目录

[开发环境准备 2](#_Toc524432205)

[功能扩展 2](#_Toc524432206)

[程序运行 5](#_Toc524432207)

[数据源设置 5](#_Toc524432208)

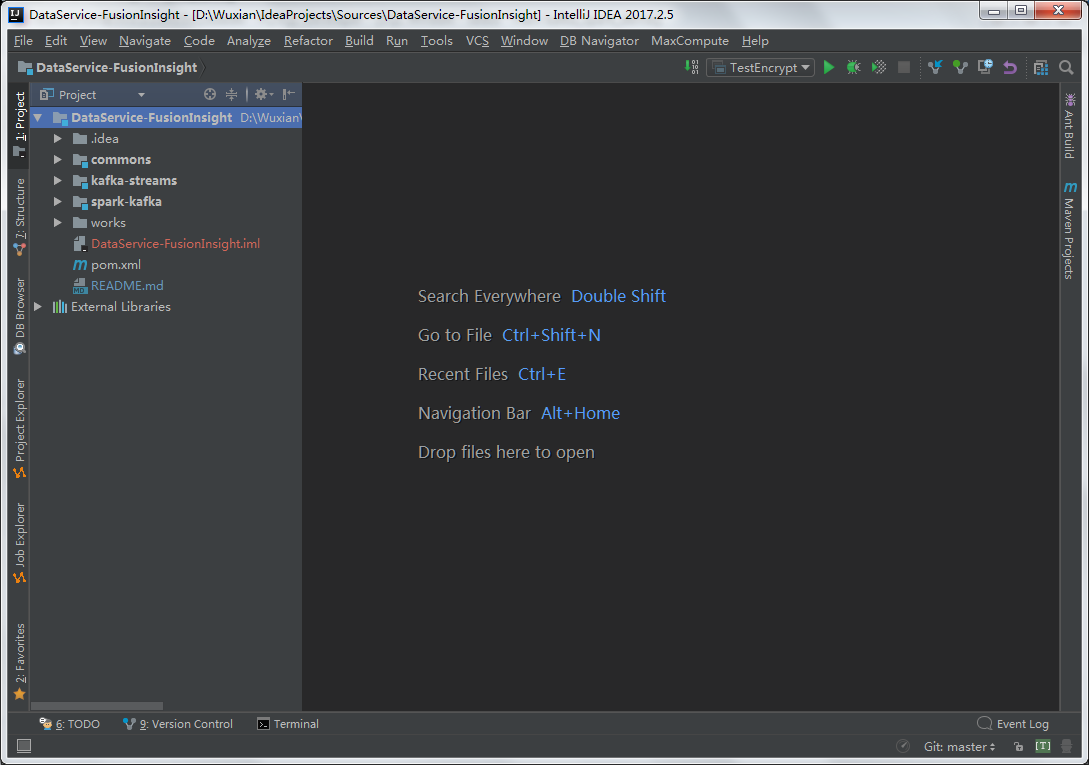
[本地运行 6](#_Toc524432209)

[打包部署 8](#_Toc524432210)

# 开发环境准备

这里以Intellij IDEA为例。

打开IDEA，File -> Import Project…，选择项目目录DataService-FusionInsight下的pom.xml，打开项目。



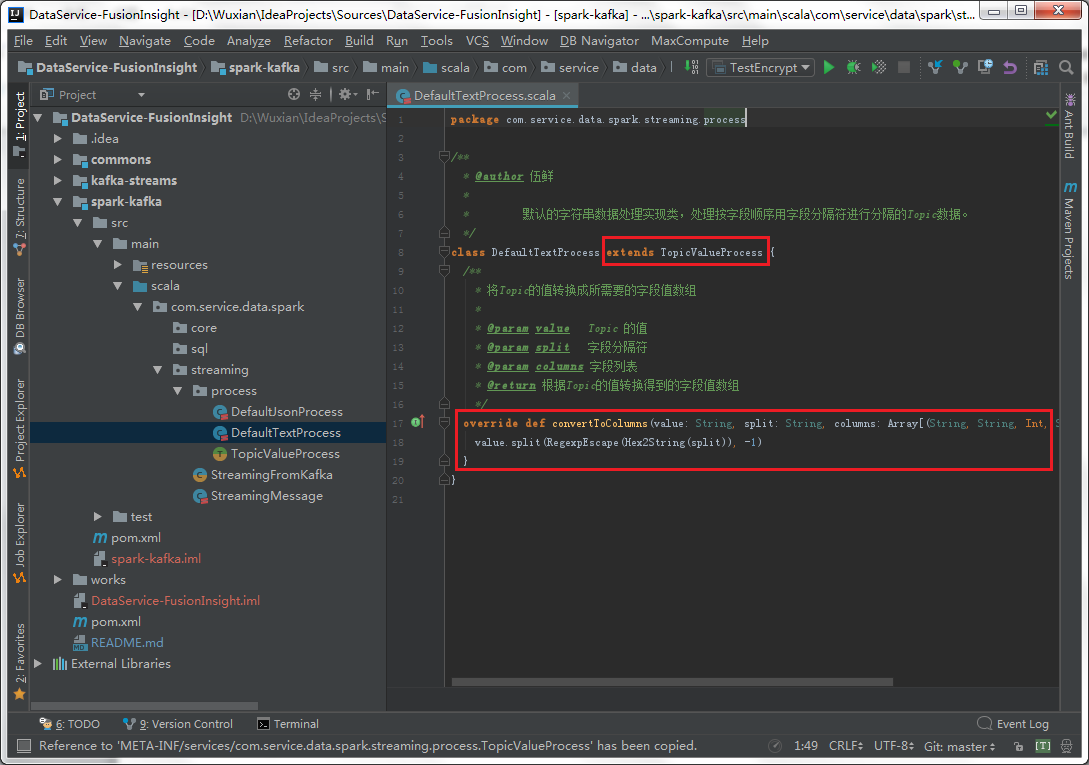
打开了项目之后，可以根据自己的喜好，采用Java或Scala语言进行功能开发、扩展。

