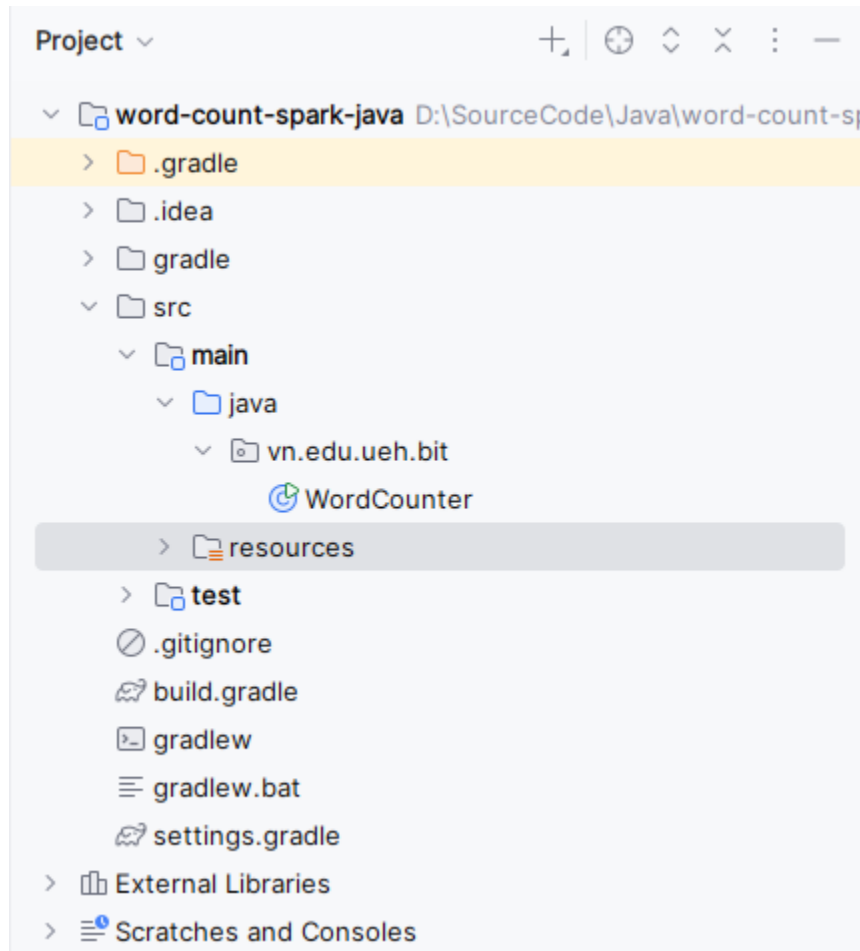


# jvmVí dụ words count với Spark và Java

## Tạo project

Trong IntelliJ IDEA tạo 1 project java với build-tool là gradle. Cấu trúc như sau



## Viết code

Thêm spark core dependency như sau

```
implementation("org.apache.spark:spark-core_2.13:3.5.5")
```

Sourcode được cho như sau

```
package vn.edu.ueh.bit;

import org.apache.spark.SparkConf;
import org.apache.spark.api.java.JavaPairRDD;
import org.apache.spark.api.java.JavaRDD;
import org.apache.spark.api.java.JavaSparkContext;
import scala.Tuple2;
import java.util.Arrays;
```

```

public class WordCounter {
    private static void wordCount(String inputFile, String outputFile) {
        SparkConf conf = new SparkConf()
            .setAppName("Word Counter")
            .setMaster("local") //1 JVM. Use local[*] if you want to use multiple JVMs as much as possible
        ;
        try (JavaSparkContext sc = new JavaSparkContext(conf)) {
            JavaRDD<String> lines = sc.textFile(inputFile);
            JavaRDD<String> words = lines
                .flatMap(line -> Arrays.asList(line.split(" ")))
                .iterator();
            JavaPairRDD<String, Integer> pairs = words
                .mapToPair(word -> new Tuple2<>(word, 1));
            JavaPairRDD<String, Integer> counts = pairs
                .reduceByKey(Integer::sum); //a + b
            counts = counts.sortByKey();
            counts.saveAsTextFile(outputFile); //save results
        }
    }

    public static void main(String[] args) throws Exception {
        // String inputFile = "src/main/resources/input.txt";
        // String outputFolder = "src/main/resources/out";
        String inputFile = "hdfs://localhost:9000/data/input.txt";
        String outputFolder = "hdfs://localhost:9000/output";
        wordCount(inputFile, outputFolder);
    }
}

```

Bạn có thể dung input/output file được lưu trữ trên file cụ thể hoặc bạn có thể dung HDFS để lưu trữ file. Trong trường hợp dung HDFS thì bạn phải start haddop lên (chạy lệnh start-dfs và start-yarn).

## Giải thích

Trong Spark, chúng ta cần thực thi một công việc nào đó thì cần phải có 1 session. Session này cần phải cấu hình với rất nhiều thuộc tính. Tuy nhiên, để đơn giản, chúng ta cung cấp cho session tên của ứng dụng (AppName).

