spring5.3.x -gradle

q 963060292

Spring 系列 生 分态 十 丰 多到 齐 是 Spring 生 [redacted]
节 理 要 解 原 理 就要 深 入 源码

本着文 述 spring 5.3.x 由 当 源码 构 建 团 [redacted]
maven ）构 建 全 gradle 很，伙伴 对 gradle 悉 所
大 把 家 spring 手 建 源码 读 环 构 建

1 git

拉取源码 使 用

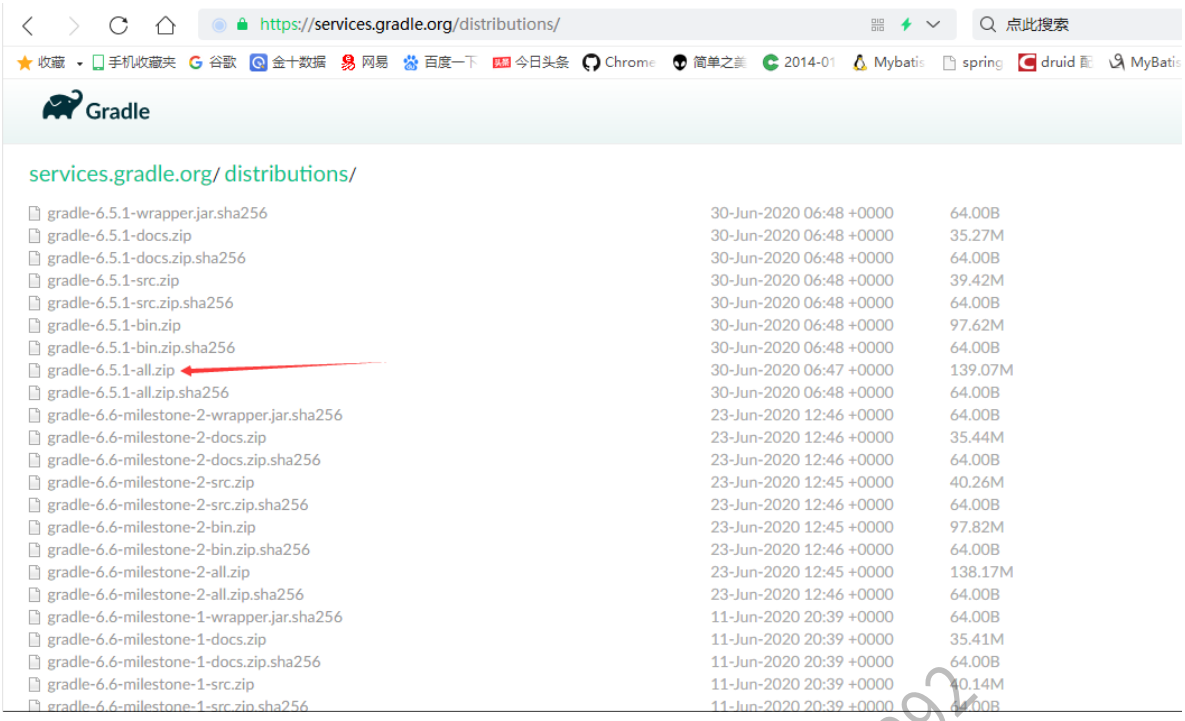
2 jdk8

已 般 伙伴 都 上 装 好 了

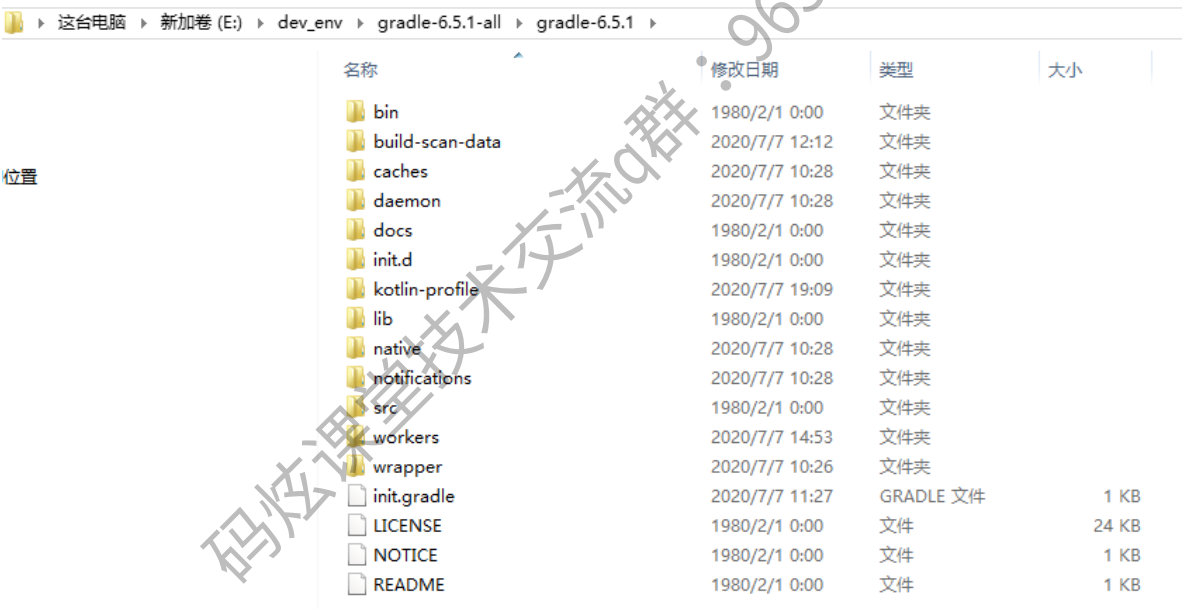
```
E:\mypro\IdeaProjects\spring-boot>java -version
java version "1.8.0_40"
Java(TM) SE Runtime Environment (build 1.8.0_40-b25)
Java HotSpot(TM) 64-Bit Server VM (build 25.40-b25, mixed mode)
```

3 gradle6.5.1

打开 https://services.gradle.org/distributions/ 选择 最新版本 gradle-6.5.1-all.zip(all 版本 源码
) 的

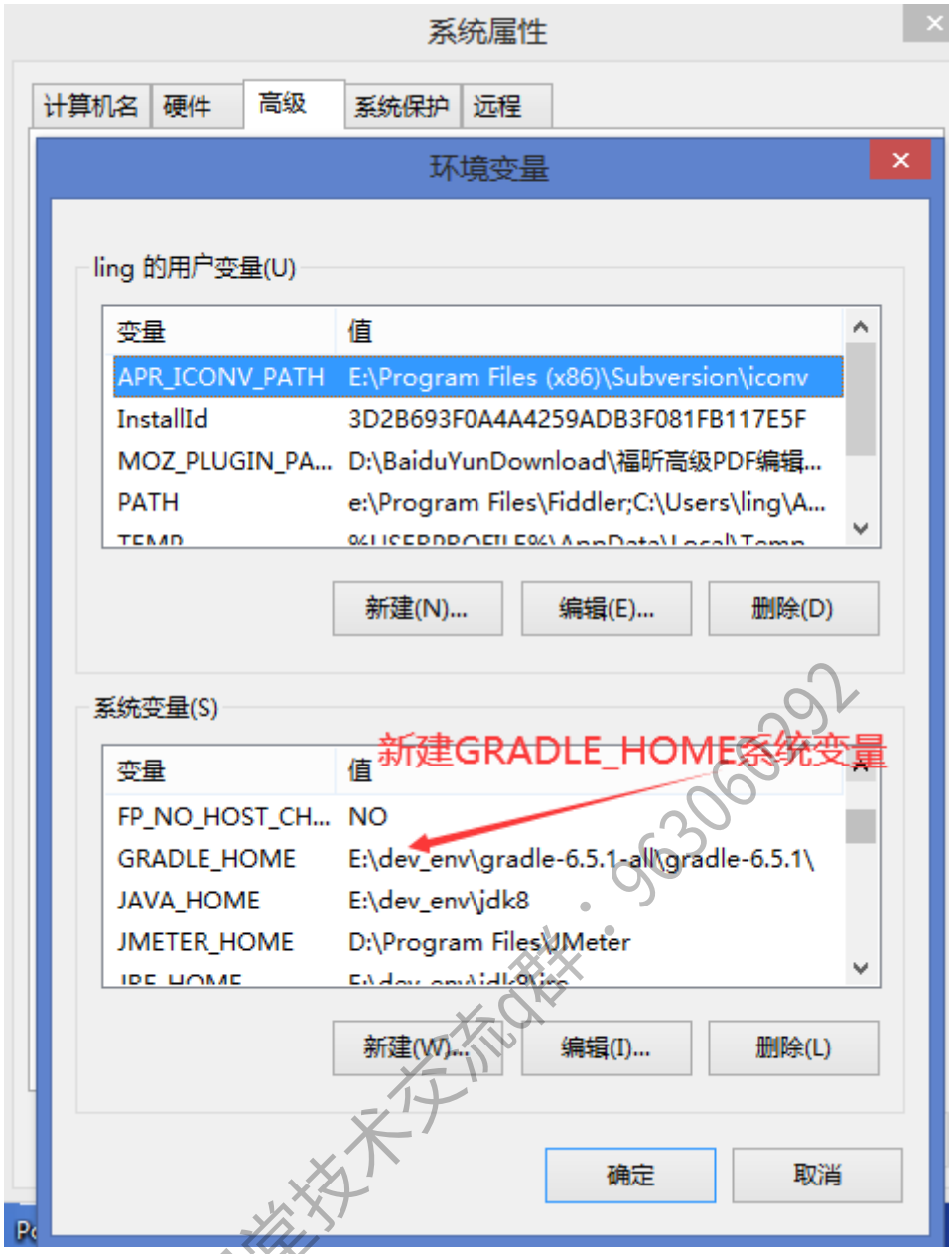


：　　解压　　录如　　下



位置

置环　境变



成后完cmd打 开 执行

```
gradle -v
```

```
E:\mypro\IdeaProjects\spring-boot>gradle -v

------------------------------------------------------------
Gradle 6.5.1
------------------------------------------------------------

Build time:   2020-06-30 06:32:47 UTC
Revision:     66bc713f7169626a7f0134bf452abde51550ea0a

Kotlin:       1.3.72
Groovy:       2.5.11
Ant:          Apache Ant(TM) version 1.10.7 compiled on September 1 2019
JVM:          1.8.0_40 (Oracle Corporation 25.40-b25)
OS:           Windows 8.1 6.3 amd64

E:\mypro\IdeaProjects\spring-boot>
搜狗拼音输入法 全 :
```