# 功能扩展

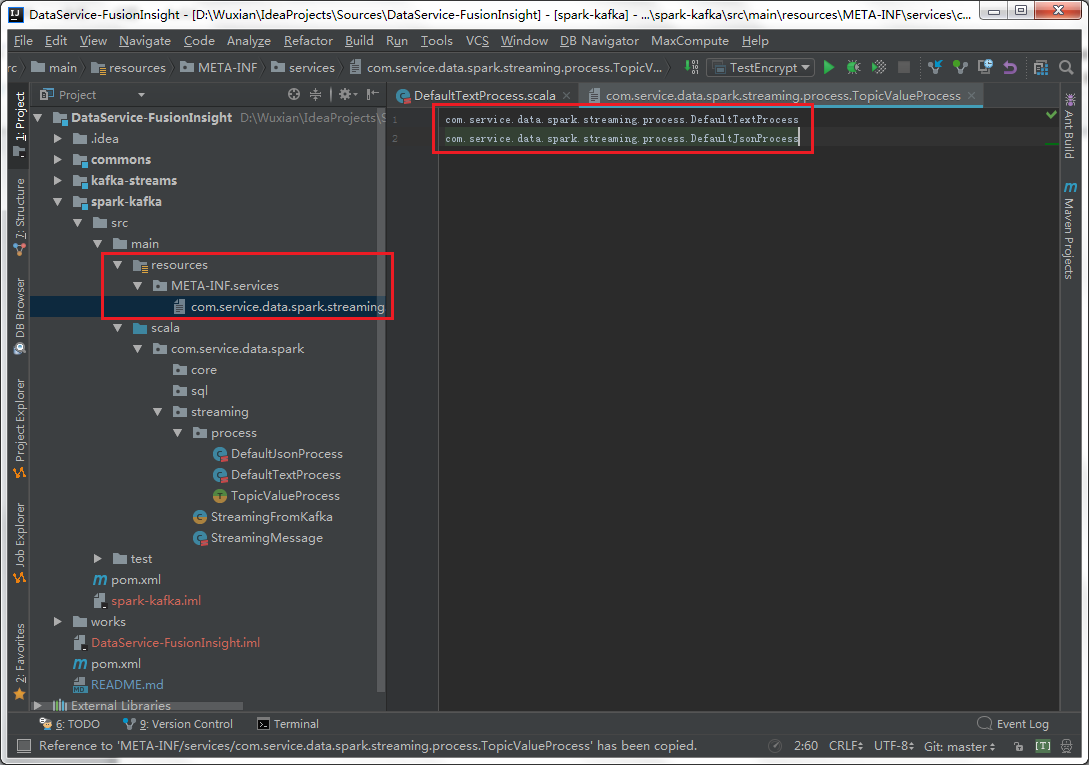
可以根据自己的实际需求对功能进行扩展，这里以扩展日志格式解析功能为例。

目前默认仅实现了分隔符分隔的数据格式和JSON格式的数据格式，如果采集到的日志信息采用了其他格式，那么就需要到解析类进行扩展。可以基于现有的模块进行类的添加，或者新建模块进行开发。如果基于现有模块进行类的添加，则需要在META-INF/services/com.service.data.spark.streaming.process.TopicValueProcess文件中添加自定义的类的名称；如果是新建模块进行开发，则需要在新建的模块中创建META-INF/services/com.service.data.spark.streaming.process.TopicValueProcess文件，并且在该文件中添加自定义的类的名称。

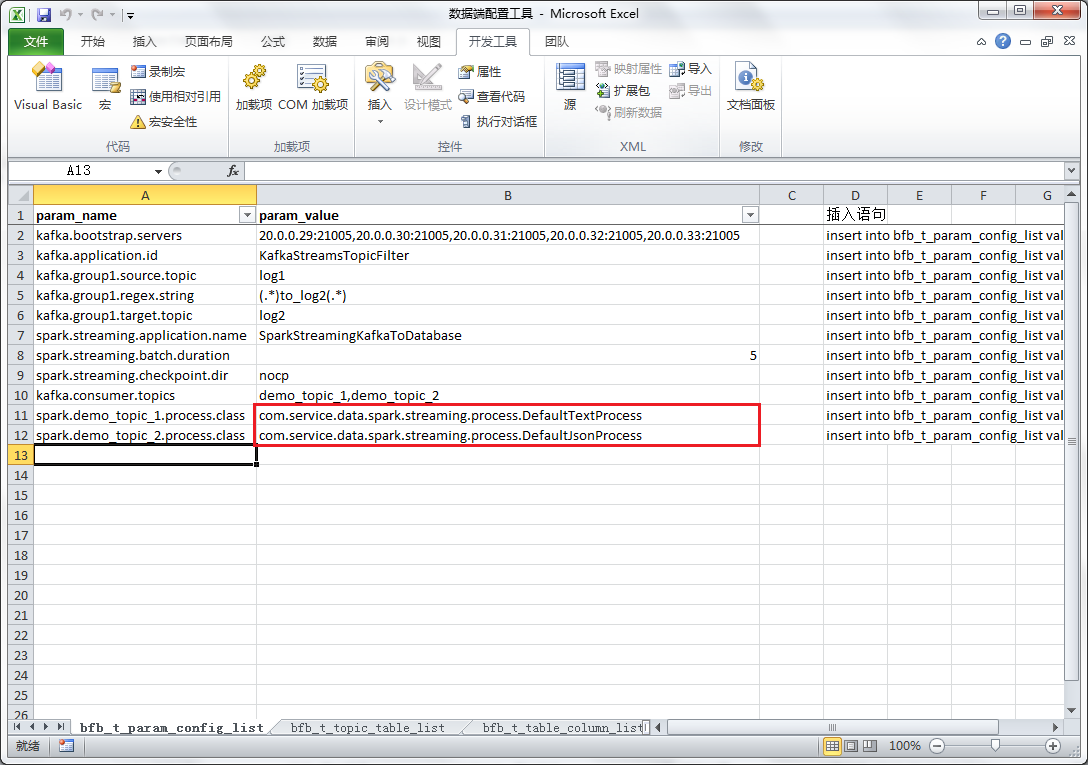
自定义的类需要继承TopicValueProcess，并实现convertToColumns方法，该方法仅实现一个功能：从给定的一行数据中解析出与表结构字段顺序匹配的字段值组成一个数组结构返回。



