Setting up PySpark in Colab

Spark is written in the Scala programming language and requires the Java Virtual Machine (JVM) to run. Therefore, our first task is to download Java.

```
!apt-get install openjdk-8-jdk-headless -qq > /dev/null
```

Next, we will install Apache Spark 3.0.1 with Hadoop 2.7 from here.

```
!wget https://downloads.apache.org/spark/spark-3.1.1/spark-3.1.1-bin-hadoop2.7.tgz
```

```
    --2021-03-21 15:52:56--  https://downloads.apache.org/spark/spark-3.1.1/spark-3.
    Resolving downloads.apache.org (downloads.apache.org)... 88.99.95.219, 2a01:4f8:
    Connecting to downloads.apache.org (downloads.apache.org)|88.99.95.219|:443... c
    HTTP request sent, awaiting response... 200 OK
    Length: 224374704 (214M) [application/x-gzip]
    Saving to: ‘spark-3.1.1-bin-hadoop2.7.tgz’

    spark-3.1.1-bin-had 100%[===================>] 213.98M  19.0MB/s    in 12s

    2021-03-21 15:53:08 (17.8 MB/s) - ‘spark-3.1.1-bin-hadoop2.7.tgz’ saved [2243747
```

Now, we just need to unzip that folder.

```
!tar xf spark-3.1.1-bin-hadoop2.7.tgz
```

There is one last thing that we need to install and that is the findspark library. It will locate Spark on the system and import it as a regular library.

```
!pip install -q findspark
```

Now that we have installed all the necessary dependencies in Colab, it is time to set the environment path. This will enable us to run Pyspark in the Colab environment.

```
import os
os.environ["JAVA_HOME"] = "/usr/lib/jvm/java-8-openjdk-amd64"
os.environ["SPARK_HOME"] = "/content/spark-3.1.1-bin-hadoop2.7"
```

Time for the real test!

We need to locate Spark in the system. For that, we import findspark and use the findspark.init() method.

```
import findspark
findspark.init()
```

Bonus – If you want to know the location where Spark is installed, use findspark.find()

```
findspark.find()
```

```
    '/content/spark-3.1.1-bin-hadoop2.7'
```

Now, we can import SparkSession from pyspark.sql and create a SparkSession, which is the entry point to Spark.

You can give a name to the session using appName() and add some configurations with config() if you wish.

```
from pyspark.sql import SparkSession

spark = SparkSession.builder\
        .master("local")\
        .appName("Colab")\
        .config('spark.ui.port', '4050')\
        .getOrCreate()
```

Finally, print the SparkSession variable.

```
spark
```

SparkSession - in-memory

SparkContext

Spark UI

Version
    `v3.1.1`
Master
    `local`
AppName
    `Colab`

If you want to view the Spark UI, you would have to include a few more lines of code to create a

```
!wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip
!unzip ngrok-stable-linux-amd64.zip
get_ipython().system_raw('./ngrok http 4050 &')
!curl -s http://localhost:4040/api/tunnels
```

```
--2021-03-21 15:59:54--  https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-
Resolving bin.equinox.io (bin.equinox.io)... 34.192.67.182, 52.6.97.115, 34.226.
Connecting to bin.equinox.io (bin.equinox.io)|34.192.67.182|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13773305 (13M) [application/octet-stream]
Saving to: ‘ngrok-stable-linux-amd64.zip’

ngrok-stable-linux- 100%[===================>]  13.13M  18.3MB/s    in 0.7s

2021-03-21 15:59:55 (18.3 MB/s) - ‘ngrok-stable-linux-amd64.zip’ saved [13773305

Archive:  ngrok-stable-linux-amd64.zip
  inflating: ngrok
{"tunnels":[],"uri":"/api/tunnels"}
```

```
!curl -s http://localhost:4040/api/tunnels
```

```
":"/api/tunnels/command_line%20%28http%29","public_url":"http://620435e0ad2b.ngrok
```