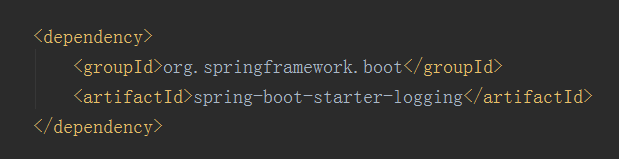
# Spring Boot 整合日志框架

SpringBoot 的版本： 2.2.2.RELEASE

Slf4j 是一种门面模式的应用，为不同的日志框架提供了相关接口的整合。

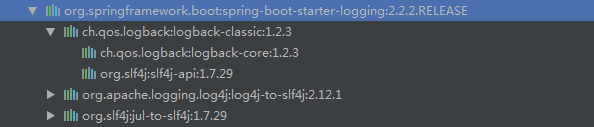
因此Spring Boot 整合框架的时候，整合了slf4j,所以日志记录方法的调用，不应该来直接调用日志的实现类，而是调用日志抽象层里面的方法;

SpringBoot 整合日志框架，直接引入相关的starter 即可。



只需要引入 spring-boot-starter-logging 对应的starter 就可以引入slf4j 的日志框架了。

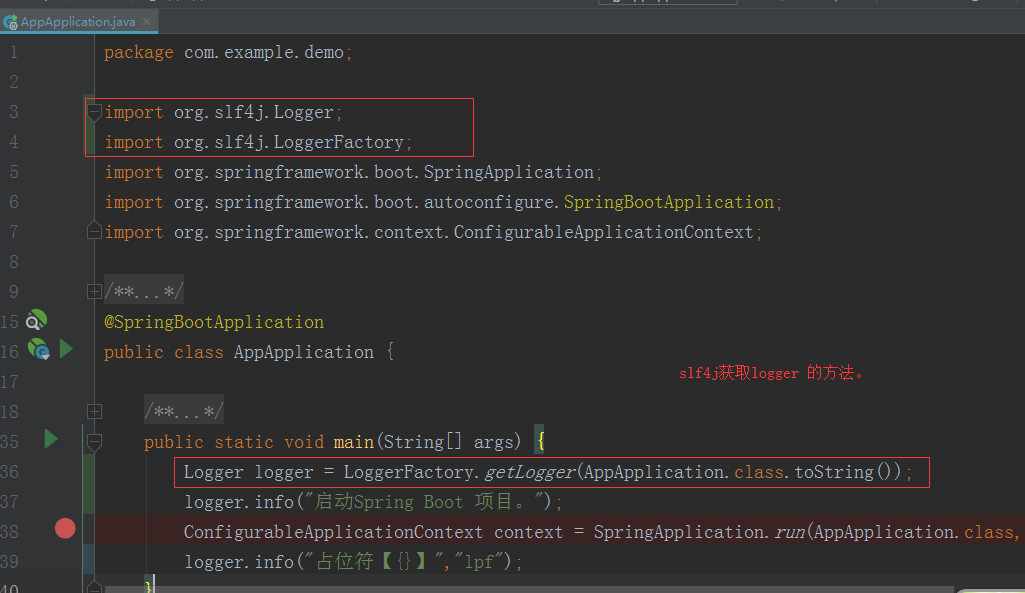
Spring-boot-starter-logging 的maven 依赖如下：



从这里可以看到这个这个starter 会将 jul和log4j 转换为slf4j的接口。 同时， slf4j的默认底层依赖的日志框架是 logback 这个日志框架。

从这里的依赖可以看到springboot 引入slf4j的时候，还引入了log4j 相关的内容，因为我们不需要使用log4j 的相关代码，因此，这里可以将log4j 相关的依赖从starter 中剔除。

