1. Spark简介

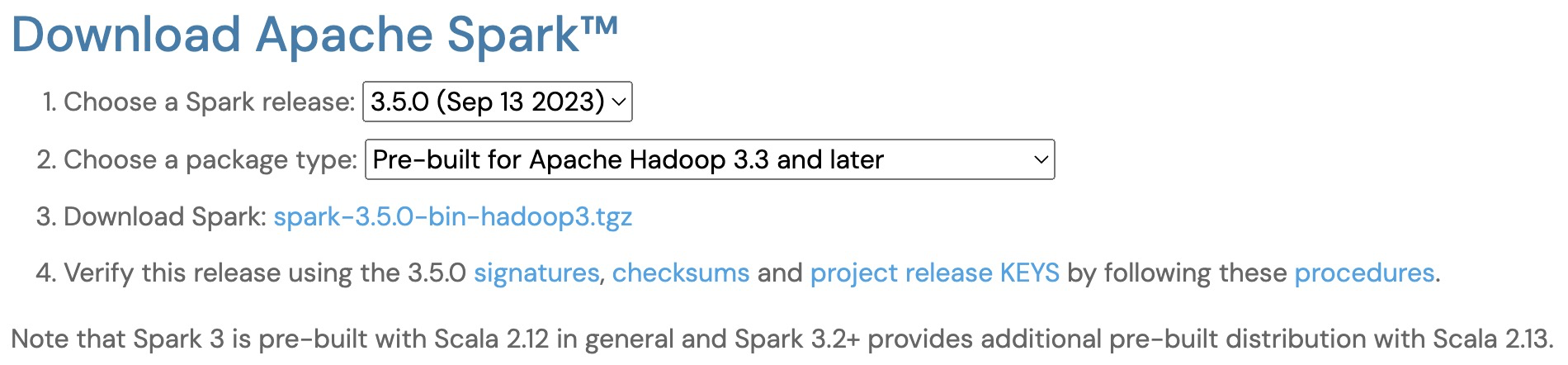
Spark是一个开源的大数据处理框架，旨在处理大规模数据集并进行分布式计算。提供了高效的数据处理能力和灵活的编程模型，可用于处理各种数据处理任务，包括数据清洗、数据转换、机器学习、实时流处理和图形处理等。

Spark的核心是RDD（Resilient Distributed Dataset），即弹性分布式数据集，属于一种分布式的内存系统的数据集应用。其主要优势就是RDD本身的特性，即能与其他系统兼容，可以导入外部储存系统的数据集，包括Hadoop Distributed File System（HDFS）、Apache Cassandra、Apache HBase、Amazon S3等。有两种对RDD进行操作的方法：

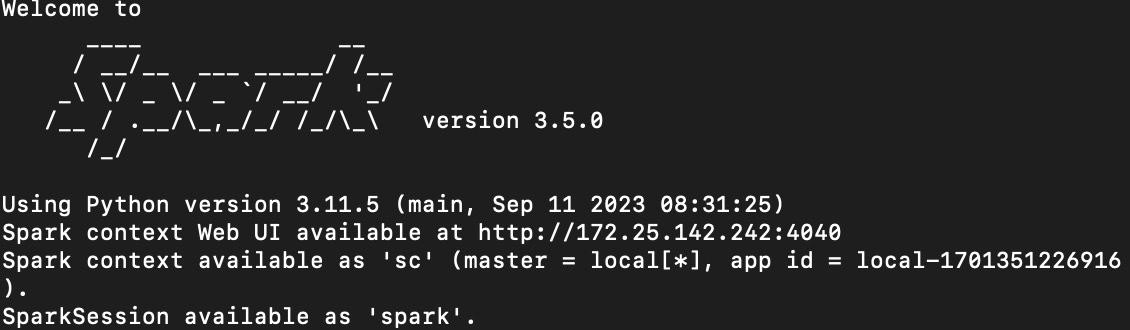
* Transformation：转换，用于创建新的数据集。
* Action：操作，指示Spark执行计算并将结果发送回驱动程序。