**local** – “local” là giá trị đặc biệt được sử dụng cho master tham số khi khởi tạo SparkContext hoặc SparkSession. Khi bạn chỉ định local là master, điều đó có nghĩa là Spark sẽ chạy ở chế độ local, chỉ sử dụng một JVM (Java Virtual Machine) duy nhất trên máy cục bộ nơi tập lệnh Python của bạn được thực thi. Chế độ này chủ yếu được sử dụng cho mục đích phát triển, thử nghiệm và gỡ lỗi.

```

SparkConf conf = new SparkConf()
    .setAppName("Word Counter")
    .setMaster("local");
JavaSparkContext sc = new JavaSparkContext(conf)

```

Spark coi mọi tài nguyên mà nó xử lý là RDD (Resilient Distributed Datasets) giúp nó sắp xếp dữ liệu trong một cấu trúc dữ liệu tìm kiếm hiệu quả hơn nhiều để phân tích. Bây giờ chúng ta sẽ chuyển đổi tệp đầu vào thành một đối tượng JavaRDD:

```

JavaRDD<String> lines = sc.textFile(inputFile);

```

Tiếp tục, chúng ta tách các từ trong file ra

```
JavaRDD<String> words = lines
    .flatMap(line -> Arrays.asList(line.split(" ")))
    .iterator();
```

Chúng ta cần một cặp (pair) theo dạng <từ, số lần xuất hiện>

```
JavaPairRDD<String, Integer> counts = pairs
    .reduceByKey(Integer::sum);
```

Trong đó Integer::sum được hiểu như là reduceByKey((x, y) -> (x + (y)

Để kết quả dễ quan sát, chúng ta cũng có thể sắp xếp lại kết quả bằng câu lệnh

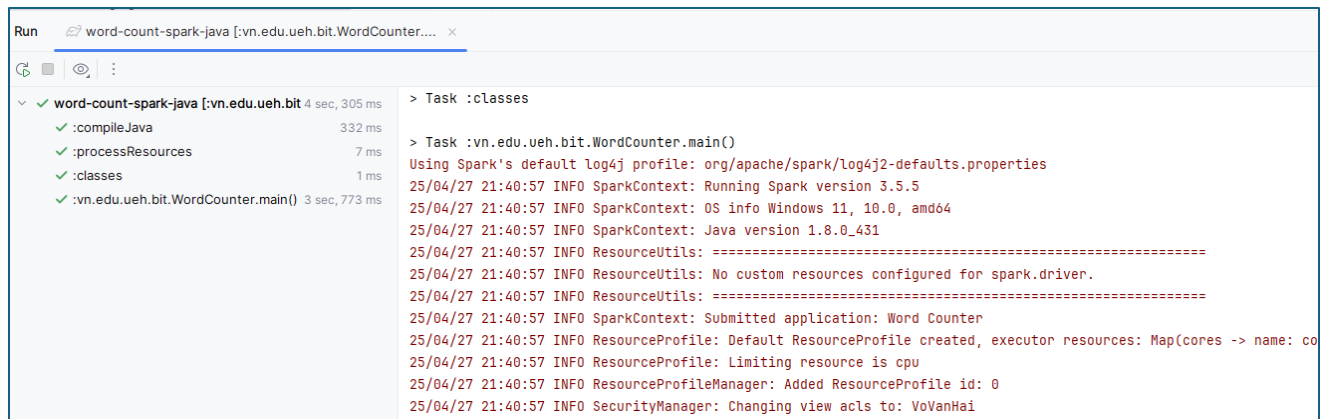
```
counts = counts.sortByKey();
```

Tiếp theo, chúng ta lưu kết quả.

```
counts.saveAsTextFile(outputFile);
```

## Thực thi

Việc thực thi chương trình trên spark đơn giản hơn rất nhiều vì chúng ta có thể chạy trực tiếp trên IDE.



```
Run word-count-spark-java [vn.edu.ueh.bit.WordCounter.... x
> Task :classes
> Task :vn.edu.ueh.bit.WordCounter.main()
Using Spark's default log4j profile: org/apache/spark/log4j2-defaults.properties
25/04/27 21:40:57 INFO SparkContext: Running Spark version 3.5.5
25/04/27 21:40:57 INFO SparkContext: OS info Windows 11, 10.0, amd64
25/04/27 21:40:57 INFO SparkContext: Java version 1.8.0_431
25/04/27 21:40:57 INFO ResourceUtils: =====
25/04/27 21:40:57 INFO ResourceUtils: No custom resources configured for spark.driver.
25/04/27 21:40:57 INFO ResourceUtils: =====
25/04/27 21:40:57 INFO SparkContext: Submitted application: Word Counter
25/04/27 21:40:57 INFO ResourceProfile: Default ResourceProfile created, executor resources: Map(cores -> name: co
25/04/27 21:40:57 INFO ResourceProfile: Limiting resource is cpu
25/04/27 21:40:57 INFO ResourceProfileManager: Added ResourceProfile id: 0
25/04/27 21:40:57 INFO SecurityManager: Changing view acls to: VoVanHai
```