、this();-->{

//spring内部加载了5个BeanDefinition：就是对bean的描述

2.2、this.reader = new AnnotatedBeanDefinitionReader(this);

//放置了Compment注解

2.3、this.scanner = new ClassPathBeanDefinitionScanner(this);

}  
3、register(componentClasses); {

//把componentClasses变成bean的描述：BeanDefinition

//然后把BeanDefinition包装成BeanDefinitionHolder

//然后注册BeanDefinitionHolder里面的BeanDefinition

//注册put到Map<String, BeanDefinition> beanDefinitionMap

}  
4、refresh(); {

扫描项目中的bean（@Service、@Compment...注解的bean）得到BeanDefinition

放入beanDefinitionMap

把beanDefinitionMap里面的BeanDefinition拿出来，依次创建所对应的对象；

}

}