自定义类编写完成后，需要在指定文件中添加实现类的名称。



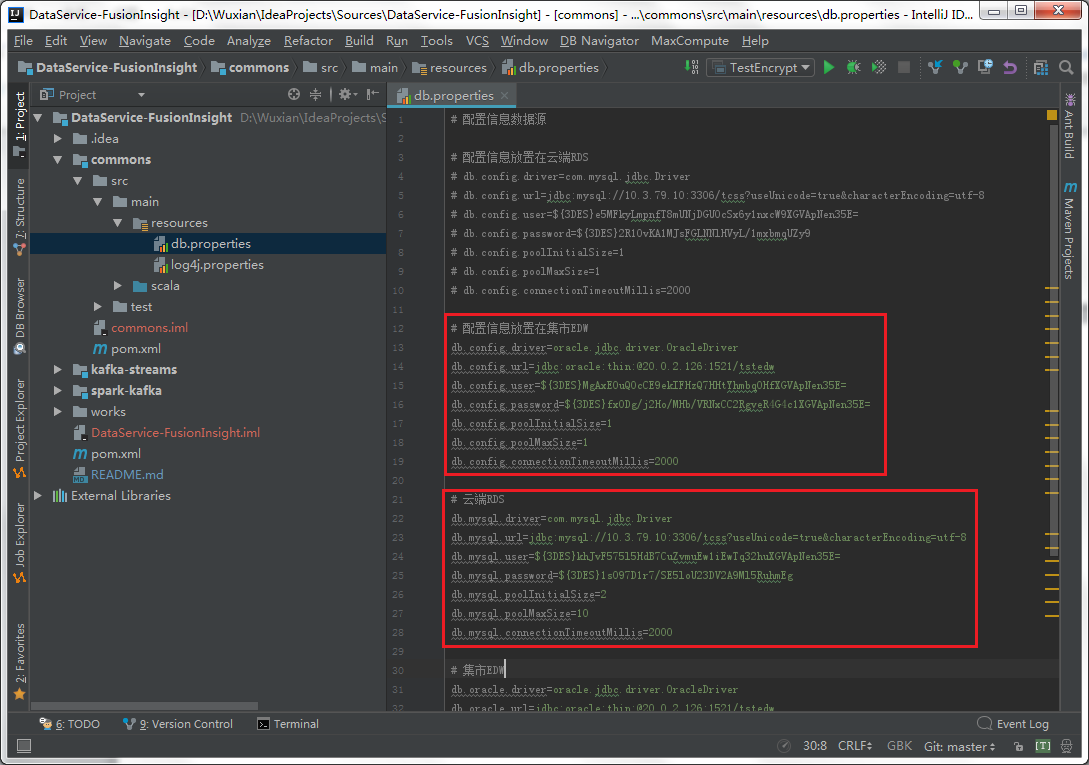
这个自定义类名，用于在配置表中配置，最终实现用该类解析对应的Topic的数据。



# 程序运行

## 数据源设置

在运行程序之前，需要设置配置信息所在的数据库的数据源，以及SparkStreaming程序处理后的数据的最终落地的数据库数据源信息。主要修改的是commons下的db.properties文件，修改里面的数据库驱动、连接、用户、密码等信息。