表 示 成 功 6版.本 为

4 idea2020.1.2

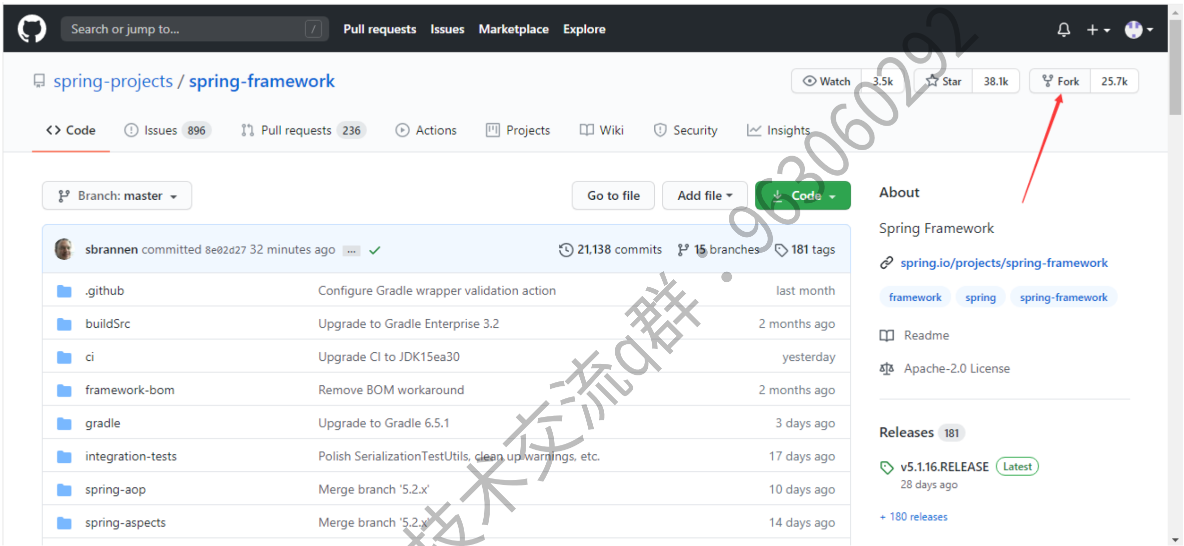
(很 idea多0 之 版前本导入 始 的 建议02版到 s版a本 , 哥 使 前) 2是用就2 版 本 的

从官方仓库 https://github.com/spring-projects/spring-framework `Fork` 出属于自己的仓库

- 为什么要 `Fork`？既然要阅读、调试源码，我们可能会写一些注释，有了自己的仓库，可以进行自由的提交。
- 本次使用的 Spring 版本为 `5.3.x` 分支（master 的为 5.3.0-SNAPSHOT 版本）
- 使用 `IntelliJ IDEA` 从 `Fork` 的仓库中拉取代码，因为从仓库拉取 Spring 的代码比较大，所以会比较慢，我们通过 clone 命令拉取，然后用 idea 再打开的

具体过程如下：

1、打开 https://github.com/spring-projects/spring-framework 点右上角 Fork 按钮，这样就把 spring 中的仓库 Fork 到自己的仓库中了