添加maven 依赖后，在项目启动类中，通过logger 打印info 级别的日志，在控制台没有找到对应的信息。

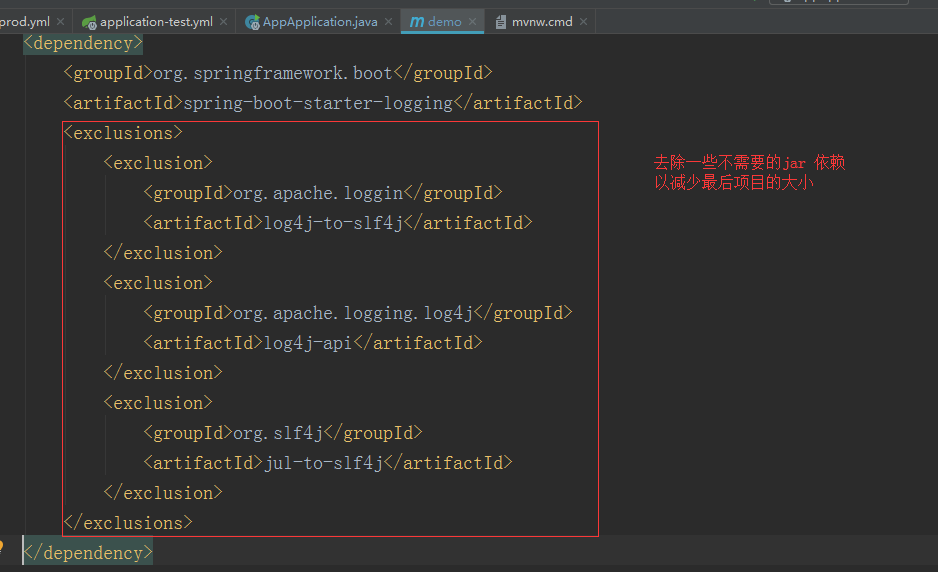


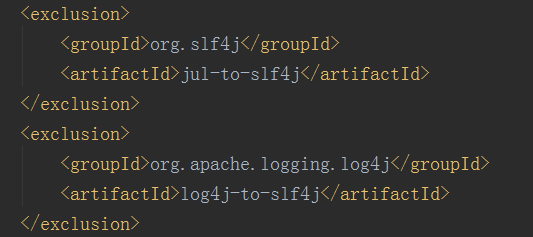
再次启动Spring Boot 项目， 就可以看到通过日志框架打印的日志。这里仅仅是引入了一个starter 依赖，就可以使用日志框架进行日志记录。 是因为 Spring Boot 的自动装配功能。

# 依赖优化：

Jar包或war包瘦身

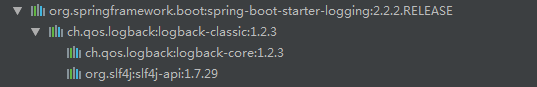
Spring Boot 因为采用 slf4j 框架，所有可以将其他的日志框架相关的jar 删除。





这里要通过exclusion 标签将项目中，不使用的日志框架删除。

处理后maven 依赖如下：



# Sprinboot 控制打印debug级别的日志

在sprinboot 的配置文件中，通过logging.level 来为特定的logger来指定日志级别。

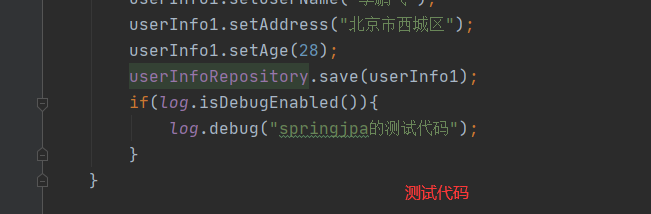
如下

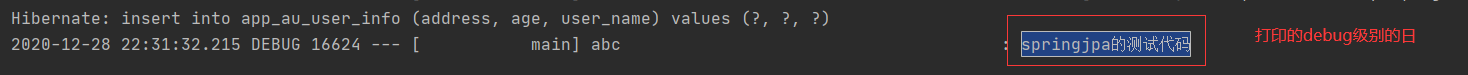


这里的abc 是自己定义的logger 的名称。

private static final Logger *log* = LoggerFactory.*getLogger*("abc");