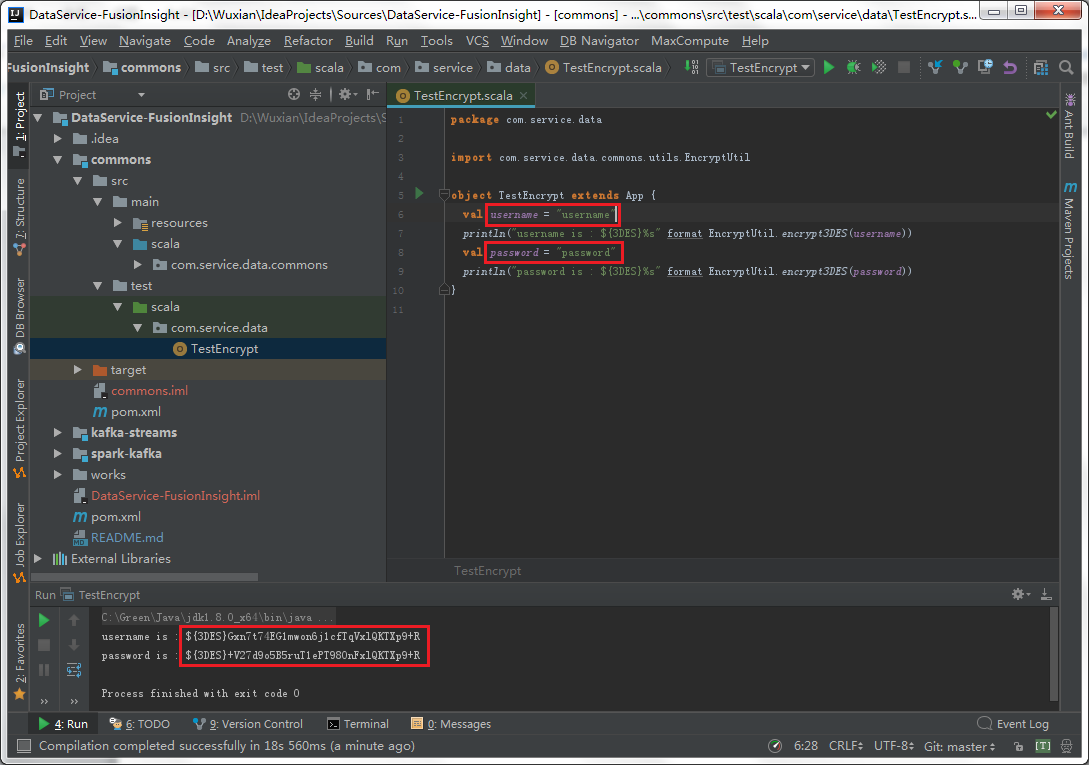


其中：

db.config.\* 配置的是配置信息所在的数据库的数据源，即InitData.sql中的表所在的数据库信息。

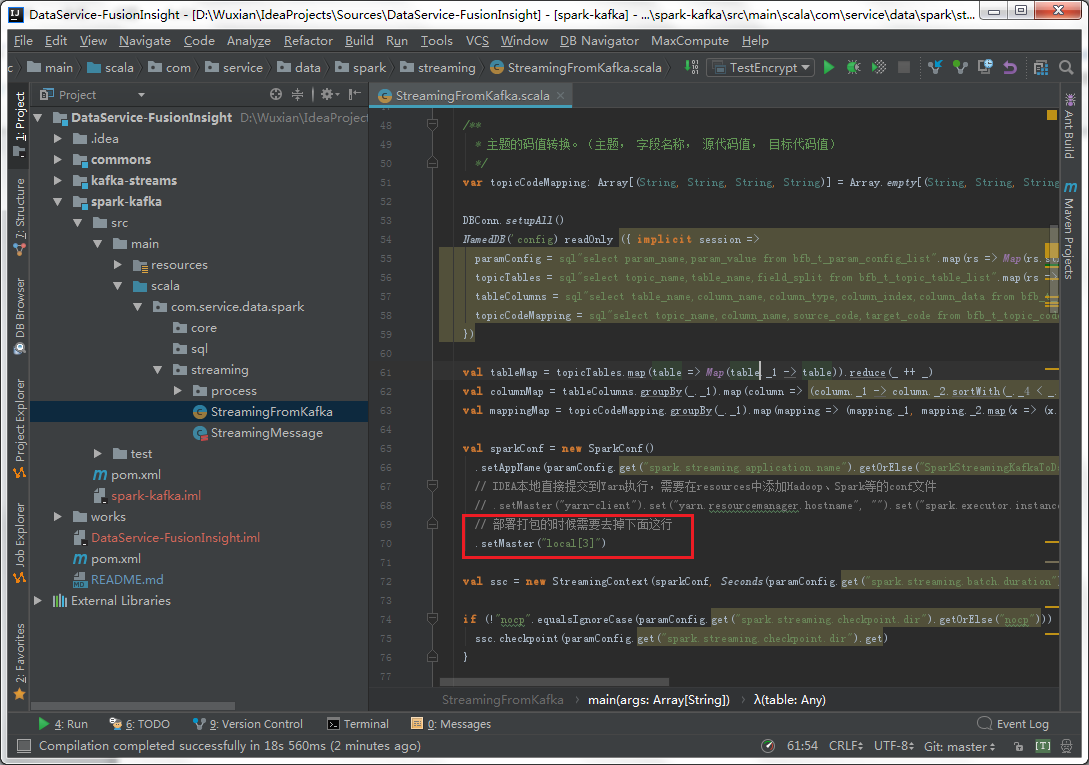
db.mysql.\*/db.oracle.\*/db.other.\* 根据实际需求配置多数据源，即SparkStreaming程序处理数据后需要落地到哪些数据库。

数据源配置信息中，用户名、密码是需要加密处理的，使用commons的test下的TestEncrypt进行用户密码的加密，并将加密后的字符串复制填充到配置文件里即可。