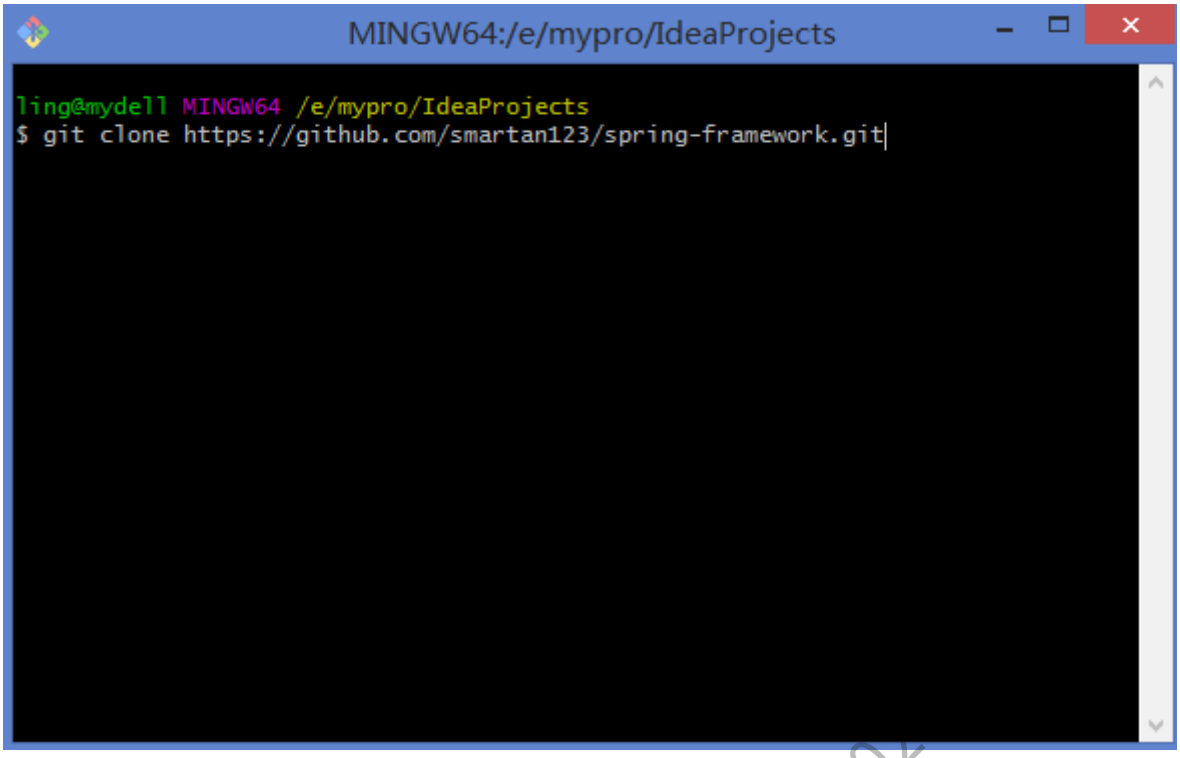


2、选择一个目录，比如 D:\mypro\ideaProjects 目录中，右键 Git Bash Here

执行

```
git clone https://github.com/smartan123/spring-framework.git
```

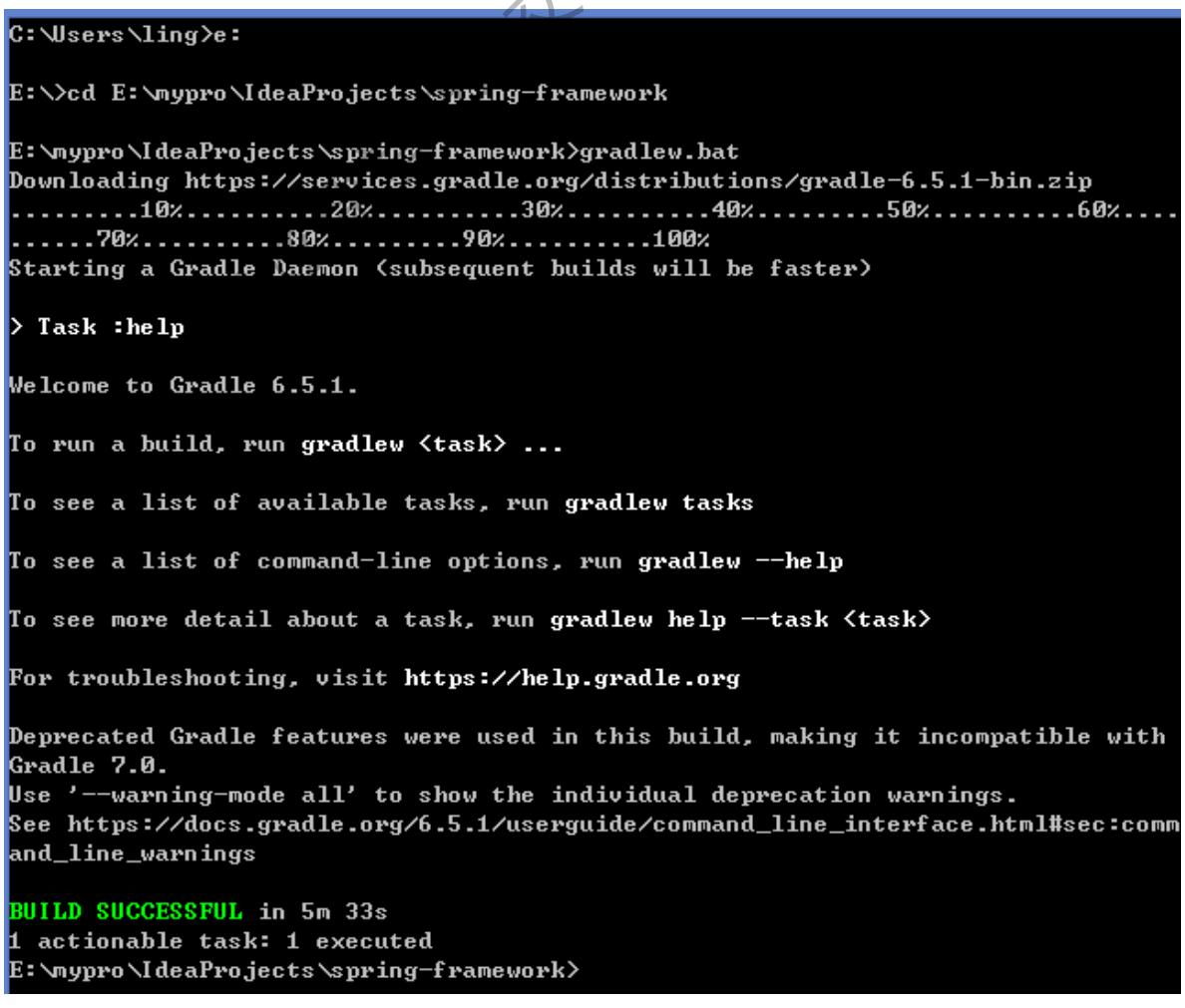
耗时大概



1 因 构之建 先gradle 蒙 spring ,gr是dle 为 构 建 的

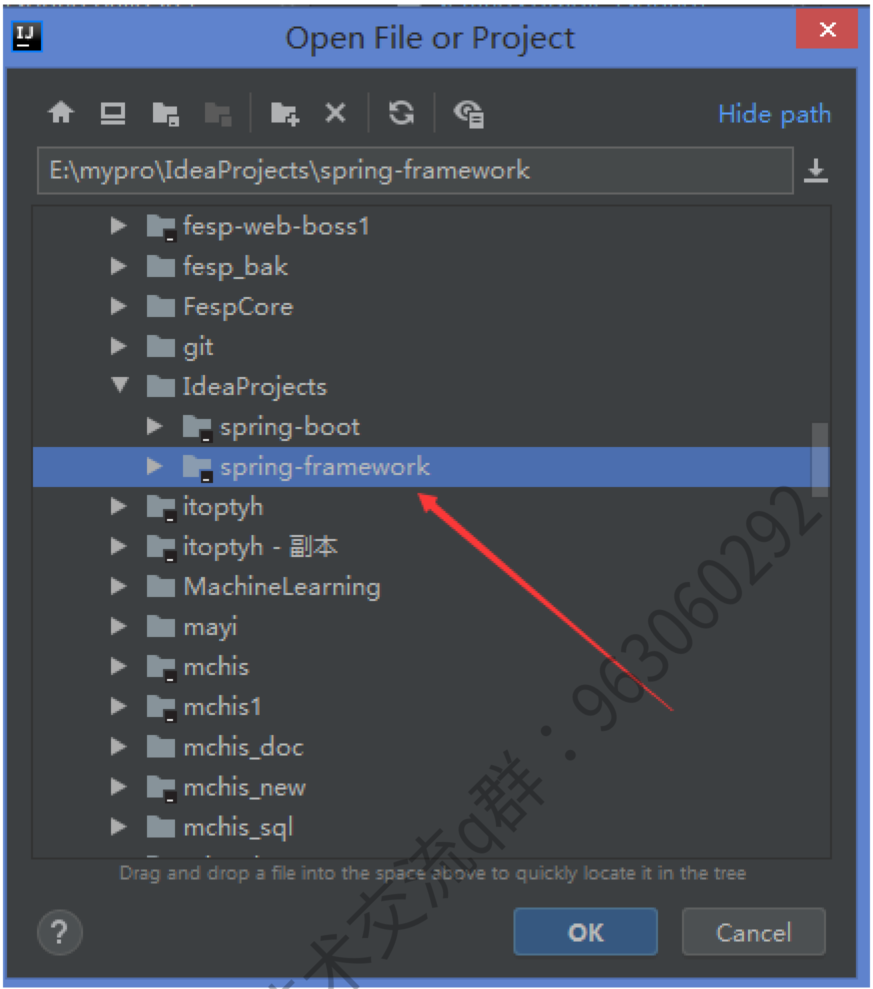
cmd在 进 入源码 根 录目 输gradlew.bat 命令自将脚 本 gradle 载下-6.5.1-bin.zip 包

这 （ 其 实步 可中 省 ） 将 直接 源码 导入 以

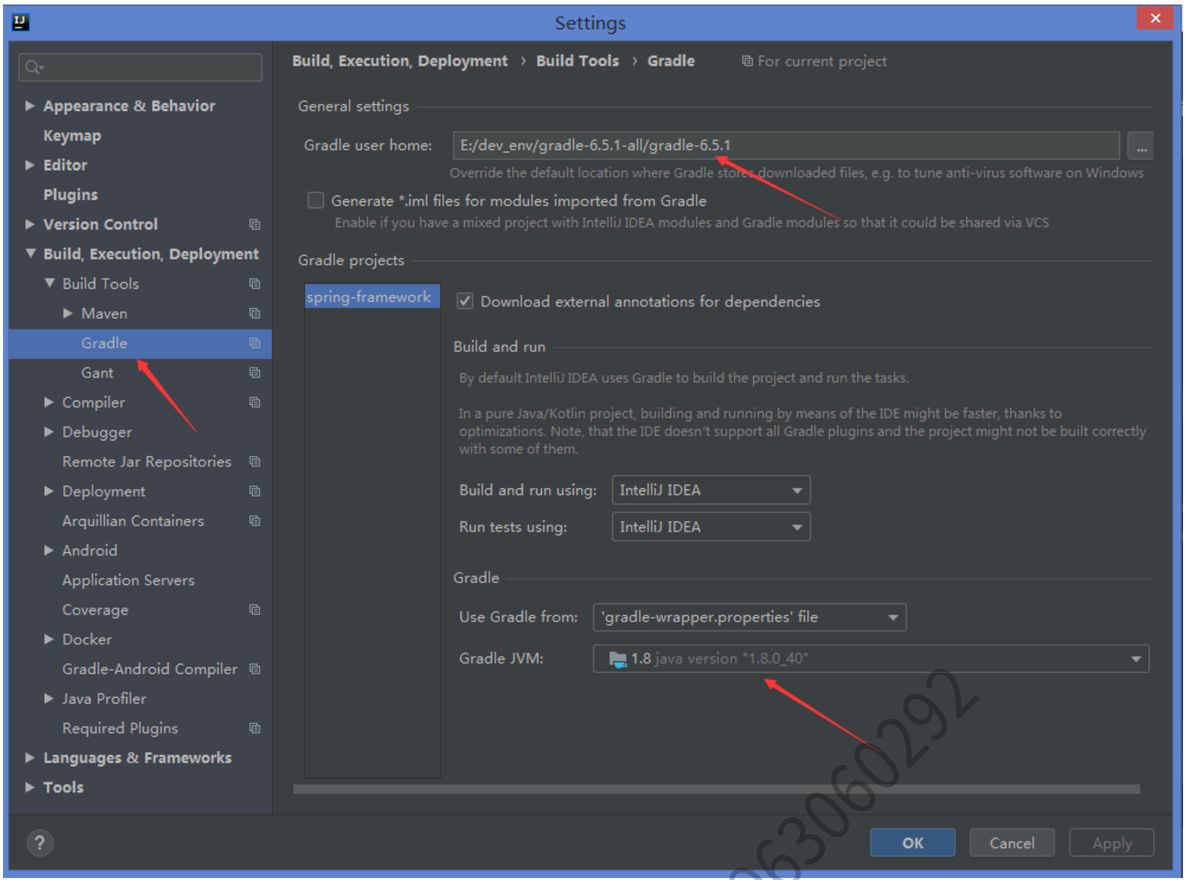


2 ID打开, 开open 直接 源码 目

->文件 Open ， 选 择源码 根 录目 ，



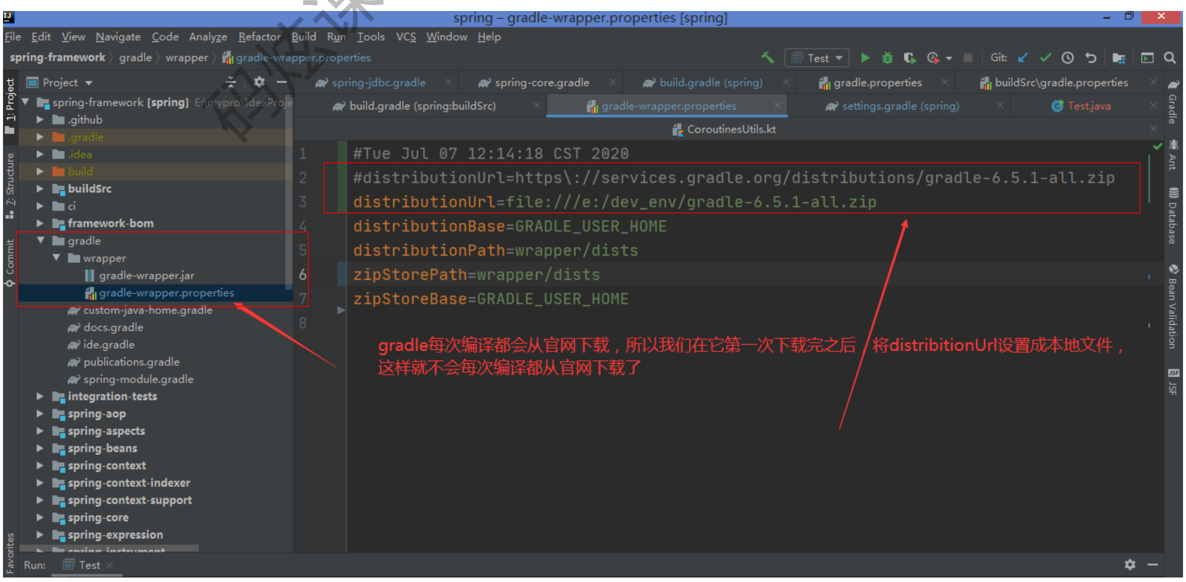
3 打 开后 File -> Settings Gradle 设置 Gradle user home 这个 目录 我 设置 gradle 安装 目录 如果 设置 的 话 C:\Users\ling\.gradle 它 这个， 目录 就可以 相 于 是 我们 本地 maven 的 仓 gradle 的 ,编译 所 依赖 的 会 放 下载 到 这个 目录 以 这 jdk8 就 jdk 这 里 就



4、设置完之后，打开工程下gradle->wrapper目录下的gradle-wrapper.properties文件，gradle每次编译都会从官网下载gradle-6.5.1-all.zip，所以我们在第一次下载完之后，将distributionUrl设置成本地文件，这样就不会每次编译都从官网下载了

```
distributionUrl=file:///e:/dev_env/gradle-6.5.1-all.zip(这里选择gradle的压缩包的全路径地址)
```

