Table of Contents

ASPARC STARTER DEVOPS GUIDE

1. WAY OF WORKING

1.1. Issues handling

Each proper time spent on planning saves 10 times more in execution, thus the tasks and aktivities related to this tool are tracked via the issue-tracker tool:
https://github.com/YordanGeorgiev/issue-tracker
and could be found @:
https://docs.google.com/spreadsheets/d/1-oYPtBM8PG_FUogk40RDmcM_Xzq91Tb81Zlyi0cMwYQ/edit#gid=135774576
the public url for the sheet is :
https://docs.google.com/spreadsheets/d/e/2PACX-1vR0wo5N32EpubwxBfeFxi6X-eOmXwOPg4WSyA4qBSz1Yu0EyU34jl0xICgWzrFUSeEA_aC4RF7LRqx9/pubhtml

1.2. Documentation

The purpose of the tool is to "grasp the concept of apache spark", thus a proper documentation set is created as well.
This is the documentation set.
The UserStories:
https://github.com/YordanGeorgiev/aspark-starter/blob/master/doc/md/aspark-starter-user-stories.md
The Requirements:
https://github.com/YordanGeorgiev/aspark-starter/blob/master/doc/md/aspark-starter-requirements.md
The DevOps Guide:
https://github.com/YordanGeorgiev/aspark-starter/blob/master/doc/md/aspark-starter-devops-guide.md
The Features and Functionalities Description:
https://github.com/YordanGeorgiev/aspark-starter/blob/master/doc/md/aspark-starter-features-and-functionalities.md
The same files could be retrieved in pdf format from the doc/pdf dir of the project.

2. NAMING CONVENTIONS

2.1. Bash scripts

2.1.1. Dirs naming conventions

The dir structure should be logical and a person navigating to a dir should almost understand what is to be find in thre by its name. While creating new dirs the main principle is that the more general the subject of the dir the upper in the dir hierarchy it should stay.

2.1.2. Attemp to not mix code from different languages / run-times in the same dirs

Avoid as much as possible the mixing code from different languages / run-times in the same dirs

2.1.3. Root Dirs naming conventions

The root dirs and named as follows:
bin - contains the produced binaries for th project

cnf - for the configuration
dat - for the data of the app
lib - for any external libraries used
src - for the source code of the actual projects and subprojects

2.1.4. Bash scripts naming conventions

Do not use capital letters - they are too noisy.

2.2. Scala code capitalization styles and naming conventions

2.2.1. Pascal case usage

Use in application wide global variables

```
val ProductInstanceDir
```

2.2.2. Camel Case usage

Use in local class variables

3. INSTALLATIONS AND CONFIGURATIONS

3.1. Install Java Development Kit 1.8

Install Java Development Kit 1.8 as follows:

```
# update your Ubuntu repositories
sudo apt-get update
# install the open jdk
sudo apt-get install -y openjdk-8-jdk
```

3.1.1. Configure java_home

Configure java_home env var to the the java_opts file.

```
echo 'export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64' >> ~/.java_opts.host-name
```

3.1.2. Verify the JDK installation and configuration

Verify the JDK installation and configuration as follows:

```
# and verify
java -version
java version "1.8.0_101"
Java(TM) SE Runtime Environment (build 1.8.0_101-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.101-b13, mixed mode)
```

3.2. Install Scala

The scala libs will be installed with the sbt build tool.

3.3. Install sbt

Install sbt scala build tool by following the instructions in the following url:

http://www.scala-sbt.org/0.13/docs/Installing-sbt-on-Linux.html

```
echo "deb https://dl.bintray.com/sbt/debian /" | sudo tee -a /etc/apt/sources.list.d/sbt.list
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv 2EE0EA64E40A89B84B2DF73499E82A75642AC823
sudo apt-get update
sudo apt-get install sbt
which sbt
```

3.4. Install apache spark

3.4.1. Download the latest stable Apache Spak package

Download the spak package with curl as follows:

```
export dir=/var/pckgs/
sudo mkdir -p $dir ; sudo chown -R ysg:ysg $dir; cd $dir

curl -O https://d3kbcqa49mib13.cloudfront.net/spark-2.2.0-bin-hadoop2.7.tgz -o /var/pckgs/spark-2.2.0-bin-hadoop2.7.tgz
```

3.4.2. Unpack and deploy

Uncompress the package and deploy it as follows:

```
 tar -zxvf /var/pckgs/spark-2.2.0-bin-hadoop2.7.tgz
mv -v spark-2.2.0-bin-hadoop2.7/ spark
mv -v spark /usr/lib/
sudo mv -v spark /usr/lib/
```

3.4.3. Add env vars

Add the following env vars

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export SPARK_HOME=/usr/lib/spark
export PATH=$PATH:$SPARK_HOME

# reload the env vars
source ~/.profile_opts
```

3.4.4. Verify the installation

Verify the installation by startin the spark shell

```
spark-shell

Spark context available as 'sc' (master = local[*], app id = local-1505242880618).
Spark session available as 'spark'.
Welcome to
    ____              __
   / __/__  ___ _____/ /__
```

```
   _\ V _ V _ `/ __/  '_/
  /___/ .__/\_,_/_/ /_/\_\   version 2.2.0
     /_/

Using Scala version 2.11.8 (OpenJDK 64-Bit Server VM, Java 1.8.0_131)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```

3.5. Configure the Ubuntu repositories

Configure the Ubuntu repositories

```
sudo add-apt-repository "deb http://apt.postgresql.org/pub/repos/apt/ xenial-pgdg main"

sudo apt-get update
sudo apt-get install postgresql-9.6
```