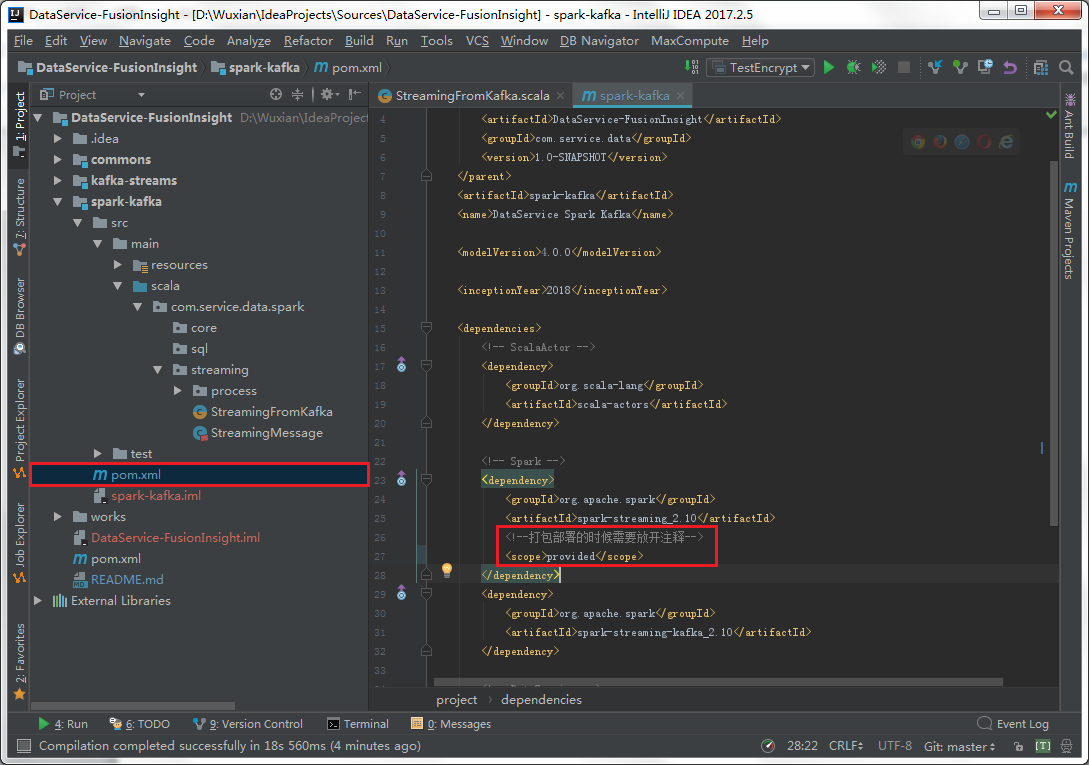


## 本地运行

SparkStreaming程序要在本地运行，需要指定master，可以设置为local[2]。



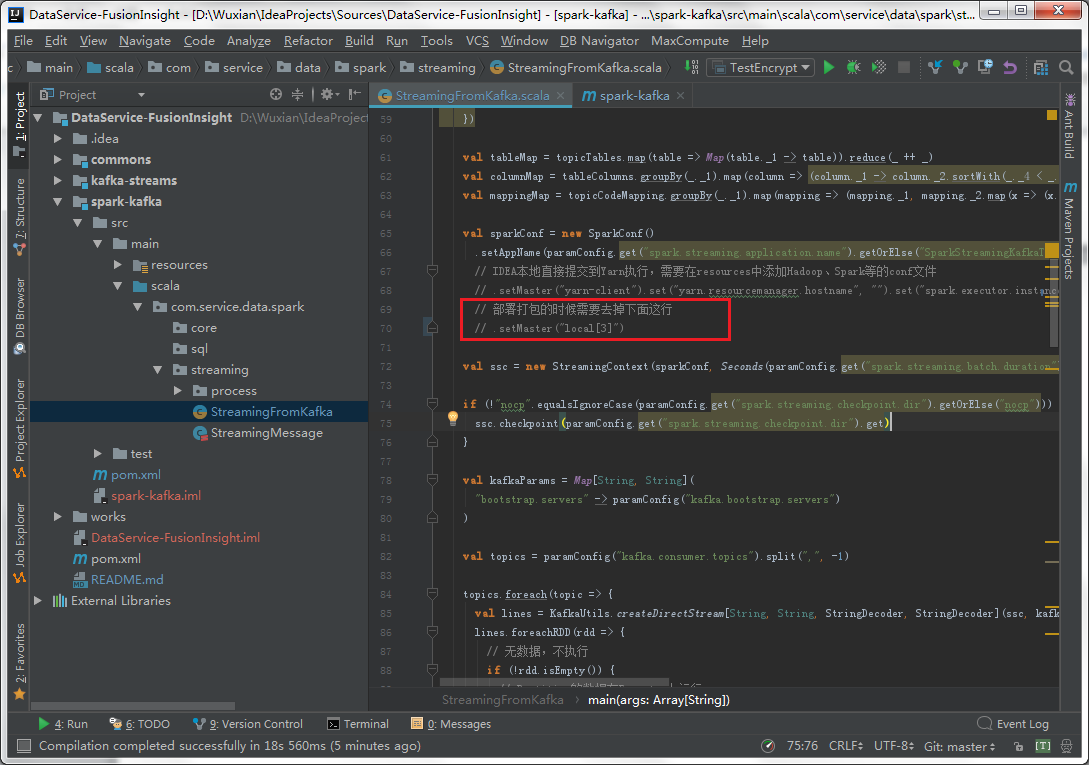
并且需要将pom.xml中的依赖的provided注释掉，以保证依赖中有Spark相关的库。