1. Spark安装

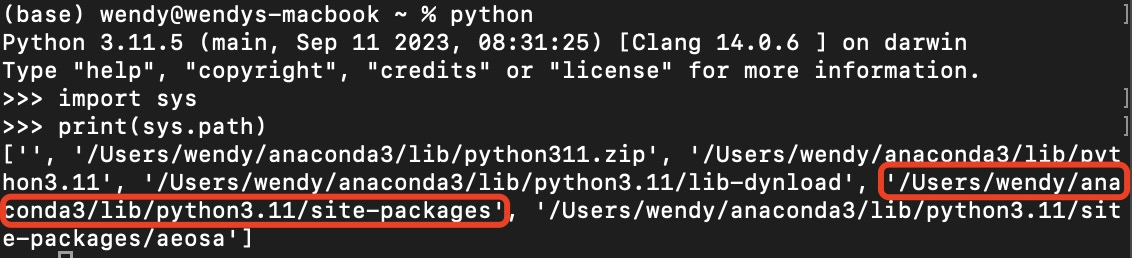
官网：<https://spark.apache.org/downloads.html>



解压即可。在终端中输入pySpark，如果安装成功会看到下图：

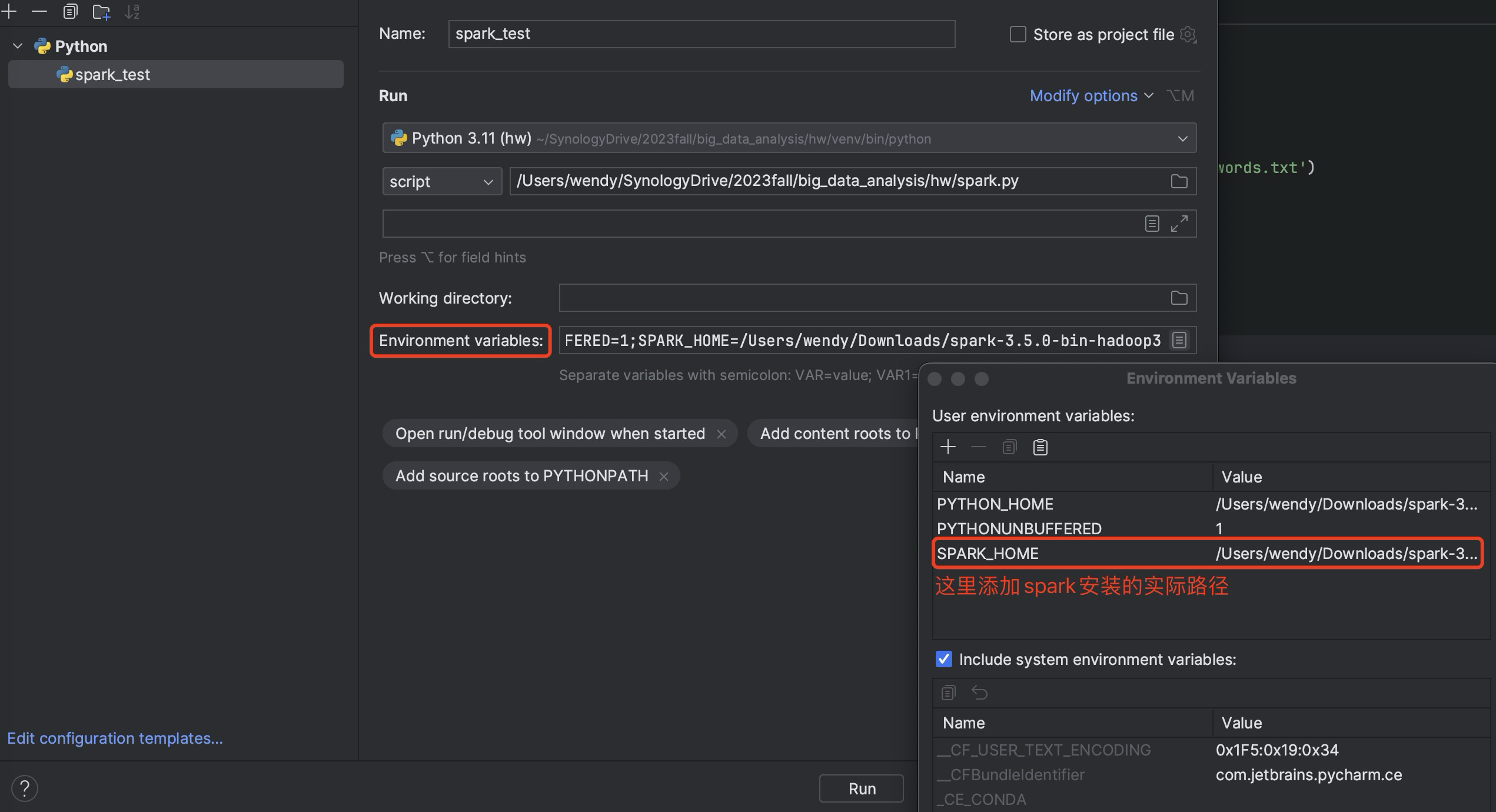


1. 配置在PyCharm中调用pySpark的加载包
2. 在mac上使用pyspark，需要强行安装第三方包。将spark文件夹下的pyspark文件拷贝到python/site-packages文件夹目录下即可。