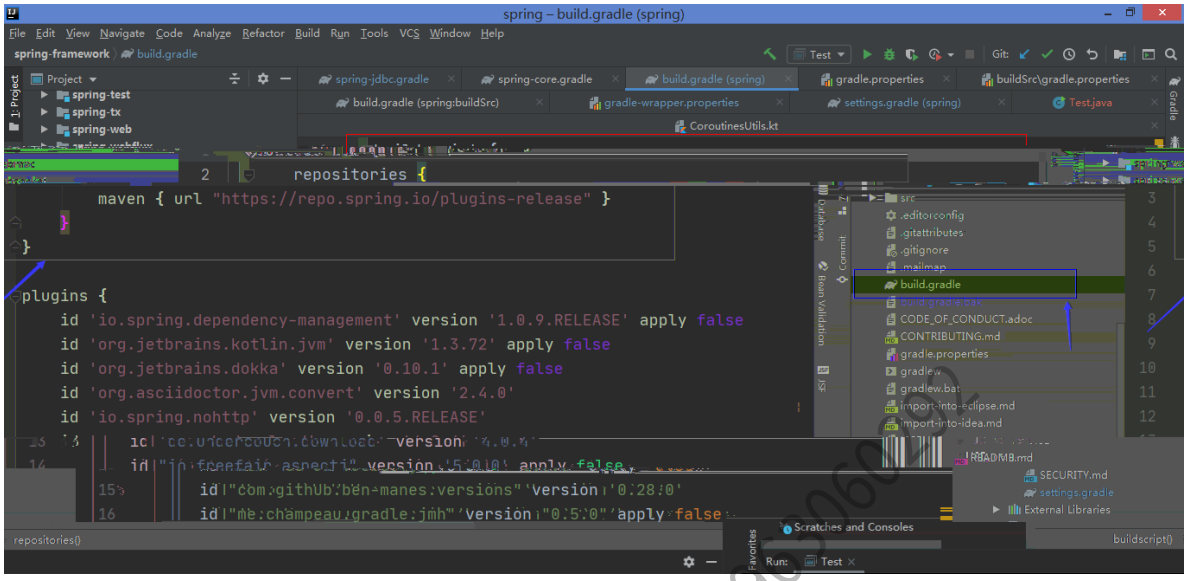
建议(idea用2020版本最新的，打开编译省去不少麻烦)的



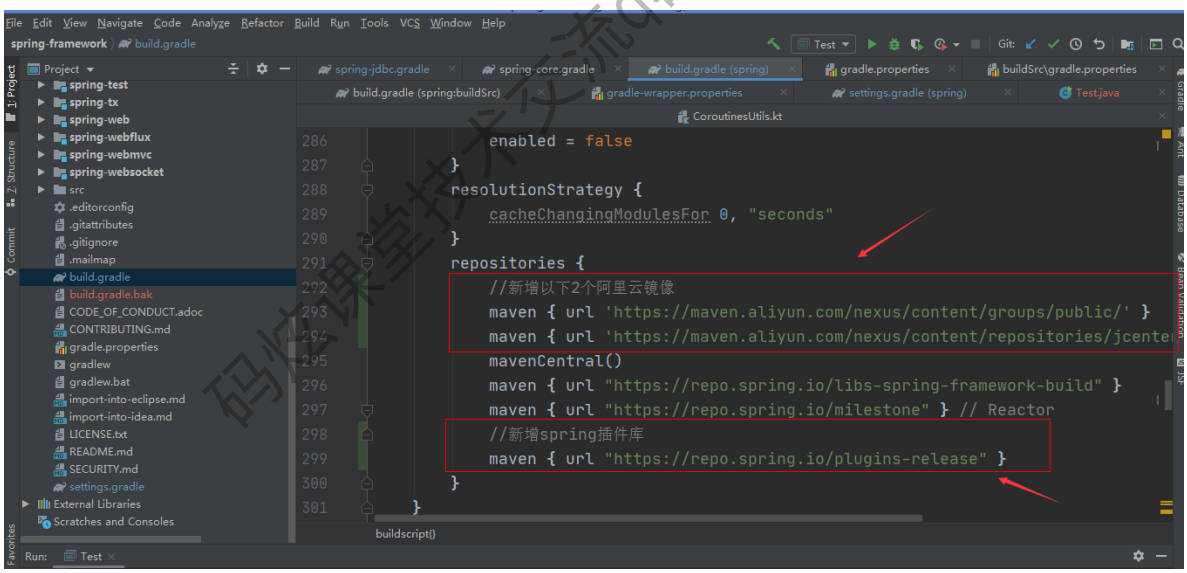
5、打开build.gradle文件，这个文件相当于maven的pom文件，在文件头部加上

```
buildscript {
    repositories {
        maven { url "https://repo.spring.io/plugins-release" }
    }
}
```

如 下



6 ： 然 后 往 继续下我 找到此 码 段



色 红 线 是 部 加 要 添 要 地 添 的 云 库 这 ， 我 们 置 的 maven 是 样 时 就 依 赖 的 度 载 下 如 果 ， 置 像 配 话 会 编 译 个 的 小

体 具 修 如 下 改

```
repositories {
            //新增以下2个阿里云镜像
            maven { url 'https://maven.aliyun.com/nexus/content/groups/public/'
}
            maven { url
'https://maven.aliyun.com/nexus/content/repositories/jcenter' }
            mavenCentral()
            maven { url "https://repo.spring.io/libs-spring-framework-build" }
            maven { url "https://repo.spring.io/milestone" } // Reactor
            //新增spring插件库
            maven { url "https://repo.spring.io/plugins-release" }
}
```

修 偏 后会 改 动 的 建

: 等待 定 后 构 建 完 注 时 (意 是 , 把 依 赖 ! 载 , 其 还 实 没 正 真 编 译

如果 构 建 失 败 es 新 次 新 般是 试 基 类 误 时 的

7 构 建 之 后 `ApplicationContext` 类

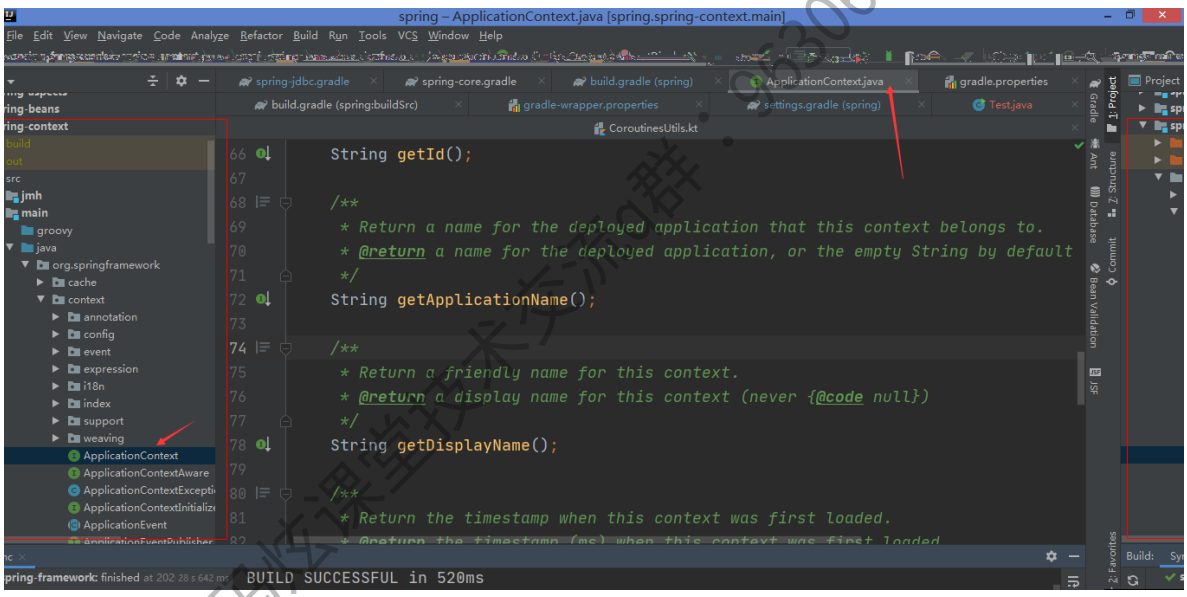
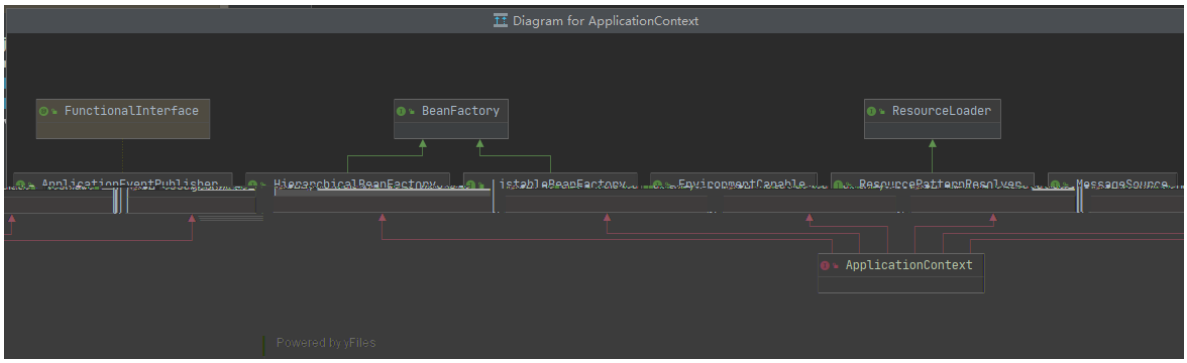


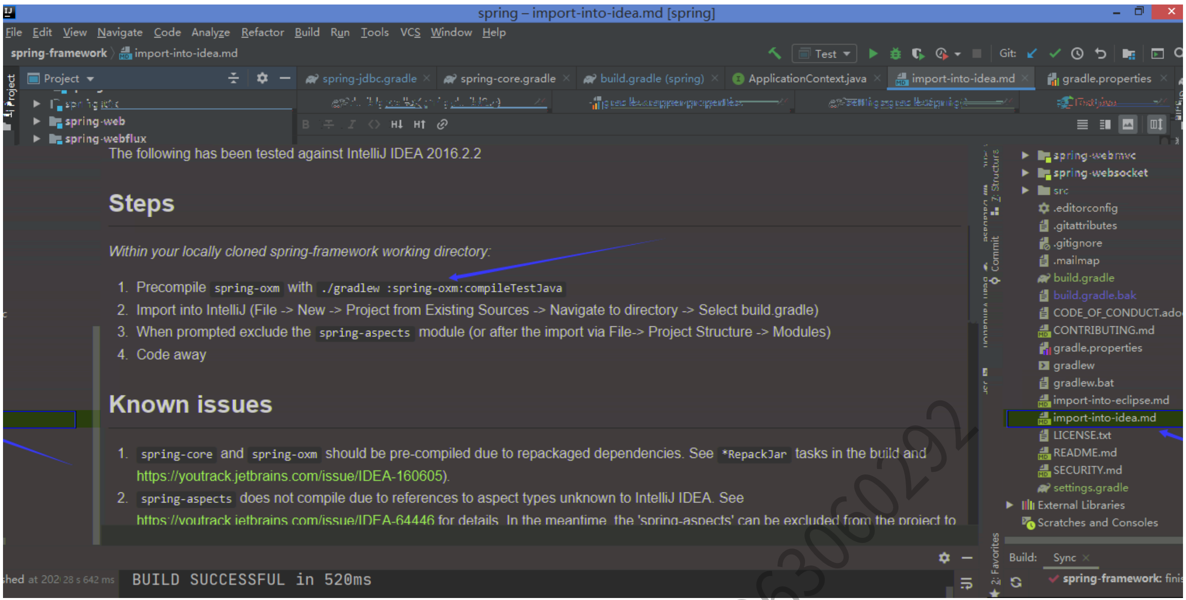
图 后 Alt+U 下 明 如果 , 现 所 示 类 说 构 建 功 构 建 是 就 对 象 !
) 程 过