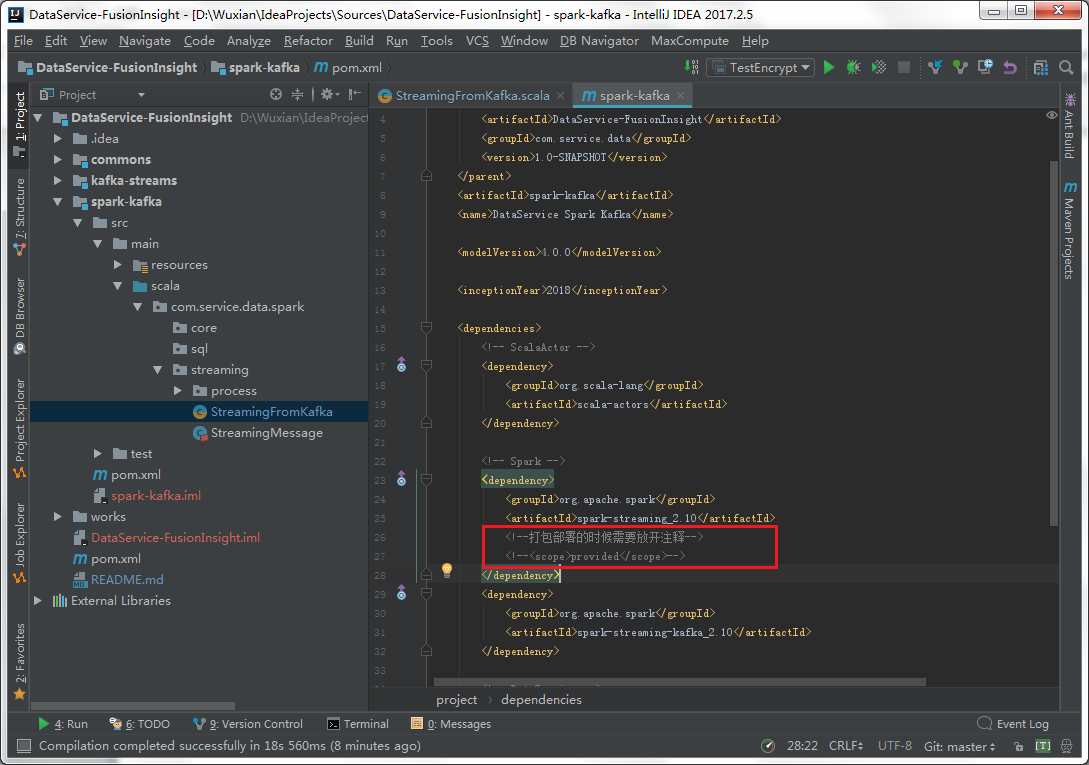
然后直接运行StreamingFromKafka类即可。

## 打包部署

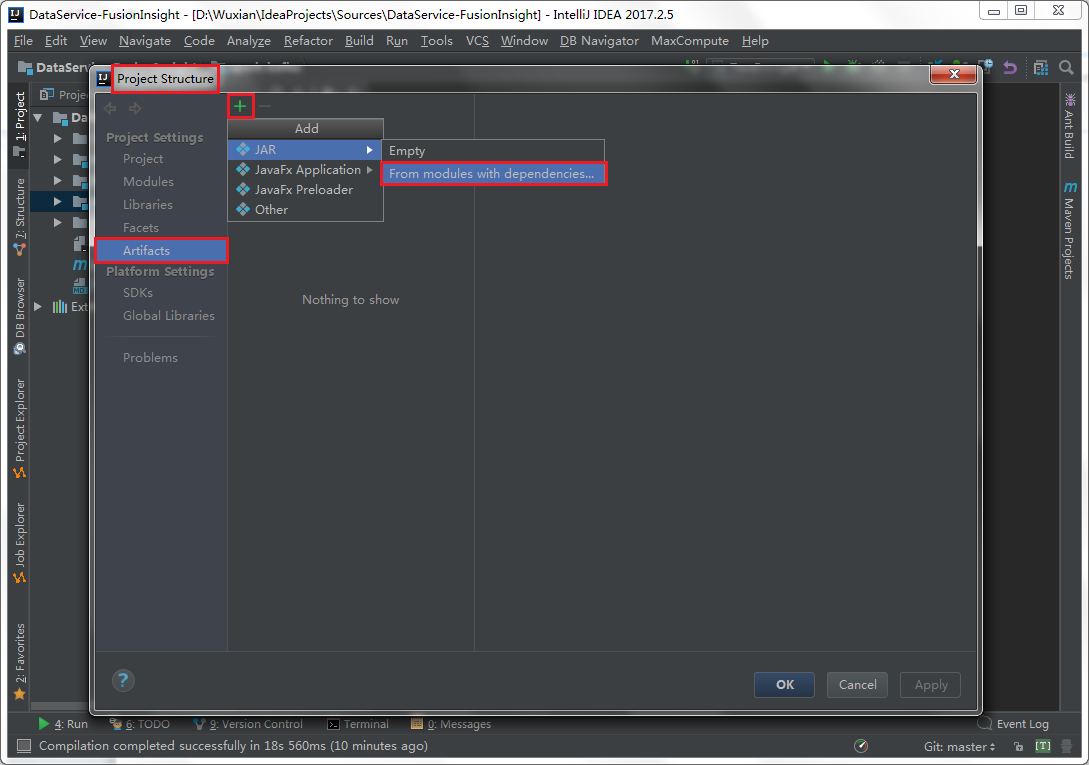
最终的SparkStreaming程序将运行在Yarn上，因此需要将master设置去掉。



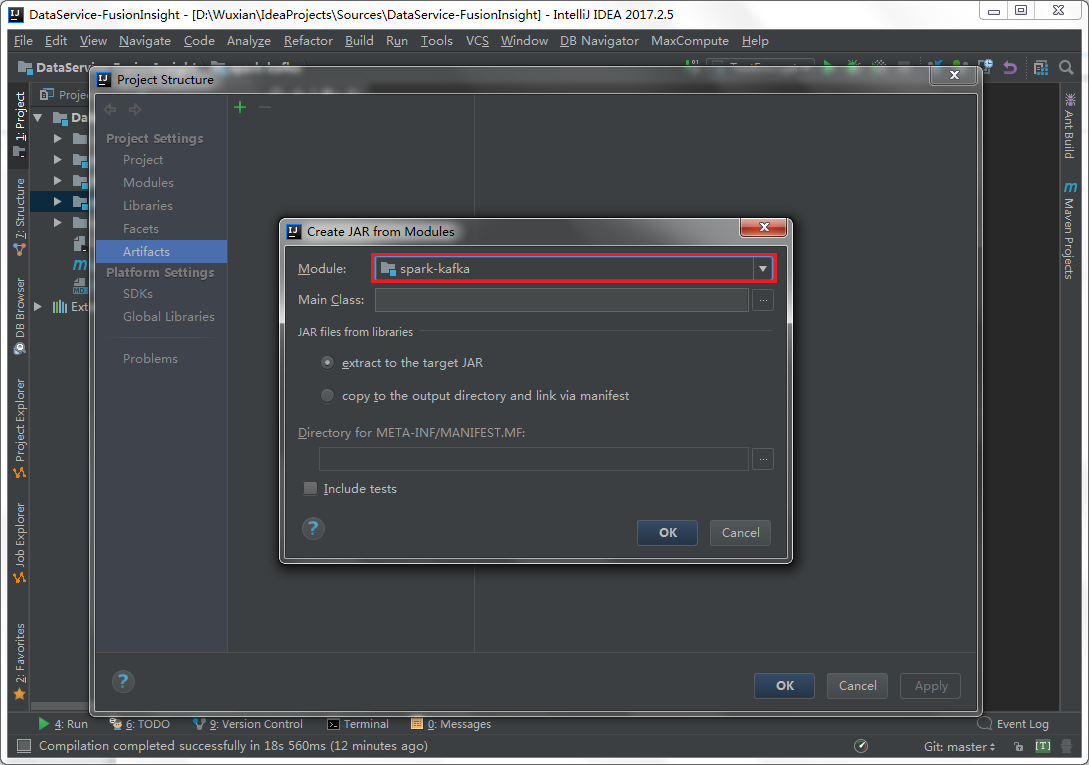
并且为了减小Jar包的大小，在打包的时候可以排除掉集群环境中已有的Jar的依赖。



