

DR. ALVIN'S PUBLICATIONS

# CONFIGURING THE APACHE SPARK SHELL

---

IN LOCAL MODE  
DR. ALVIN ANG



---

1 | PAGE

COPYRIGHTED BY DR ALVIN ANG  
WWW.ALVINANG.SG

# CONTENTS

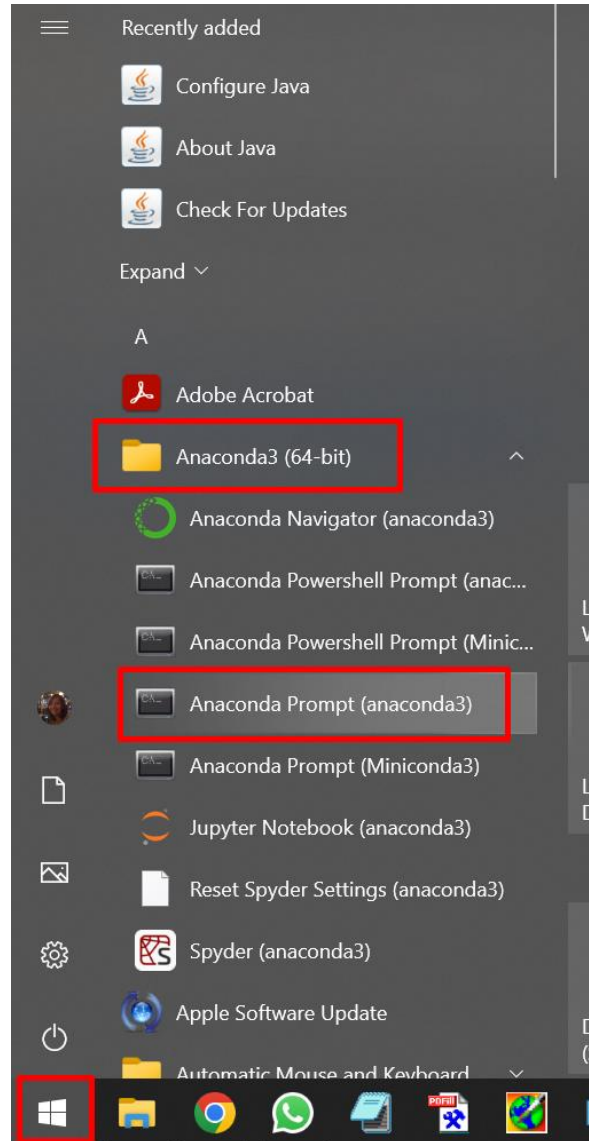
<b><i>I. Setup your PySpark Shell Configuration .....</i></b>	<b><i>3</i></b>
<b><i>II. Start your PySpark in Jupyter Notebook.....</i></b>	<b><i>5</i></b>
<b><i>A. Go to Jupyter Notebook .....</i></b>	<b><i>6</i></b>
<b><i>B. Now go to localhost:4040.....</i></b>	<b><i>7</i></b>
<b><i>C. Now click on Executors to take a look... ..</i></b>	<b><i>8</i></b>
<b><i>About Dr. Alvin Ang .....</i></b>	<b><i>9</i></b>

---

## I. SETUP YOUR PYSARK SHELL CONFIGURATION

---

- First, follow <https://www.alvinang.sg/s/Building-a-Apache-Spark-Local-Cluster-on-Windows-by-Dr-Alvin-Ang.pdf> to install PySpark fully in your Windows laptop.



```

Anaconda Prompt (anaconda3)
(base) C:\Users\User>cd C:\SparkProjectFolder first change your directory to your Spark working folder

(base) C:\SparkProjectFolder>pyspark --help
Usage: bin\pyspark.cmd [options]

Options:
--master MASTER_URL      spark://host:port, mesos://host:port, yarn,
                        k8s://https://host:port, or local (Default: local[*]).
--deploy-mode DEPLOY_MODE Whether to launch the driver program locally ("client") or
                        on one of the worker machines inside the cluster ("cluster")
                        (Default: client).
--class CLASS_NAME       Your application's main class (for Java / Scala apps).
--name NAME               A name of your application.
--jars JARS               Comma-separated list of jars to include on the driver
                        and executor classpaths.
--packages                Comma-separated list of maven coordinates of jars to include
                        on the driver and executor classpaths. Will search the local
                        maven repo, then maven central and any additional remote
                        repositories given by --repositories. The format for the
                        coordinates should be groupId:artifactId:version.
--exclude-packages       Comma-separated list of groupId:artifactId, to exclude while
                        resolving the dependencies provided in --packages to avoid
                        dependency conflicts.
--repositories            Comma-separated list of additional remote repositories to
                        search for the maven coordinates given with --packages.
--py-files PY_FILES       Comma-separated list of .zip, .egg, or .py files to place
                        on the PYTHONPATH for Python apps.
--files FILES             Comma-separated list of files to be placed in the working
                        directory of each executor. File paths of these files
                        in executors can be accessed via SparkFiles.get(fileName).
--archives ARCHIVES       Comma-separated list of archives to be extracted into the
                        working directory of each executor.

--conf, -c PROP=VALUE     Arbitrary Spark configuration property.
--properties-file FILE    Path to a file from which to load extra properties. If not
                        specified, this will look for conf/spark-defaults.conf.

--driver-memory MEM       Memory for driver (e.g. 1000M, 2G) (Default: 1024M).
--driver-java-options      Extra Java options to pass to the driver.
--driver-library-path      Extra library path entries to pass to the driver.