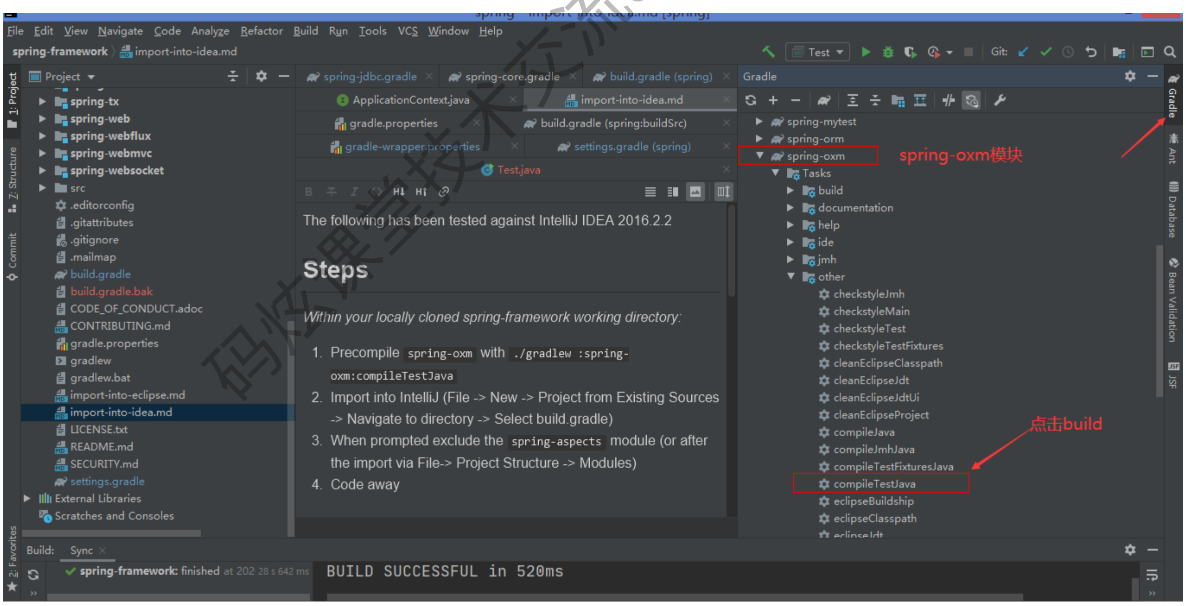
此 Spring 源 以 是 我 们 要 的 源 础 基 进 上 修 发 开 调 加 添 注 等 等
操 作 所 将 要 源 进 编 译 包 是 就 源 编 译 程 过 的

1 在构建完成之后，就搭建好了读源码的环境，此我们还将要对源码编译打包

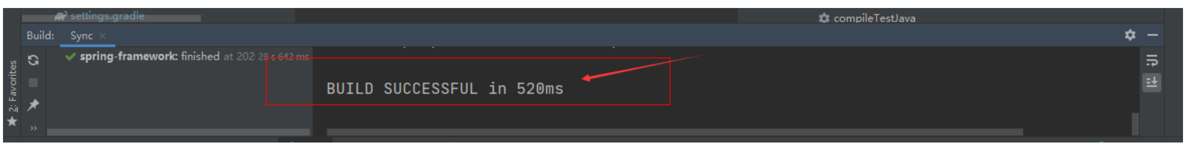
在编译之前需要进行一些修改配置，可查看import-into-idea.md文档



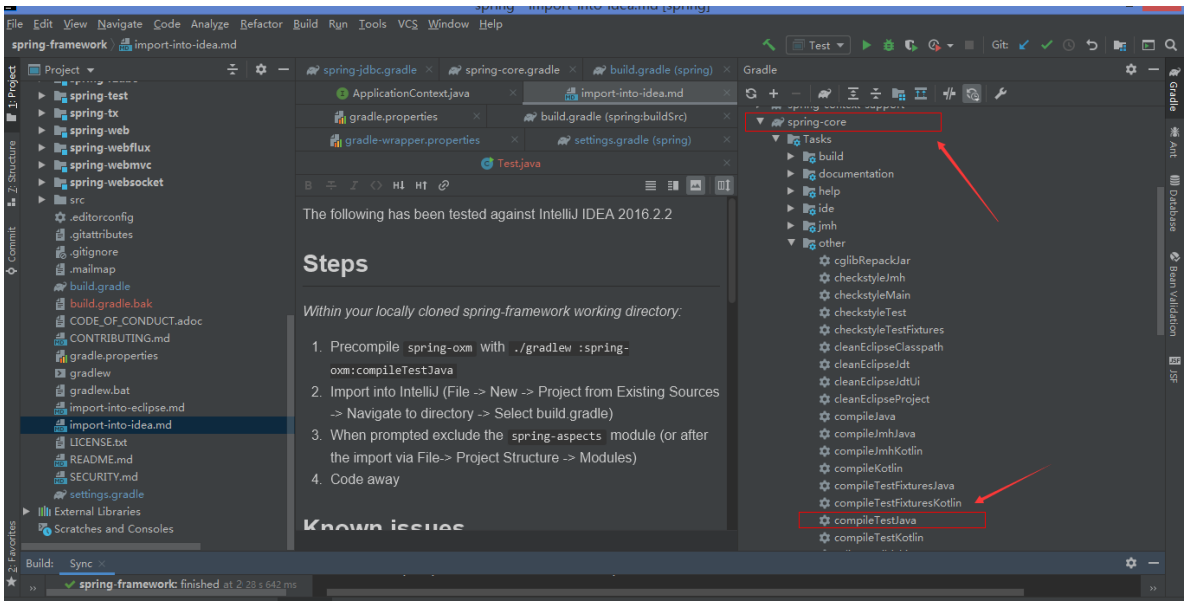
2 文档要求先编译spring-oxm下的compileTestjava，点击右上角gradle打开编译视图，找到spring-oxm模块，然后在other下找到compileTestjava，双击即可！



