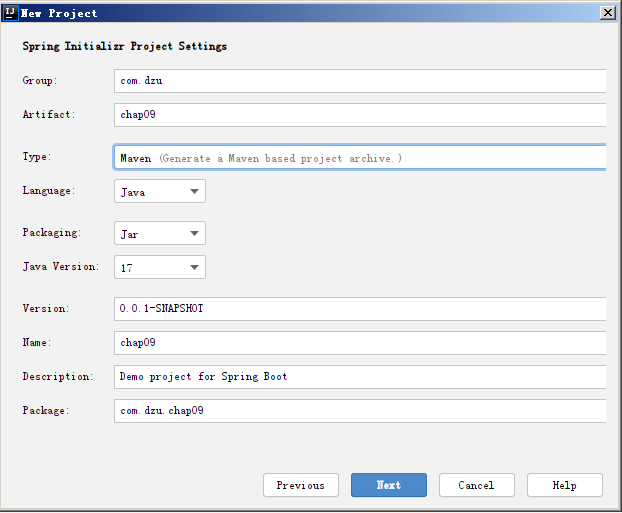
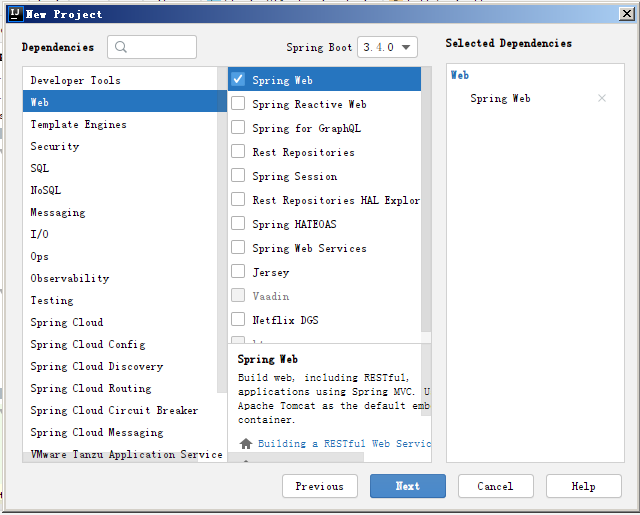
# 实验十八 Spring Boot异步任务管理

一、无返回值异步任务调用

1．创建项目

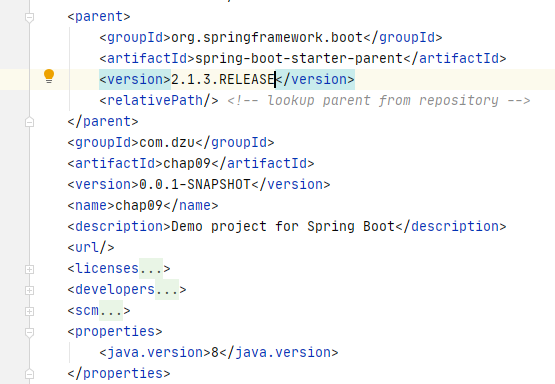
（1）使用Spring Initializer方式创建项目，选择web依赖。





（2）修改pom.xml文件

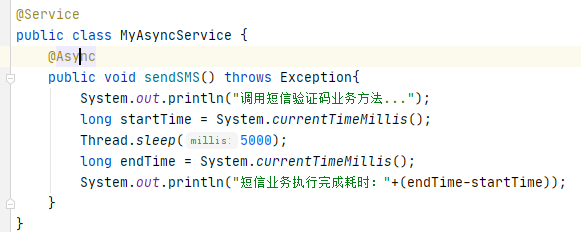
修改父依赖版本和Java版本。



（3）设置本地仓库

2．编写异步调用方法

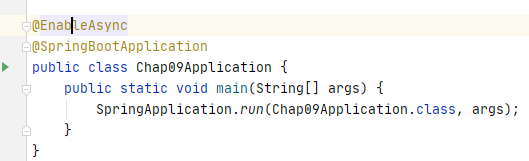
创建service包，在service包中编写业务实现类MyAsyncService类