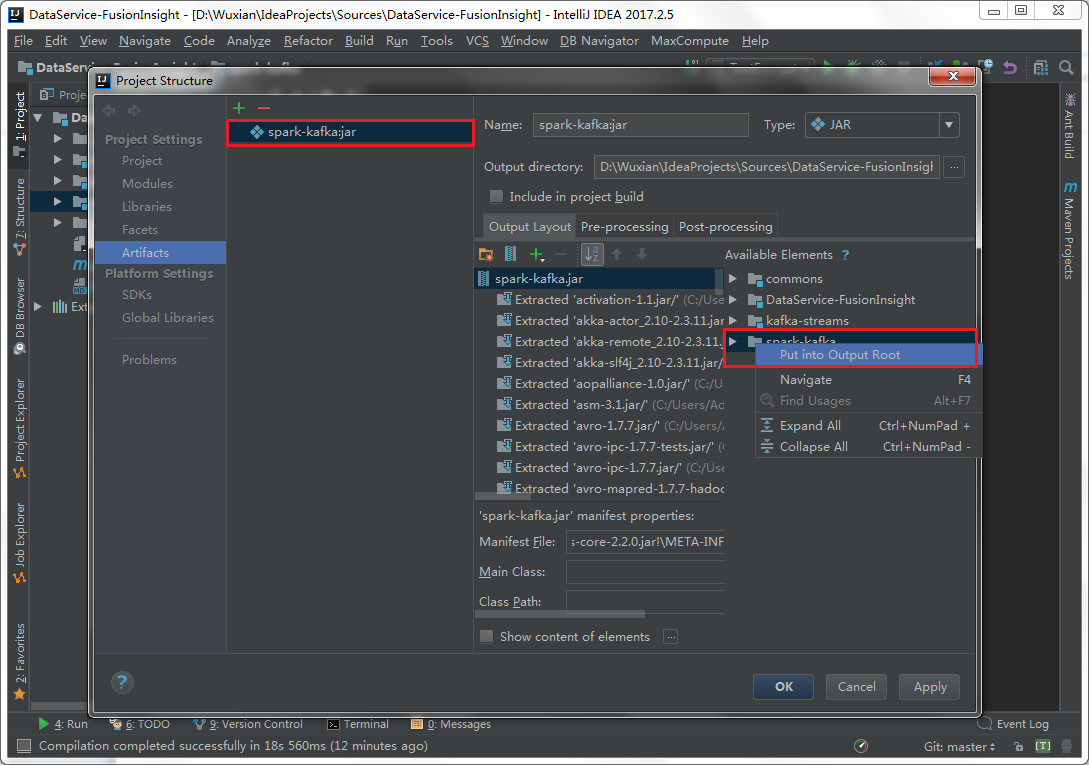
打开Project Structure，在Artifacts中选择“+”，并选择从已有模块添加。