```

we are going to use 3 threads on Local Cluster Manager

we are working inside a local 'client' (jupyter IDE) and not inside the 'cluster'

we will use 2GB for JVM heap size, but by default is set at 1GB which is sufficient

Ignore the JVM heap size term for now... or google to find out more...

```

Anaconda Prompt (anaconda3) - pyspark --master local[3] --driver-memory 2G
Spark on YARN and Kubernetes only:
--num-executors NUM      Number of executors to launch (Default: 2).
                        If dynamic allocation is enabled, the initial number of
                        executors will be at least NUM.
--principal PRINCIPAL    Principal to be used to login to KDC.
--keytab KEYTAB          The full path to the file that contains the keytab for the
                        principal specified above.

Spark on YARN only:
--queue QUEUE_NAME       The YARN queue to submit to (Default: "default").

(base) C:\SparkProjectFolder>pyspark --master local[3] --driver-memory 2G startup your local spark shell with this command
[I 2023-02-28 11:12:37.893 LabApp] JupyterLab extension loaded from C:\Users\User\anaconda3\lib\site-packages\jupyterlab
[I 2023-02-28 11:12:37.893 LabApp] JupyterLab application directory is C:\Users\User\anaconda3\share\jupyterlab
[I 11:12:37.893 NotebookApp] Serving notebooks from local directory: C:\SparkProjectFolder
[I 11:12:37.893 NotebookApp] Jupyter Notebook 6.4.12 is running at:
[I 11:12:37.893 NotebookApp] http://localhost:8888/?token=8cd4c36a28f1bf8addf83c578908184ce8f6b941b75a5a2a
[I 11:12:37.893 NotebookApp] or http://127.0.0.1:8888/?token=8cd4c36a28f1bf8addf83c578908184ce8f6b941b75a5a2a
[I 11:12:37.893 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 11:12:37.956 NotebookApp]

To access the notebook, open this file in a browser:
file:///C:/Users/User/AppData/Roaming/jupyter/runtime/nbsrvr-7564-open.html
Or copy and paste one of these URLs:
http://localhost:8888/?token=8cd4c36a28f1bf8addf83c578908184ce8f6b941b75a5a2a
or http://127.0.0.1:8888/?token=8cd4c36a28f1bf8addf83c578908184ce8f6b941b75a5a2a
[I 11:12:46.803 NotebookApp] Kernel started: 5842fbc7-68b6-44bd-bf99-95663ab799f5, name: python3
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/02/28 11:12:53 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
[I 11:13:12.184 NotebookApp] Saving file at /Untitled.ipynb
C:\Users\User\anaconda3\lib\site-packages\nbformat\_init_.py:128: MissingIDFieldWarning: Code cell is missing an id field, this will become a hard error in future nbforma
t versions. You may want to use 'normalize()' on your notebooks before validations (available since nbformat 5.1.4). Previous versions of nbformat are fixing this issue tra
nsparently, and will stop doing so in the future.
  validate(nb)
C:\Users\User\anaconda3\lib\site-packages\notebook\services\contents\manager.py:353: MissingIDFieldWarning: Code cell is missing an id field, this will become a hard error
in future nbformat versions. You may want to use 'normalize()' on your notebooks before validations (available since nbformat 5.1.4). Previous versions of nbformat are fixi
ng this issue transparently, and will stop doing so in the future.
  validate_nb(model['content'])

```

## II. START YOUR PYSPARK IN JUPYER NOTEBOOK

localhost:8888/tree

Jupyter

Files Running Clusters

Select items to perform actions on them.