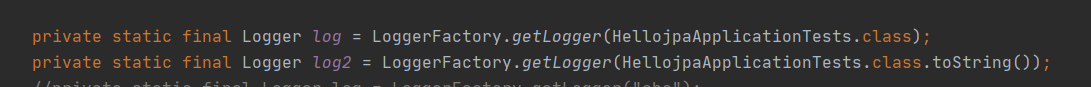
可以通过log.getName() 查看相关logger的名称。



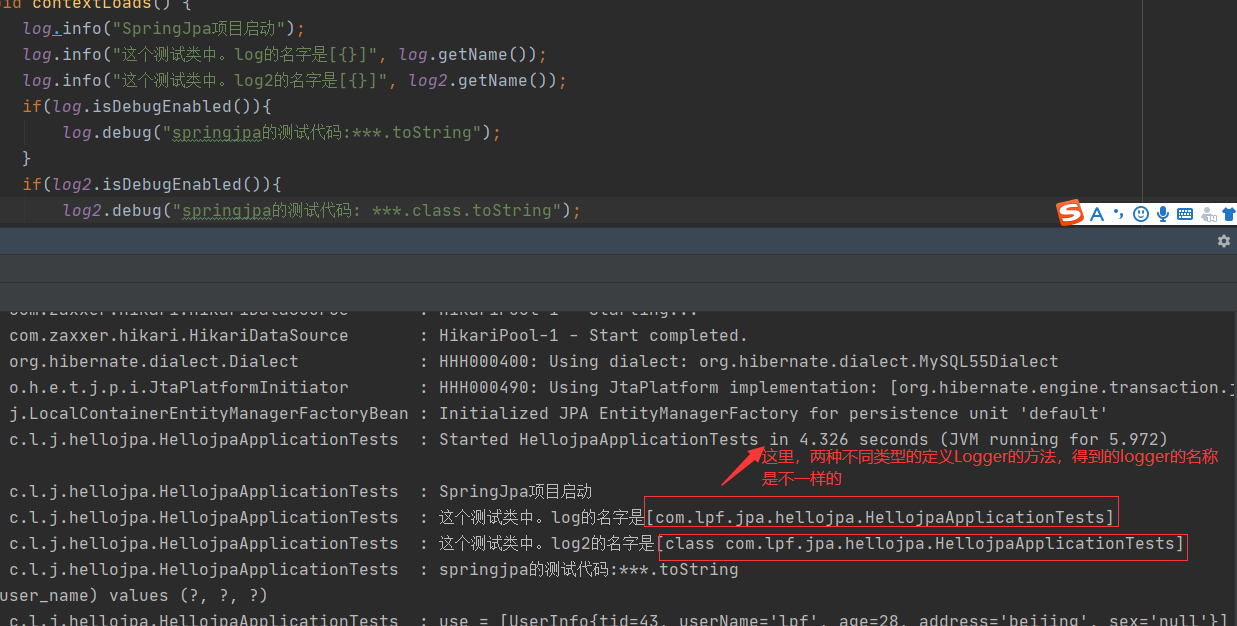


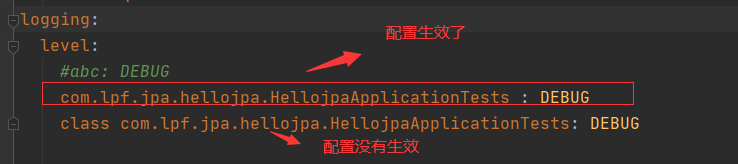
注意：

在定义Loggger的时候，有两种不同的方式，得到的logger的名字是不一样的



在打印的结果中，log2 的名字里还有一个class 的前缀， 自己在配置日志级别的时候，没有生效，是没有class的logger的日志级别生效了。



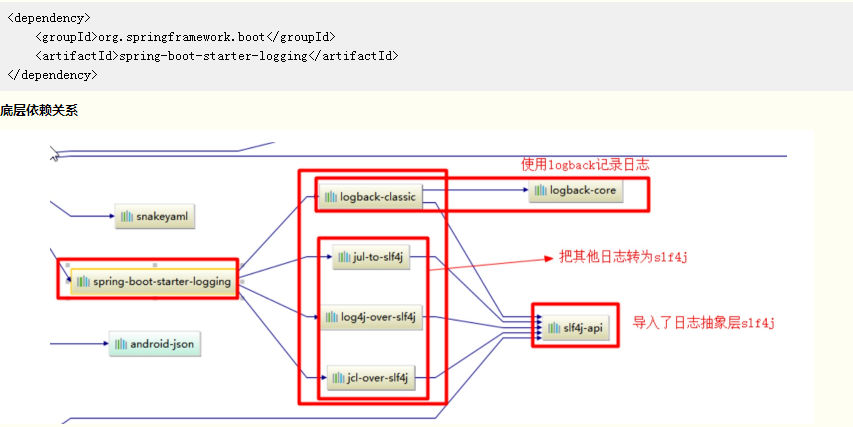


# SpringBoot中 如何 统一日志记录

即使是别的框架和我一起统一使用slf4j进行输出

要将原来的日志框架**替换成**与其相对应的**sf4j日志转换包**就可以了

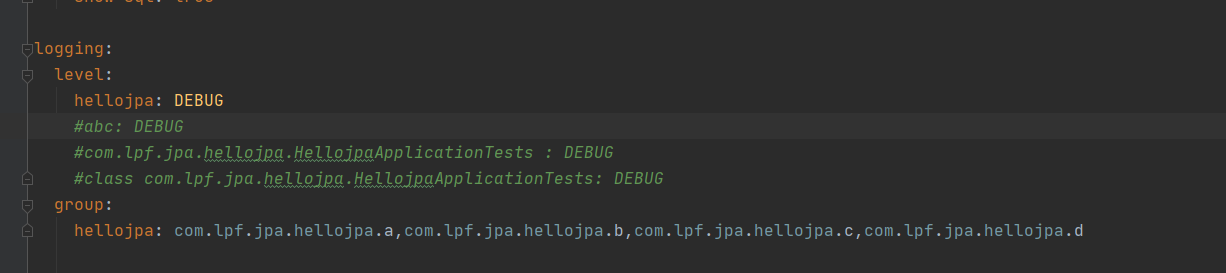
