

# Java-Success.com

Prepare to fast-track, choose & go places with 800+ Java & Big Data Q&As with lots of code & diagrams.

[Home](#) [Why? ▾](#) [300+ Java FAQs ▾](#) [300+ Big Data FAQs ▾](#) [Courses ▾](#)[👤 Membership ▾](#) [Your Career ▾](#)[Home](#) > [bigdata-success.com](#) > [Tutorials - Big Data](#) > [TUT - Spark Scala on Zeppelin](#) >

05: Spark on Zeppelin – semi-structured log file

## 05: Spark on Zeppelin – semi-structured log file

 Posted on [July 28, 2018](#)

This tutorial extends the series: [Spark on Apache Zeppelin Tutorials](#).

**Step 1:** Pull apache/zeppelin image from the docker hub, and build the image with the following command.

```
1 $ docker pull apache/zeppelin:0.7.3
2
```

“docker images” will show the image that was created.

### 300+ Java Interview FAQs

300+ Java FAQs



16+ Java Key Areas Q&amp;As



150+ Java Architect FAQs



80+ Java Code Quality Q&amp;As



150+ Java Coding Q&amp;As



### 300+ Big Data Interview FAQs

300+ Big Data FAQs



Tutorials - Big Data

TUT -  Starting Big Data

TUT - Starting Spark &amp; Scala

**Step 2:** Run the above image to create a container with the following command.

```
1 $ docker run -it -p 8080:8080 apache/zeppelin:0.7.3
2
```

**Step 3:** Go to a browser and type:  
"http://localhost:8080" to verify if zeppelin note book is accessible.

**Step 4:** Create a ssh session into the docker container from a terminal:

Firstly, get the container id with:

```
1 $ docker ps
2
```

Use the container id E.g. 89bce1cdc799 to ssh.

```
1 $ docker exec -it 89bce1cdc799 /bin/bash
2
```

**Step 5:** Once inside the docker container running on ubuntu, install "vim" editor.

```
1 $ apt-get update
2 $ apt-get install vim
3
```

**Step 6:** Create a temp.log file with vim.

```
1 $ vim temp.log
2
```

TUT - Starting with Python

TUT - Kafka

TUT - Pig

TUT - Apache Storm

TUT - Spark Scala on Zeppelin

TUT - Cloudera

TUT - Cloudera on Docker

TUT - File Formats

TUT - Spark on Docker

TUT - Flume

TUT - Hadoop (HDFS)

TUT - HBase (NoSQL)

TUT - Hive (SQL)

TUT - Hadoop & Spark

TUT - MapReduce

TUT - Spark and Scala

TUT - Spark & Java

TUT - PySpark on Databricks

TUT - Zookeeper

## 800+ Java Interview Q&As

300+ Core Java Q&As



300+ Enterprise Java Q&As



150+ Java Frameworks Q&As



120+ Companion Tech Q&As



Tutorials - Enterprise Java



Enter the following text by pressing "i" to insert.

```
1 ERROR 2009-09-19 09:56:01 main RDFDefaultErrorHand
2 INFO 2009-09-13 09:56:01 login Some.java:44 Some
3 INFO 2009-09-11 09:56:01 login Some.java:44 Some
4 WARNING 2009-09-09 09:56:01 main MarketData.java:4
5 INFO 2009-09-06 09:56:01 login Some.java:44 Some
6 INFO 2009-09-05 09:56:01 login Some.java:44 Some
7 INFO 2009-09-02 09:56:01 login Some.java:44 Some
8 INFO 2009-09-01 09:56:01 login Some.java:44 Some
9 ERROR 2009-08-25 09:56:01 main RDFDefaultErrorHand
10
```

exit vim by pressing "wq!".

**Step 7:** On zeppelin note book on the browser  
<http://localhost:8080>, create a note book and use  
"%sh" to list the files.

```
1
2 %sh
3
4 ls -ltr
5 pwd
6
```

**Step 8:** Create a spark RDD by reading this temp.log  
file. The path to temp.log file is  
"file:///zeppelin/temp.log"

```
1
2 %spark
3
4 val logsRdd = sc.textFile("file:///zeppelin/temp.log")
5 logsRdd.take(10).mkString("\n")
6
```

**Step 9:** Create a DataFrame as shown below:

```
1
```

```
2 %spark
3
4 import java.text.SimpleDateFormat
5 import java.sql.Date
6
7 val DATE_FORMAT = new SimpleDateFormat("yyyy-mm-dd")
8
9 case class Log(level: String, date: java.sql.Date)
10
11 val logsDf = logsRdd.map { line =>
12
13     val s = line.split(" ")
14     val logLevel = s(0)
15     val dateTime = DATE_FORMAT.parse(s(1) + " " +
16     val fileName = s(3).split(":")(0)
17     Log(logLevel,new Date(dateTime.getTime()), fi
18
19 }.toDF()
20
21 logsDf.show()
22
```

## Output:

```
1
2 +-----+-----+-----+
3 | level|      date|fileName|
4 +-----+-----+-----+
5 | ERROR|2009-01-19|    main|
6 |  INFO|2009-01-13|   login|
7 |  INFO|2009-01-11|   login|
8 |WARNING|2009-01-09|    main|
9 |  INFO|2009-01-06|   login|
10 |  INFO|2009-01-05|   login|
11 |  INFO|2009-01-02|   login|
12 |  INFO|2009-01-01|   login|
13 | ERROR|2009-01-25|    main|
14 +-----+-----+-----+
15
```

**Step 10:** Register as temp table so that you can run SQL queries to analyze the data.

```
1
2 %spark
3
4 logsDf.registerTempTable("logs")
5
```

## Step 11: Analyse with SQL – tabular and graph.

```
1  
2 %sql  
3  
4 select level, count(*) from logs group by level  
5
```



The image shows a Zeppelin SQL console interface. At the top, the SQL query is entered: `select level, count(*) from logs group by level`. Below the query, the results are displayed in a table. The table has two columns: 'level' and 'count(\*)'. The data rows are: INFO (6), ERROR (2), and WARNING (1). The status 'FINISHED' is visible in the top right corner of the console.

level	count(*)
INFO	6
ERROR	2
WARNING	1

zeppelin log file analysis

◀ 04: Spark on Zeppelin – DataFrame joins in Scala

40+ Apache Spark best practices & optimisation interview FAQs – Part-1

▶

## Disclaimer

The contents in this Java-Success are copyrighted and from EmpoweringTech pty Ltd. The EmpoweringTech pty Ltd has the right to correct or enhance the current content without any prior notice. These are general advice only, and one needs to take his/her own circumstances into consideration. The EmpoweringTech pty Ltd will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. Links to external sites do not imply endorsement of the linked-to sites. [Privacy Policy](#)