编译成功！

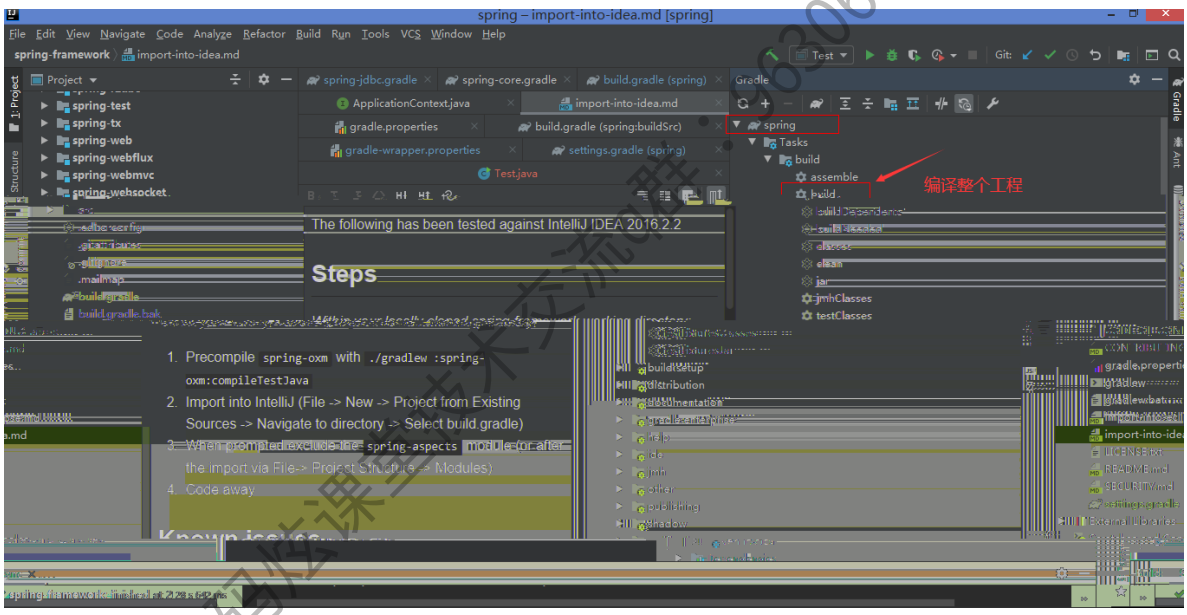


3 保因起编译spring-core下模块之后，因为spring-context依赖core的方法，上

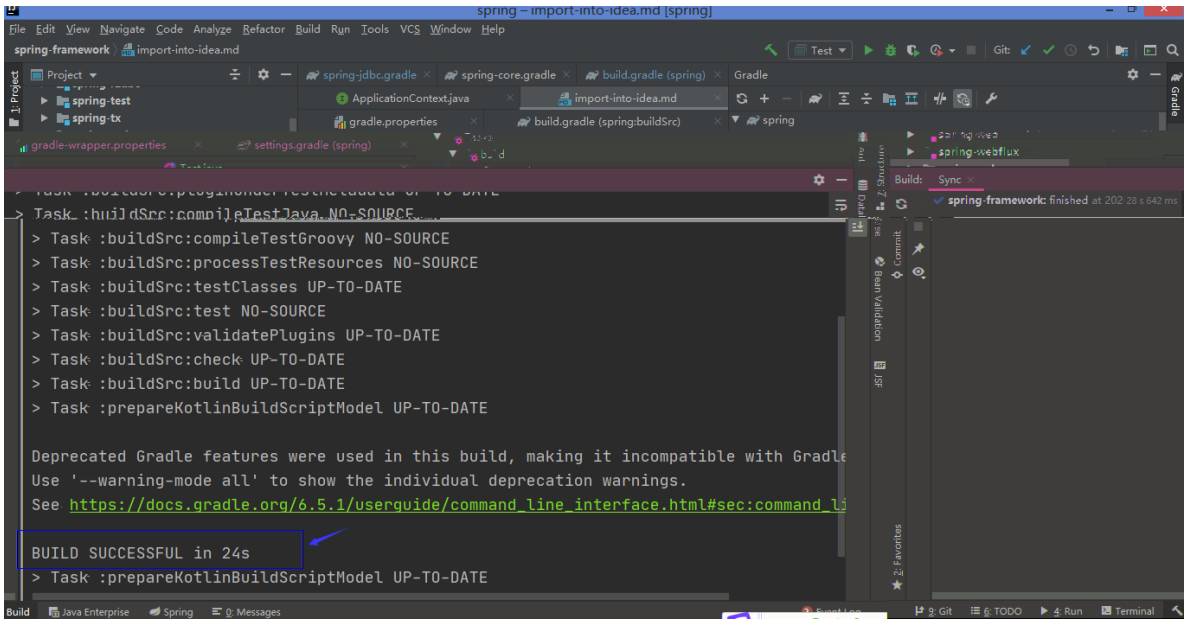


4 都编 诫 成且 之后 嫩 译 个程 这 程(图) 耗 10-20 能分 常 时 如 ， 下

打 spring->build->build



电 经过段 编uild 成，功 时 每 (人脑 性下)能 样所 ，不的 样 时

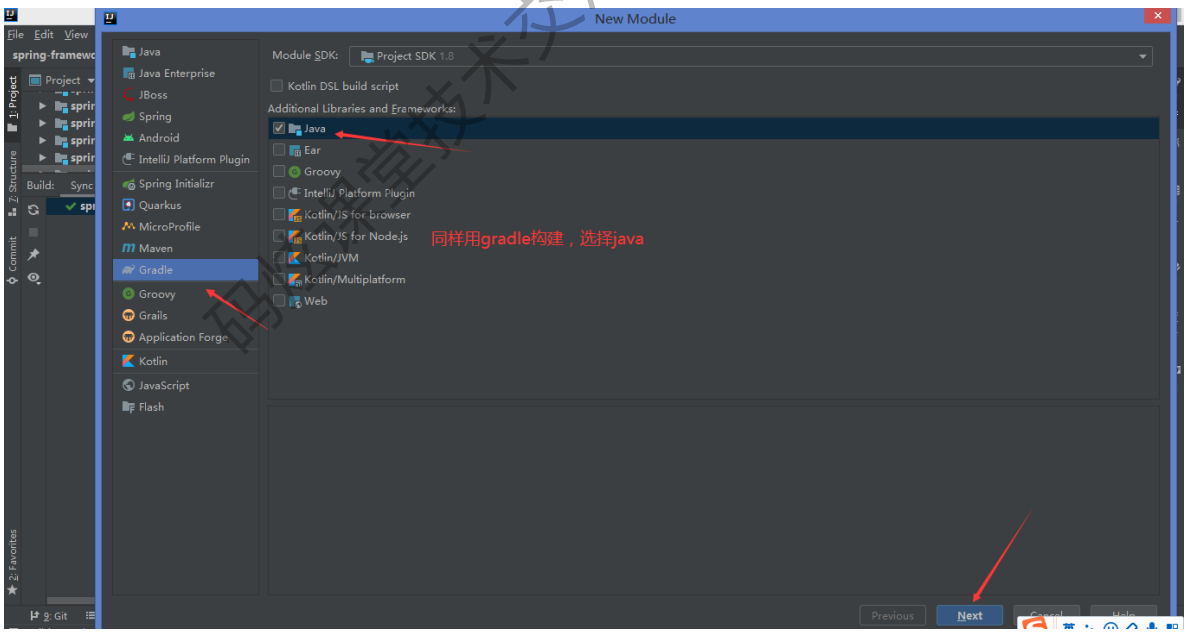


中 smart 在编写程过 发现 中 ,利 是 在 测下试发现 的 了

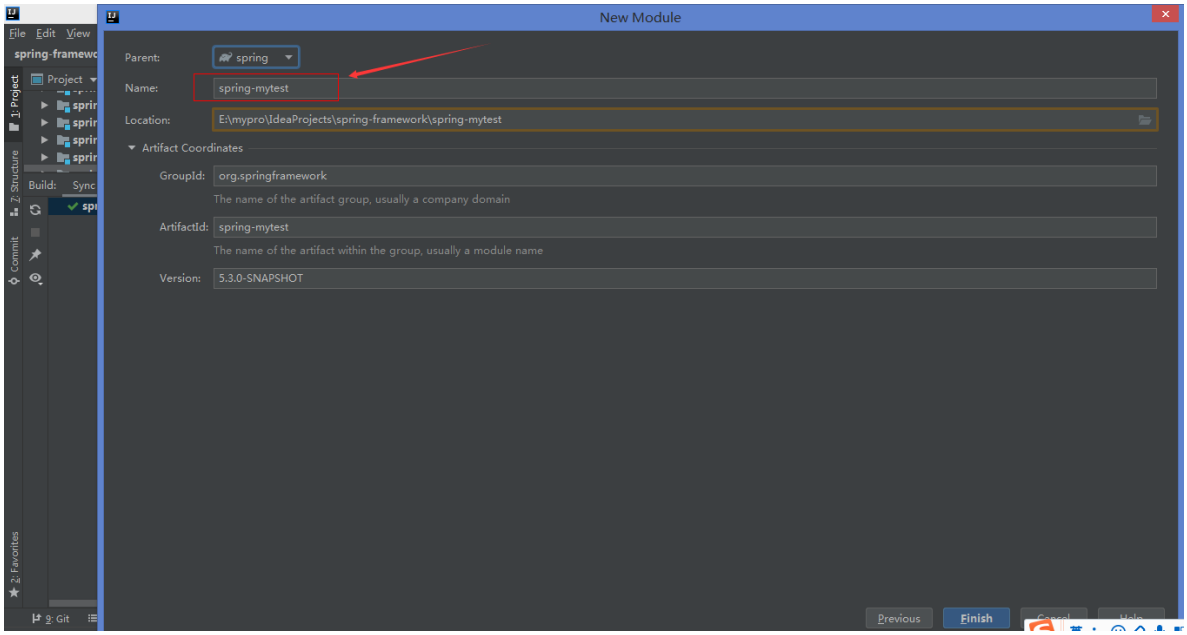
1 成 完 程后 们了 可 编 写模 测块以试该 程是过 否成正 试完

File ->步New -> Module...

Spring在己 自加添 module 模 块 gradle构建 构 建

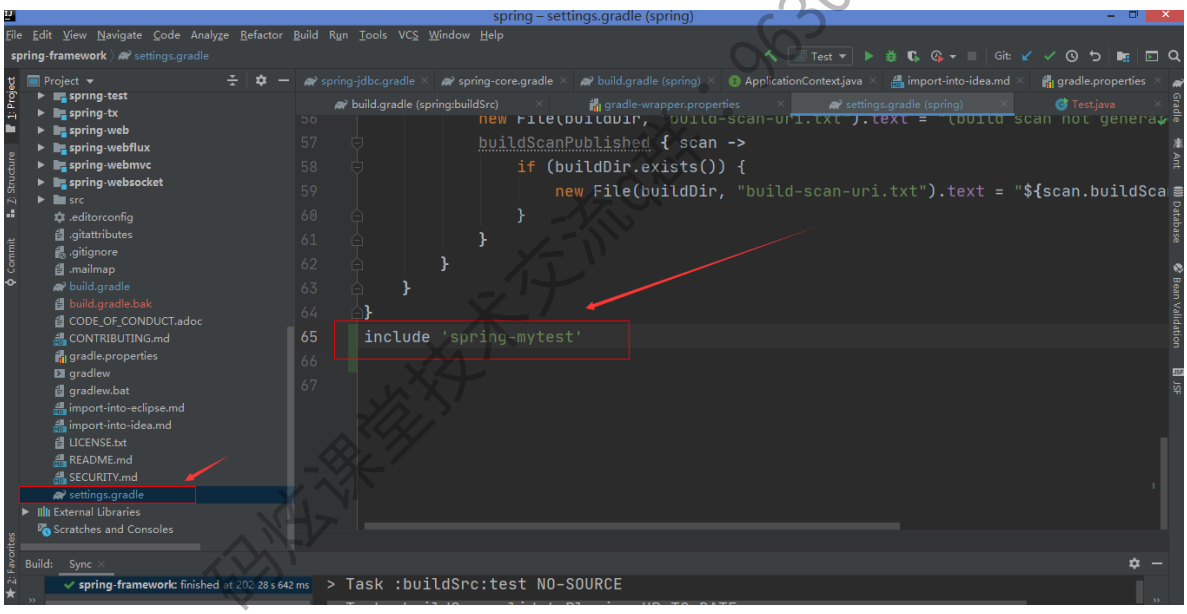


: 2 Parent块 spring Name spring-mytest 我们测(模 选择spring 测为 模 加添 进) 来 , 下置 认配 可 的 即



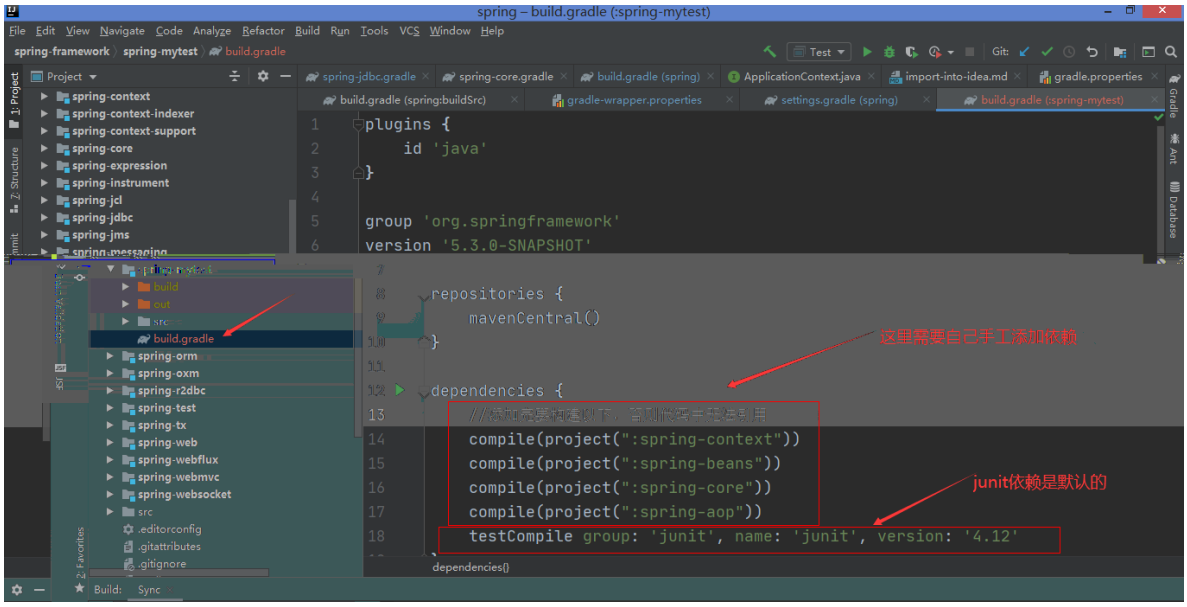
3 Finish idea自 会， 动 我们构建 spring-mytest 模 块