1. 补充配置

PyCharm->Run->Edit Configurations，添加环境变量。

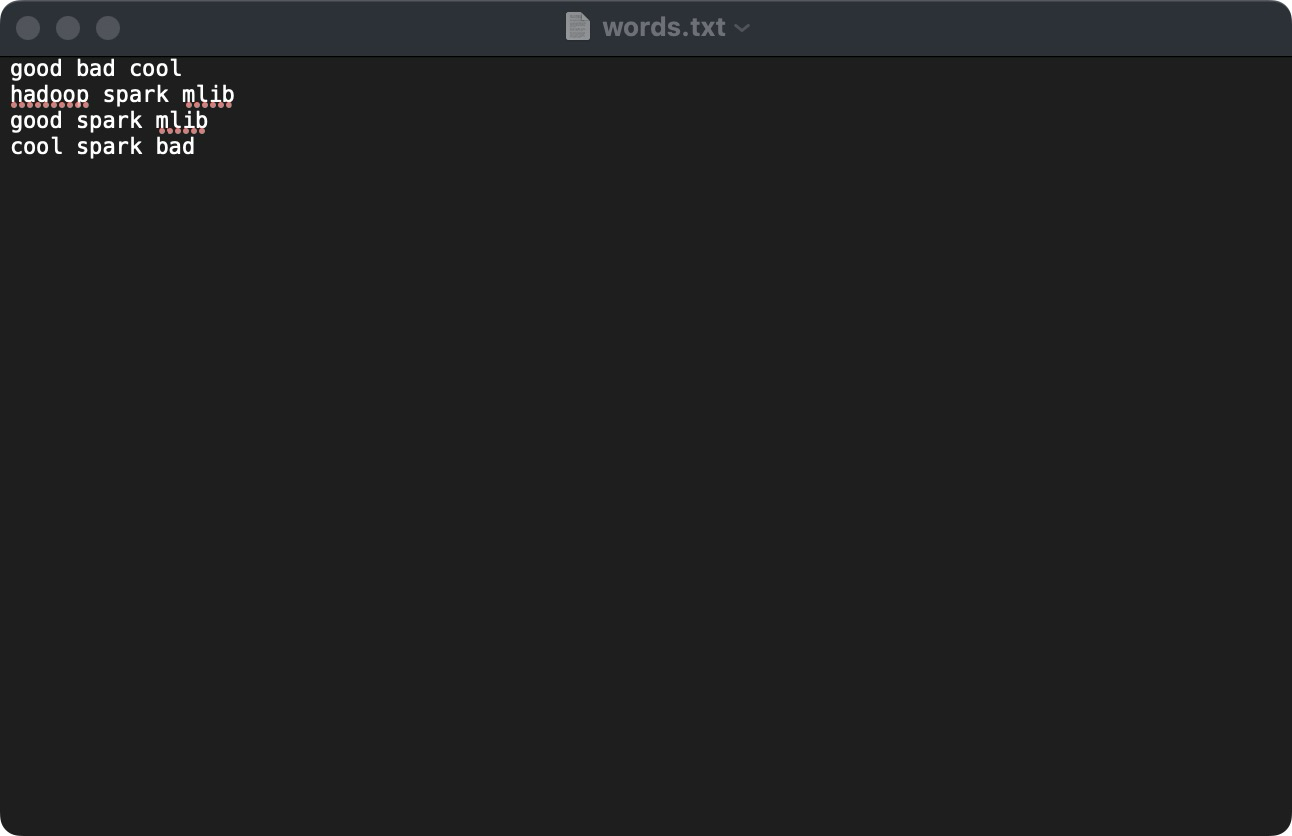


1. 初步运行

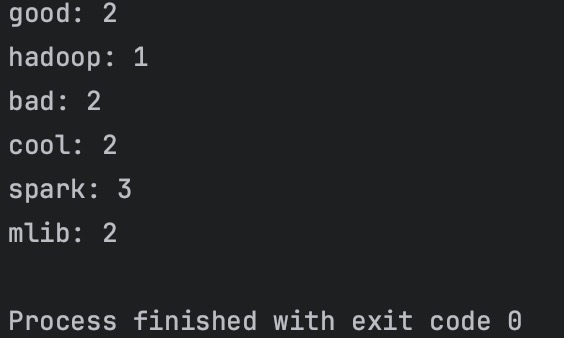
运行这段代码：

from operator import add  
from pyspark import SparkContext  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 sc = SparkContext(appName="PythonWordCount")  
 lines = sc.textFile('words.txt')  
 counts = lines.flatMap(lambda x: x.split(' ')) \  
 .map(lambda x: (x,1)) \  
 .reduceByKey(add)  
 output = counts.collect()  
 for (word, count) in output:  
 print("%s: %i" % (word, count))  
 sc.stop()

其中，words.txt文件长这样：



运行结果如下：



1. pyspark使用语法

使用sparkcontext读取文件：

* data\_rdd = sc.textFiles('.txt’) #读取文件内容，返回RDD
* path\_data\_rdd = sc.wholeTextFile((‘.txt’)) #读取文件内容与path

1. count()返回元素个数

from pyspark import SparkContext  
sc = SparkContext("local", "count app")  
words = sc.parallelize(  
 ["scala",  
 "java",  
 "hadoop",  
 "spark",  
 "akka",  
 "spark vs hadoop",  
 "pyspark",  
 "pyspark and spark"  
 ])  
counts = words.count()  
print("Number of elements in RDD -> %i" % counts)



1. collect()返回所有元素，并转换为python数据类型