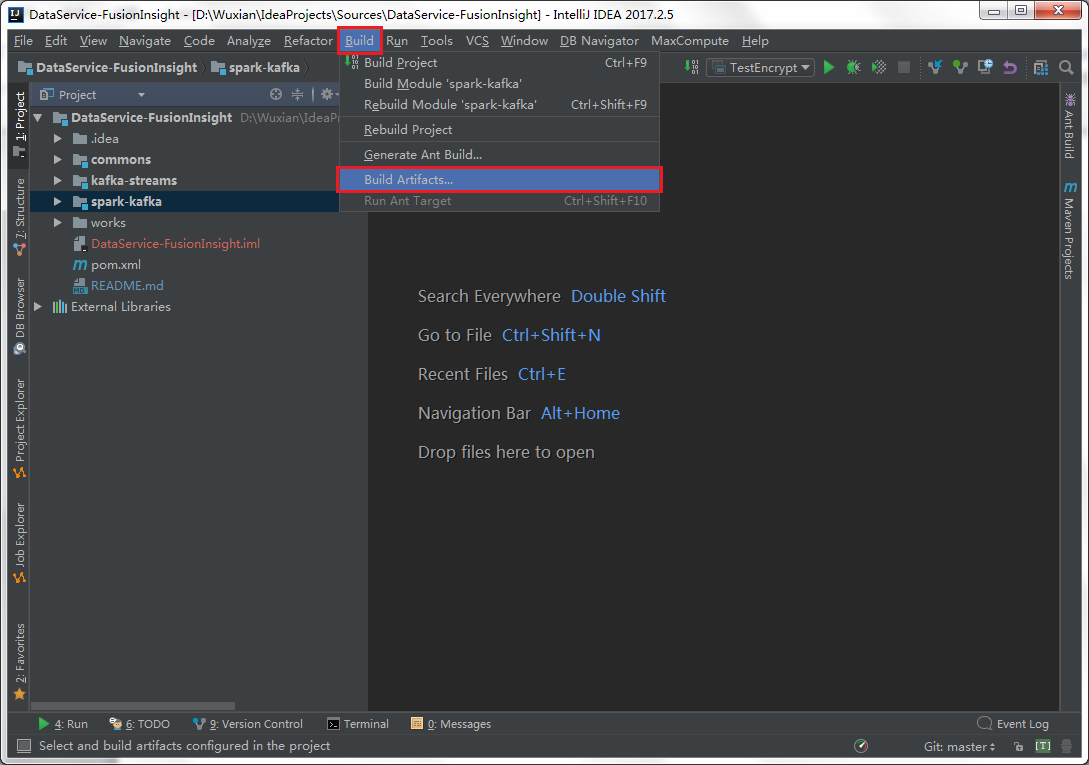
选择spark-kafka模块。



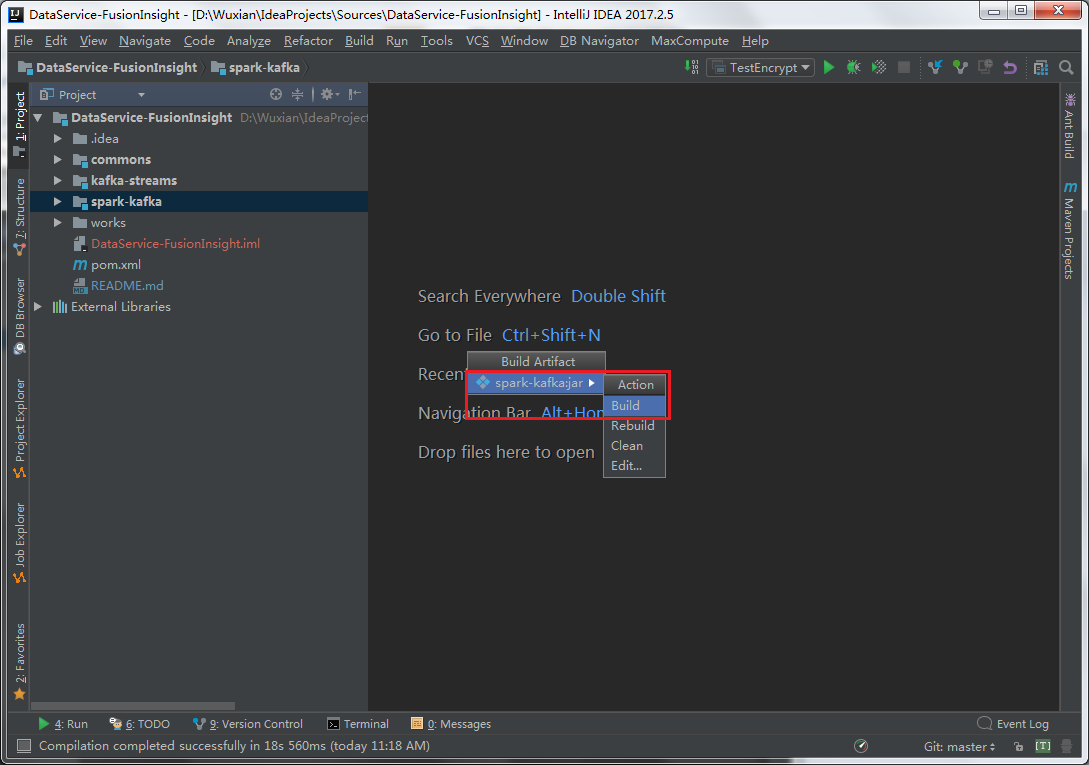
在配置中将spark-kafka设置为Put into Output Root。



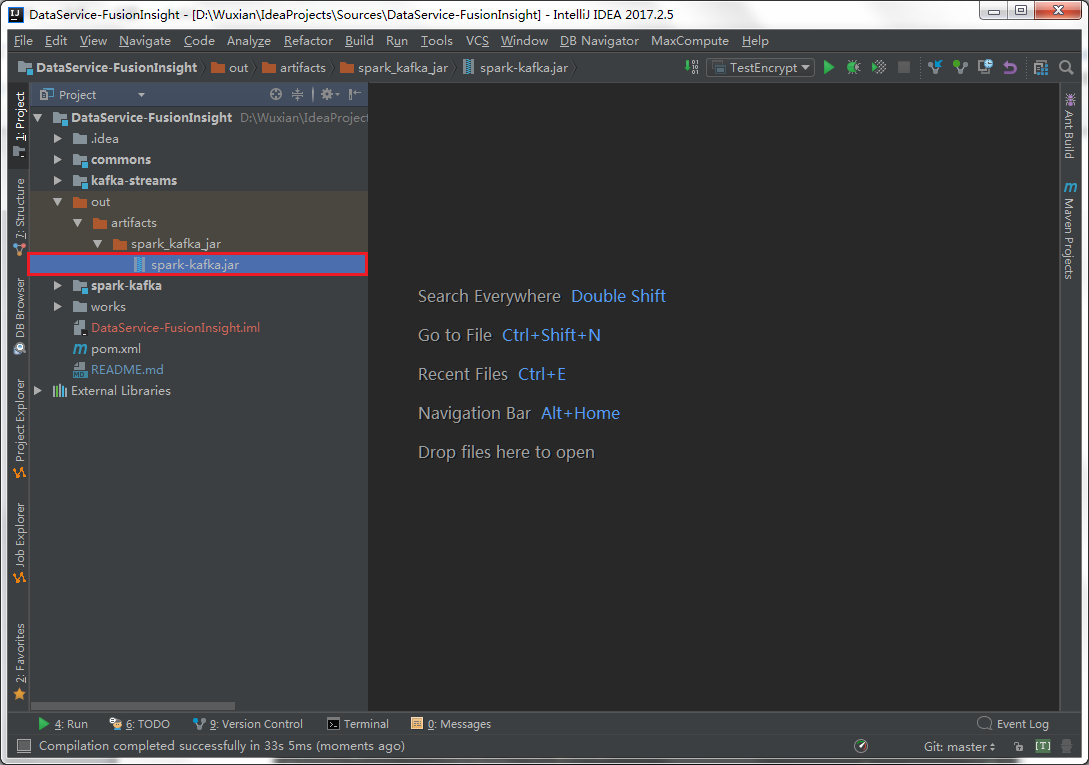
通过Build –> Build Artifacts…



选择Build进行编译打包。



最后，在out目录下有一个spark-kafka.jar的软件包。



这个包就是打包后的最终结果，可以直接提交到Yarn上运行。