打 全开局 置件 settings.gradle 件 文 拉到最 我们看 到 统加 spring-mytest 模 块 了

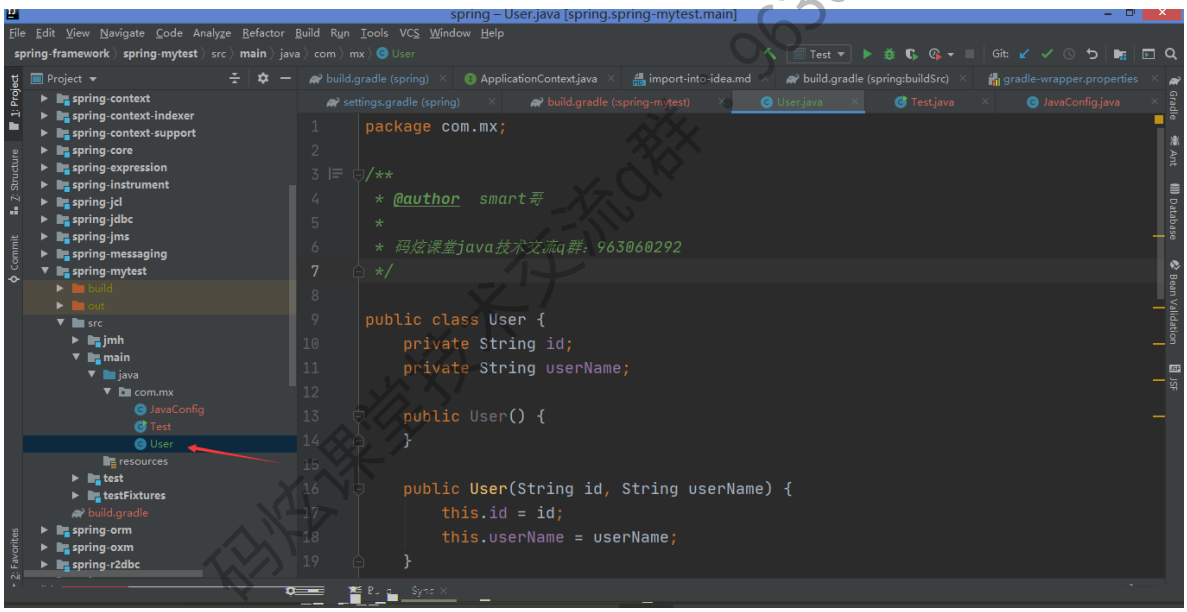


己 自找 我们 测 spring 模 块 mytest 的 build.gradle ） 件 文 相 是（ 于 件 文 当，dependencies （依 赖 这 dependencies maven 和 的 依 是 赖 ） 样 有 unit 个 的 我们 的 要 添 ：加 spring-context spring-beans、 spring-core 、spring-aop 、4 这 个 核 模 心 块 体 具 如 下

```
dependencies {
    //添加完要构建一下，否则代码中无法引用
    compile(project(":spring-context"))
    compile(project(":spring-beans"))
    compile(project(":spring-core"))
    compile(project(":spring-aop"))
    testCompile group: 'junit', name: 'junit', version: '4.12'
}
```



4 编下写个简单的ApplicationContext 获取容器bean 用是测试的源码构建编译过是成否功！



新建个体实java类

```
package com.mx;

/**
 * @author  smart哥
 *
 * 码炫课堂java技术交流q群：963060292
 */

public class User {
    private String id;
    private String userName;

    public User() {
    }
```

```
    public User(String id, String userName) {
        this.id = id;
        this.userName = userName;
    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    @Override
    public String toString() {
        return "User{" +
                "id='" + id + '\'' +
                ", userName='" + userName + '\'' +
                '}';
    }
}
```

新建JavaConfig.java（使用注解的方式注入bean）

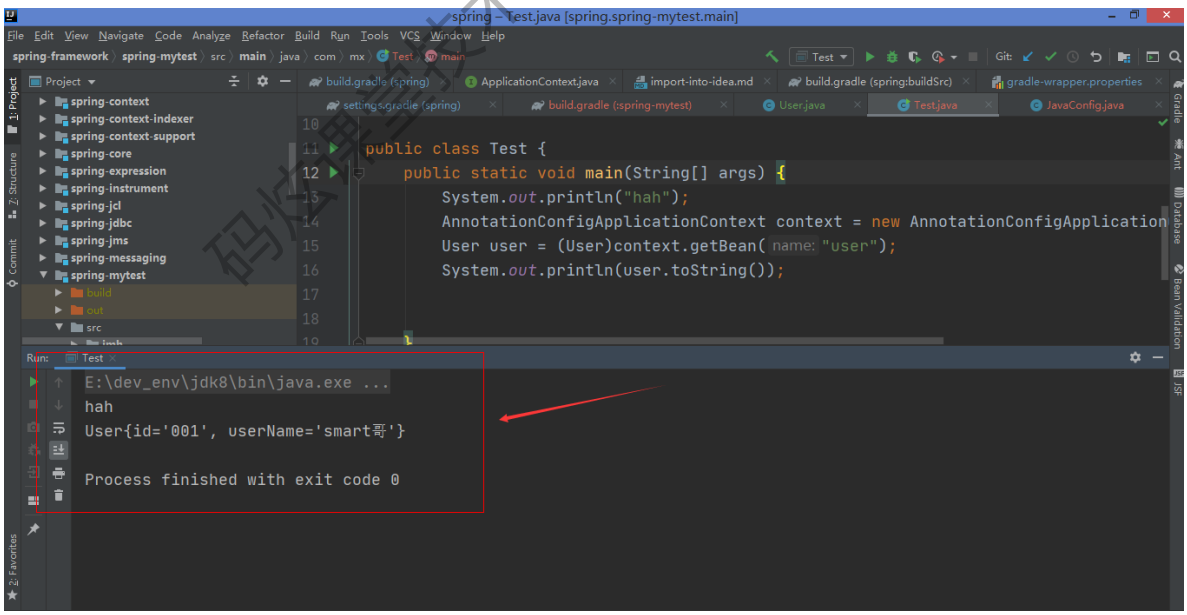
```
package com.mx;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

/**
 * @author  smart哥
 *
 * 码炫课堂java技术交流q群：963060292
 */

@Configuration
@ComponentScan
public class JavaConfig {

    @Bean
    public User user(){
        return new User("001","smart哥");
    }
}
```

后最　写测Test试java 类

```
package com.mx;

import
org.springframework.context.annotation.AnnotationConfigApplicationContext;

/**
 * @author  smart哥
 *
 * 码炫课堂java技术交流q群：963060292
 */

public class Test {
    public static void main(String[] args) {
        System.out.println("hah");
        AnnotationConfigApplicationContext context = new
AnnotationConfigApplicationContext(JavaConfig.class);
        User user = (User)context.getBean("user");
        System.out.println(user.toString());

    }
}
```