from pyspark import SparkContext  
sc = SparkContext("local", "collect app")  
words = sc.parallelize(  
 ["scala",  
 "java",  
 "hadoop",  
 "spark",  
 "akka",  
 "spark vs hadoop",  
 "pyspark",  
 "pyspark and spark"  
 ])  
coll = words.collect()  
print("Elements in RDD -> %s" % coll)



1. filter()返回一个包含元素的新的RDD，满足过滤器内部的功能。

from pyspark import SparkContext  
sc = SparkContext("local", "Filter app")  
words = sc.parallelize(  
 ["scala",  
 "java",  
 "hadoop",  
 "spark",  
 "akka",  
 "spark vs hadoop",  
 "pyspark",  
 "pyspark and spark"]  
 )  
words\_filter = words.filter(lambda x: 'spark' in x)  
filtered = words\_filter.collect()  
print("Filtered RDD -> %s" % (filtered))



1. map(f,preservesPartitioning=False)，通过将该函数应用于RDD中的每个元素来返回新的RDD。

from pyspark import SparkContext  
sc = SparkContext("local", "Map app")  
words = sc.parallelize(  
 ["scala",  
 "java",  
 "hadoop",  
 "spark",  
 "akka",  
 "spark vs hadoop",  
 "pyspark",  
 "pyspark and spark"]  
 )  
words\_map = words.map(lambda x: (x, 1))  
mapping = words\_map.collect()  
print("Key value pair -> %s" % (mapping))