@Service  
public class MyAsyncService {  
 @Async  
 public void sendSMS() throws Exception{  
 System.*out*.println("调用短信验证码业务方法...");  
 long startTime = System.*currentTimeMillis*();  
 Thread.*sleep*(5000);  
 long endTime = System.*currentTimeMillis*();  
 System.*out*.println("短信业务执行完成耗时："+(endTime-startTime));  
 }  
}

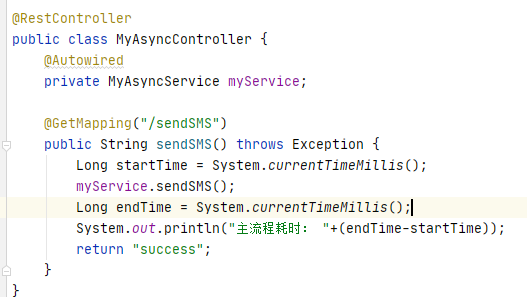
3．开启基于注解的异步任务支持

在项目启动类前添加@EnableAsync注解



4．编写控制层业务调用方法

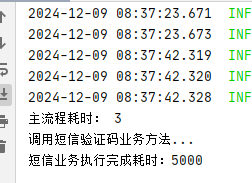
创建controller包，在此包中创建类MyAsyncController。



5．异步任务效果测试

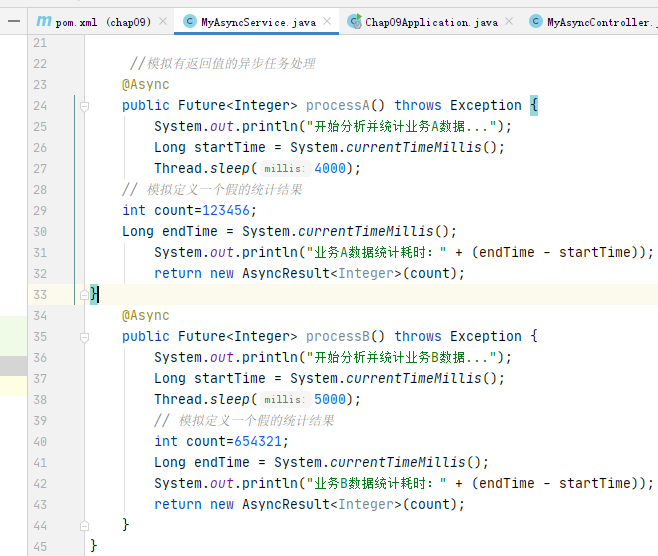
运行项目启动类





二、有返回值异步任务调用

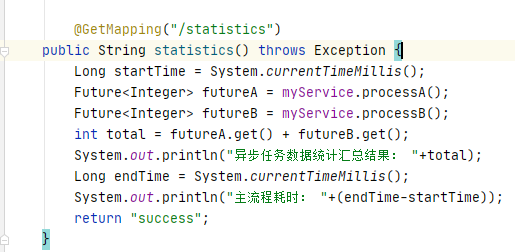
1．在service包中编写异步任务调用方法



*//模拟有返回值的异步任务处理* @Async  
 public Future<Integer> processA() throws Exception {  
 System.*out*.println("开始分析并统计业务A数据...");  
 Long startTime = System.*currentTimeMillis*();  
 Thread.*sleep*(4000);  
 *// 模拟定义一个假的统计结果* int count=123456;  
 Long endTime = System.*currentTimeMillis*();  
 System.*out*.println("业务A数据统计耗时：" + (endTime - startTime));  
 return new AsyncResult<Integer>(count);  
}  
 @Async  
 public Future<Integer> processB() throws Exception {  
 System.*out*.println("开始分析并统计业务B数据...");  
 Long startTime = System.*currentTimeMillis*();  
 Thread.*sleep*(5000);  
 *// 模拟定义一个假的统计结果* int count=654321;  
 Long endTime = System.*currentTimeMillis*();  
 System.*out*.println("业务B数据统计耗时：" + (endTime - startTime));  
 return new AsyncResult<Integer>(count);  
 }

2．编写控制层业务调用方法

在MyAyncController类中添加如下方法



3．测试

启动项目启动类