出　　co运行ole　，　输

```
hah
User{id='001', userName='smart哥'}
```



成　功　撒 花　　！　！！

1　CoroutinesUtils

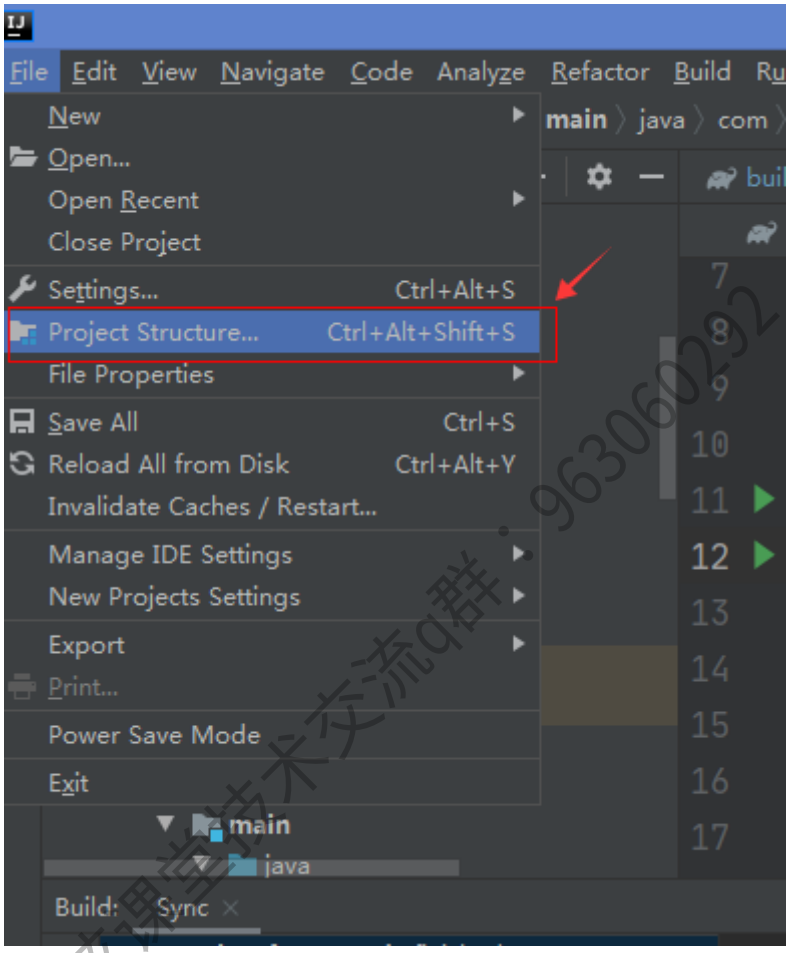
：

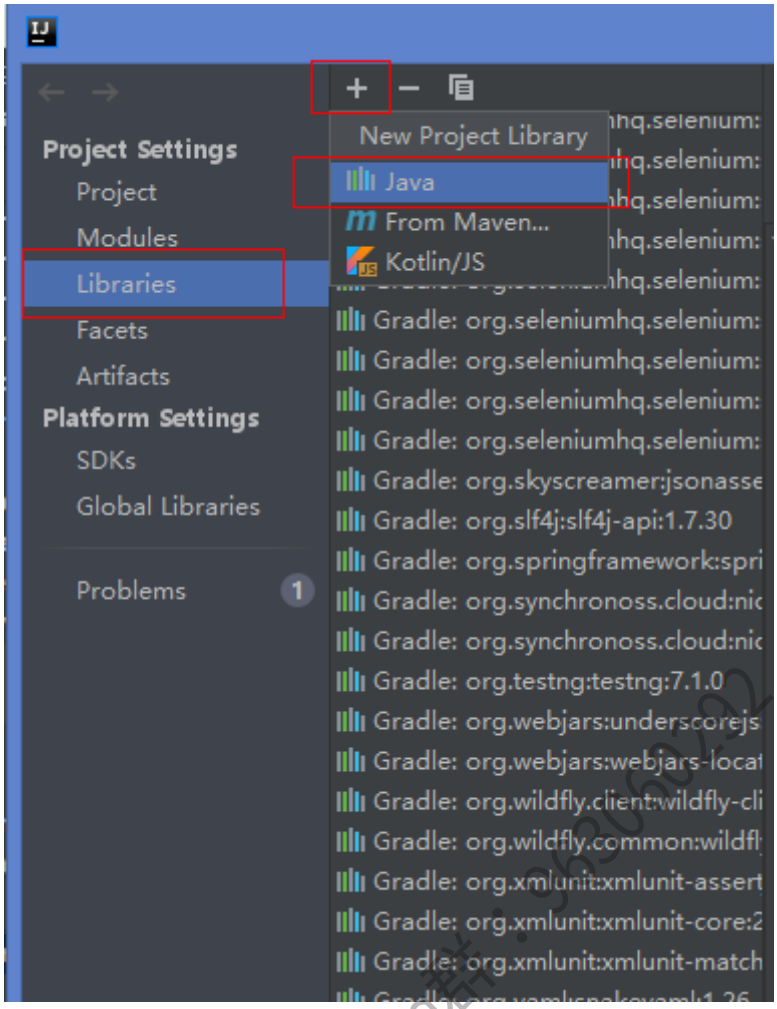
Error:(354, 51) java: 不找 到符 号

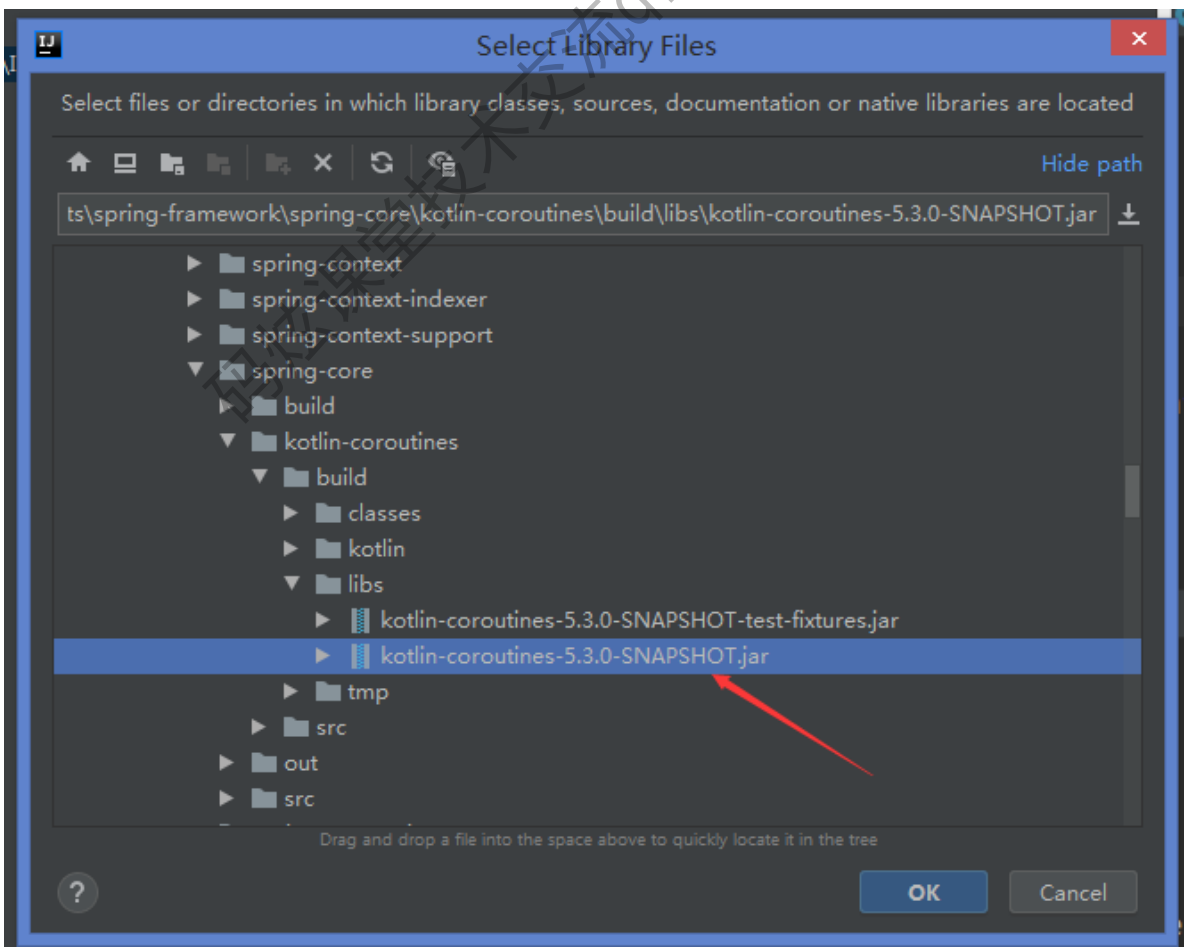
： 符 C号 变 tinesUtils

： 位 org置 spring 类 framework.core.ReactiveAdapterRegistry.CoroutinesRegistrar

File 点 ->击 Project Structure -> Libraries -> + -> Java 然 后spring选 择 framework/spring-core/kotlin-coroutines/build/libs/kotlin-coroutines-5.3.0-SNAPSHOT.jar 出 在 ， 弹 对 话 框 spring选 择 core.main 的 build 两 目 可 即







2 InstrumentationSavingAgent

Error:(26, 38) java: 找不到符号
　　符号: 类 InstrumentationSavingAgent
　　位置: 程序包 org.springframework.instrument

修改spring-context模块下spring-context.gradle文件，找到optional(project(":spring-instrument")) 将optional改为compile

```
//optional改为compile，否则报错：找不到InstrumentationSavingAgent
//optional(project(":spring-instrument"))
compile(project(":spring-instrument"))
```

3 H2DatabasePopulatorTests > executesHugeScriptInReasonableTime() FAILED

H2DatabasePopulatorTests > executesHugeScriptInReasonableTime() FAILED

修改spring-jdbc模块下spring-jdbc.gradle文件，找到optional("com.h2database:h2") 将optional改成compile

```
//报错：H2DatabasePopulatorTests > executesHugeScriptInReasonableTime() FAILED
//解决方案：将optional换成compile
//optional("com.h2database:h2")
compile("com.h2database:h2")
```


4 header.mismatch [SpringHeader]

该错误是修改完3个错误后，对整个工程进行重新编译时报的，这个是我们新建的spring-mytest模块报的

这个其实无关紧要，是格式问题，可以忽略不管，也不影响编译，我们可以直接执行Test.java的main方法

如果某些伙伴有强迫症，一定要全部编译成功才能停手，可以尝试修改全局配置文件，在编译的时候把spring-mytest模块去掉，有兴趣的可以尝试一下

小伙伴们如果在编译过程中遇到相关问题，可以在**java**技术qq群 **963060292** 中找我一起交流，共同探讨研究

spring

spring-core: 核心模块 依赖注入 IOC 和 DI 的最基本实现
spring-beans:Bean 工厂 装配
spring-context: IOC 容器 上下文 即
spring-context-support: IOC 对 扩展 IOC 及 容器子 的 以
spring-context-indexer: 管理 类 Classpath 和 扫描
spring-expression: 表达式语句

向 spring-aop: 切面 CGLIB,JDKProxy
spring-aspects: 集成 AspectJ Aop 应用 , 框架
spring-instrument: Class 动态 Loading 模 块

spring-jdbc: 提供 JDBC 主要 模块 JDBC 简化 操 作
spring-tx:spring-jdbc 事务 管理
spring-orm: 主要 Hibernate,jpa,jdo 等
映 spring-oxm: 将 java 对象 映射 成 数据 或 将 java 对象 为
spring-jms: 发 送 和 接 收 消息

web

spring-web: 提 供 最 基 础 web 支 持 , 主 要 建 立 在 核心容 器 的 上
spring-webmvc: spring mvc 实现 了 web 应 用 的
spring-websocket: 主 要 与 前 端 全 双 通 讯 协 议 的
spring-webflux: 一 个 新 的 非 阻 塞 函 数 式 Reactive Web 的 框 架

spring-messaging:4.0 加 入模 块 主 要 基 础 报文 发 送 用

spring-test 测 试 组 件

spring-framework-bom: 解 决 不 同 模 块 依 赖 版 本

smart **3 1** 到 -- 之 - **spring5.3.x** 最新 程 将 课 开 启 全 解 析 即
spring5.3.x 底层 源码 来 , 带 你 , 来 一 再 时 !

? ~~ 群 在 哪

java Q 963060292