Upload New

Name	Last Modified	File size
FIRSTTRY.ipynb	8 days ago	2.97 kB
Untitled.ipynb	Running seconds ago	665 B

click this or start a new ipynb instance

localhost:4040/jobs/

Spark 3.3.1

Jobs Stages Storage Environment Executors

PySparkShell application UI

### Spark Jobs (?)

User: User  
Total Uptime: 2.6 min  
Scheduling Mode: FIFO

Event Timeline

Enable zooming

Executors

- Added
- Removed

Jobs

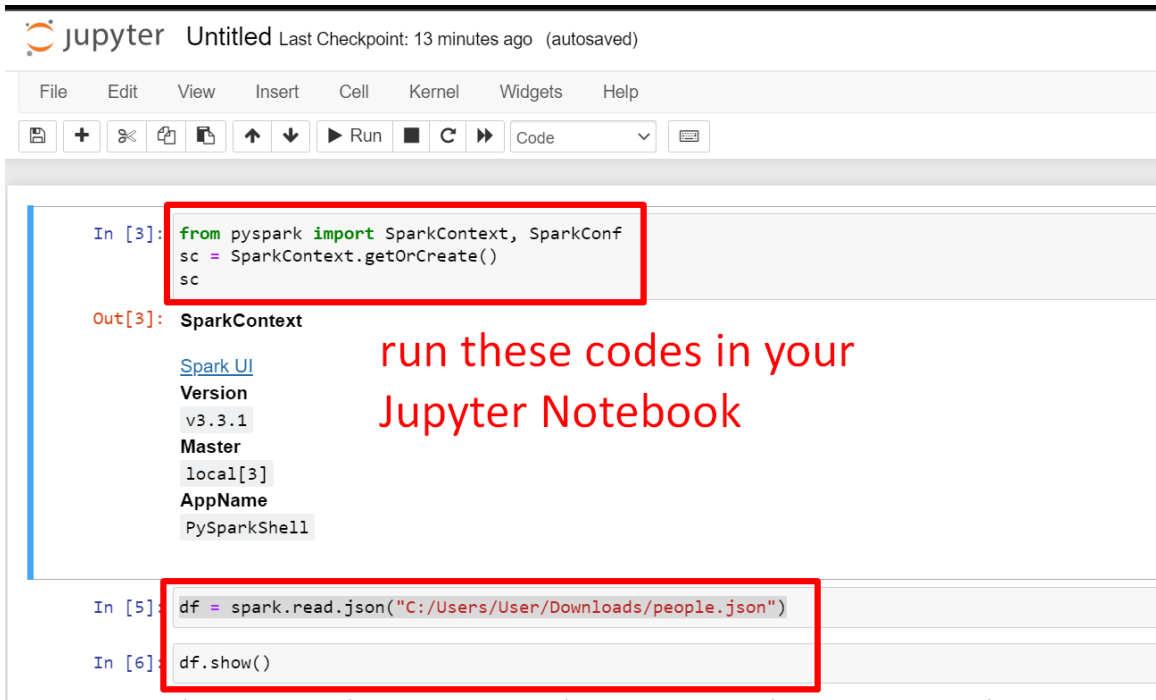
- Succeeded
- Failed
- Running

Time	Executors	Jobs
11:12:53		
11:12:54		
11:12:55		

go localhost:4040 and look at all jobs

Currently nothing is running because no jobs have been runned...

## A. GO TO JUPYTER NOTEBOOK



The screenshot shows a Jupyter Notebook titled "Untitled" with a last checkpoint 13 minutes ago. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations and execution. The notebook content consists of three code cells:

```
In [3]: from pyspark import SparkContext, SparkConf
        sc = SparkContext.getOrCreate()
        sc
```

The output for cell [3] is a **SparkContext** object with the following attributes:

- Spark UI
- Version: v3.3.1
- Master: local[3]
- AppName: PySparkShell

Cell [5] contains the code to read a JSON file:

```
In [5]: df = spark.read.json("C:/Users/User/Downloads/people.json")
```

Cell [6] contains the code to display the DataFrame:

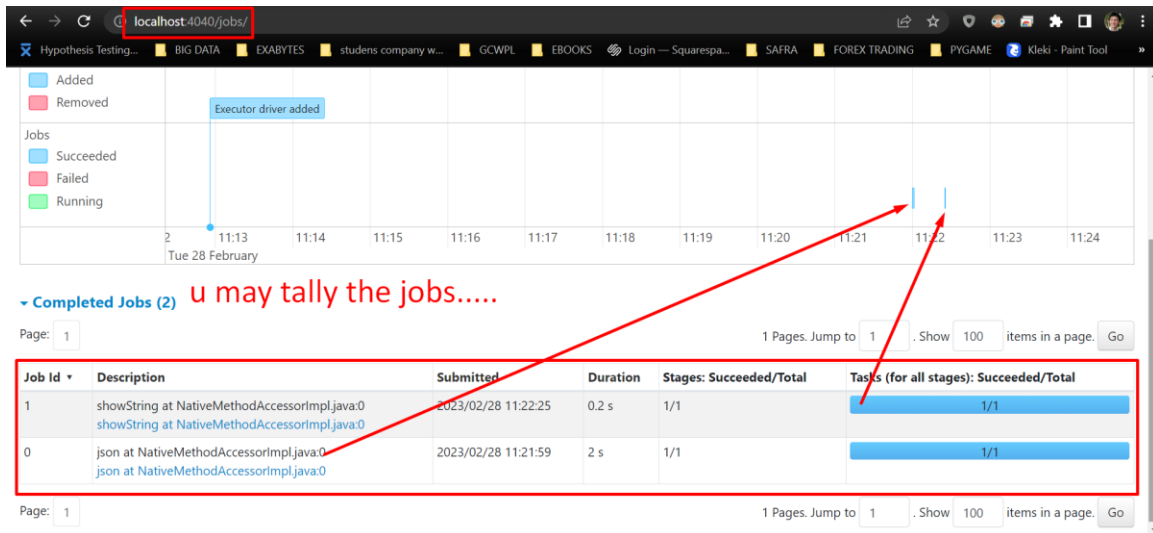
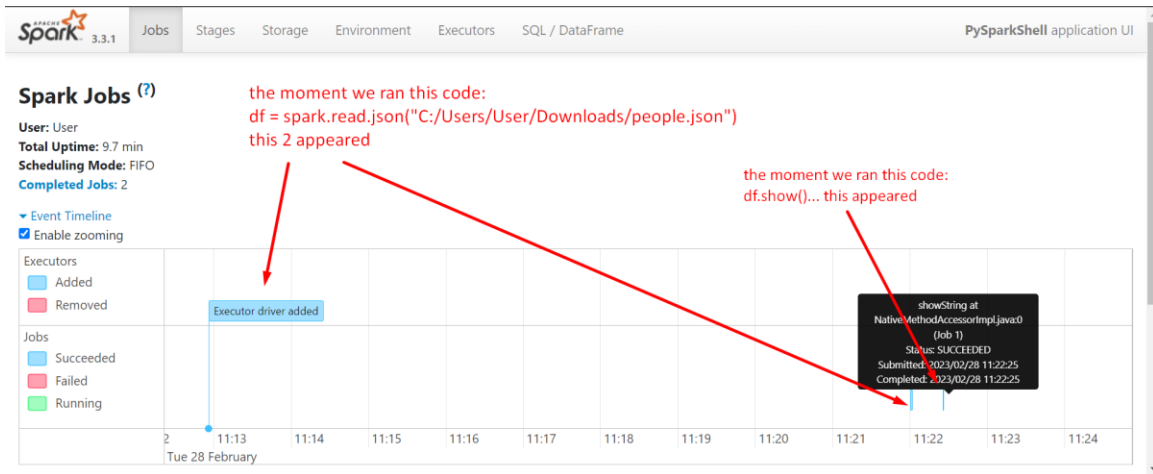
```
In [6]: df.show()
```

Red boxes highlight the code in cells [3], [5], and [6]. A red text overlay in the center of the notebook area reads: "run these codes in your Jupyter Notebook".

The json file is here:

<https://www.alvinang.sg/s/people.json>

**B. NOW GO TO LOCALHOST:4040**



C. NOW CLICK ON EXECUTORS TO TAKE A LOOK...

**Executors Summary**

	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Excluded
Active(1)	0	30.1 KIB / 912.3 MIB	0.0 B	3	0	0	2	2	18 min (0.4 s)	1.6 MIB	0.0 B	0.0 B	0
Dead(0)	0	0.0 B / 0.0 B	0.0 B	0	0	0	0	0	0.0 ms (0.0 ms)	0.0 B	0.0 B	0.0 B	0
Total(1)	0	30.1 KIB / 912.3 MIB	0.0 B	3	0	0	2	2	18 min (0.4 s)	1.6 MIB	0.0 B	0.0 B	0

**Executors**

Show 20 entries

we have 3 threads but system only shows 1 driver (which is also the executor)

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Thread Dump
driver	DESKTOP-2U1FVFJF:50973	Active	0	30.1 KIB / 912.3 MIB	0.0 B	3	0	0	2	2	18 min (0.4 s)	1.6 MIB	0.0 B	0.0 B	Thread Dump

localhost:4040 1 of 1 entries

we asked for 2GB but they only gave around 1 GB



---

ABOUT DR. ALVIN ANG

---



Dr. Alvin Ang earned his Ph.D., Masters and Bachelor degrees from NTU, Singapore. He is a scientist, entrepreneur, as well as a personal/business advisor. More about him at [www.AlvinAng.sg](http://www.AlvinAng.sg).