# SpringBoot 的jar 包依赖关系



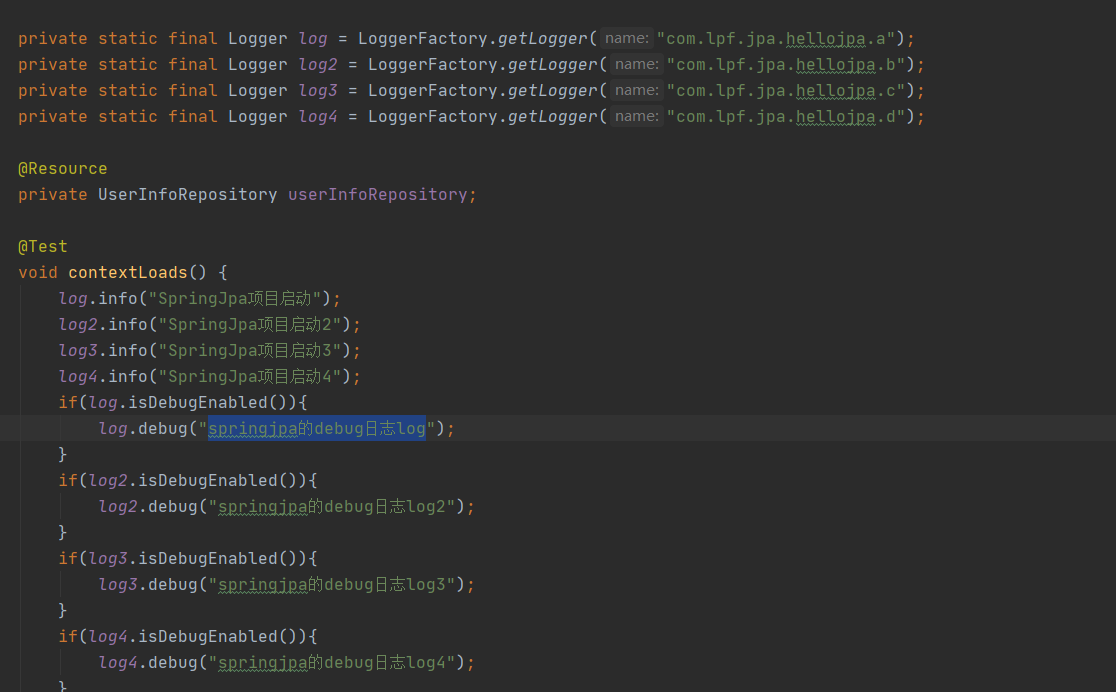
# Spring Boot 中logging group 的使用

在springboot中，使用log group 的功能，将多个logger 合并为一个log group ，可以在配置文件中，同时为该log 组的多个logger 设置日志级别

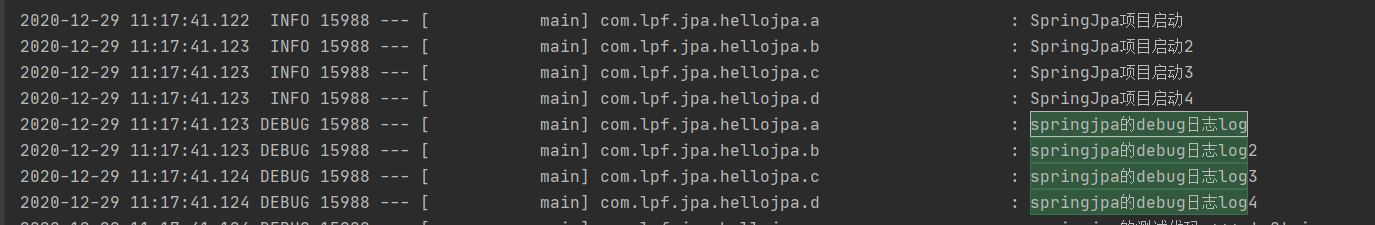
配置内容如下：



测试代码如下：



结果是：



这里在application.yml 文件中，同时配置多个多个logger为一个log group，并且同时设置日志级别为debug。



# TODO

1. 加下来要学习springboot中日志的各种配置。以及如何打印springmvc，spring，mybatis 等框架的信息(debug级别)。
2. 如何为 logger名称为 [class \*\*\*.\*\*\*] 的logger 指定日志级别

# 通过源码解惑

## SpringBoot 是如何和日志框架 slf4j 框架进行整合的

TODO