3.6. Add the media keys

Add the media keys as follows:

```
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
```

3.7. Install the postgre package with apt

Install the postgre package with apt

```
# update your repos
sudo apt-get update

# install the postgresql binary
sudo apt-get install postgresql postgresql-contrib

# enable postgre
sudo update-rc.d postgresql enable
```

3.8. Change the postgre user password

Configure the Ubuntu repositories

```
sudo passwd postgres
# Type a pw - add to your password manager !!!

# and verify
su - postgres
```

3.8.1. start the postgreSQL

Start the postgreSQL by issuing the following command

```
sudo /etc/init.d/postgresql start
```

3.8.2. Start the psql client as the postgres shell user

Start the psql client as the postgres shell user
source:
http://dba.stackexchange.com/a/54253/1245

```
sudo su - postgres
# start the psql client
psql

# the psql prompt should appear as
# postgres=#

# list the databases
\l
#and quit
\q
```

3.8.3. Create the pgsql user

Create the pgsql user and grant him the privileges to create dbs and to connect to the postgres db.
You could alternatively configure different way of authenticatio according to the options provided in this stackoverflow answer:
http://stackoverflow.com/a/9736231/65706

```
# create the pgsql user to be the same as the shell
# user you are going to execute the scripts with
sudo su - postgres  -c "psql -c 'CREATE USER '$USER' ;'"

# grant him the priviledges
sudo su - postgres  -c "psql -c 'grant all privileges on database postgres to '$USER' ;'"

# grant him the privilege to create db's
sudo su - postgres  -c "psql -c 'ALTER USER '$USER' CREATEDB;'"

sudo su - postgres  -c 'psql -c "select * from information_schema.role_table_grants
 where grantee="'"$USER"'";'"
```

3.8.4. add the uuid generation capability enabling extension

add the uuid generation capability enabling extension

```
sudo su - postgres  -c "psql template1 -c 'CREATE EXTENSION IF NOT EXISTS \"uuid-ossp\";'"

sudo su - postgres  -c "psql template1 -c 'CREATE EXTENSION IF NOT EXISTS \"pgcrypto\";'"
```

3.8.5. Install the dblink extension as follows

Install the dblink extension as follows

```
sudo su - postgres  -c "psql template1 -c 'CREATE EXTENSION IF NOT EXISTS \"dblink\";'"
```

3.9. Install the perl modules (optional)

Install the perl module by first installing the server development package

```
# check which server development packages are available
sudo apt-cache search postgres | grep -i server-dev | sort

# install it
sudo apt-get install -y postgresql-server-dev-9.6

# install the DBD::Pg module
sudo perl -MCPAN -e 'install DBD::Pg'

sudo perl -MCPAN -e 'Tie::Hash::DBD'
```

3.10. Install and configure Hadoop

This section has been created largely by following this tutorial:
http://www.bogotobogo.com/Hadoop/BigData_hadoop_Install_on_ubuntu_16_04_single_node_cluster.php

3.10.1. Add the hadoop group

Add the hadoop Linux group as follows:

```
export group=hadoop
export gid=1001
sudo groupadd -g "$gid" "$group"
sudo cat /etc/group | grep --color "$group"
```

3.10.2. Add the hadoop user

Add the hduser Linux user as follows:

```
export user=hduser
export uid=1001
export home_dir=/home/$user
export desc="the hadoop group"
#how-to add an user
sudo useradd --uid "$uid" --home-dir "$home_dir" --gid "$group" \
--create-home --shell /bin/bash "$user" \
--comment "$desc"
sudo cat /etc/passwd | grep --color "$user"
groups "$user"
```

3.10.3. Configure the ssh keys for the hduser

Hadoop requires SSH access to manage its nodes, i.e. remote machines plus this local machine.

```
sudo su - hduser
```

```
# create the ssh keys of rsa type
ssh-keygen -t rsa

# add localhost to the trusted hosts
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys

# verify that ssh works
ssh localhost
```

3.10.4. Fetch and install hadoop

Fetch and install hadoop as follows:

```
pckgs_dir=/var/pckgs/apache
sudo mkdir -p "$pckgs_dir" ; cd "$pckgs_dir"

install_dir=/usr/local/hadoop
sudo mkdir -p "$install_dir" ; cd "$install_dir"

# download the hadoop package
curl -O 'http://mirror.netinch.com/pub/apache/hadoop/common/hadoop-2.7.4/hadoop-2.7.4.tar.gz'

# uncompress it
sudo gzip -dc hadoop-2.7.4.tar.gz | tar -C "$install_dir" -xvf -

# go back to the starting dir
cd -

echo "created the following dir"
sudo find $install_dir/hadoop-2.7.4/ -type d -maxdepth 2

sudo chown -Rv hduser:hadoop /usr/local/hadoop/
```

3.10.5. Edit the hadoop-env.sh file

Edit the hadoop-env.sh file

```
cd $product_instance_dir

# edit the hadoop-env.sh file as follows:
diff /usr/local/hadoop/hadoop-2.7.4/etc/hadoop/hadoop-env.sh cnf/hosts/host-name/usr/local/hadoop/hadoop-
2.7.4/etc/hadoop/hadoop-env.sh

sudo vim -o `find /usr/local/hadoop -name hadoop-env.sh`
```

3.10.6. create the hadoop data dir

create the hadoop data dir

```
hadoop_tmp_data_dir=/var/hadoop/tmp
sudo mkdir -p "$hadoop_tmp_data_dir"
sudo chown -Rv hduser:hadoop $hadoop_tmp_data_dir
ls -al $hadoop_tmp_data_dir
```

3.10.7. copy the map-reg.site.template file

copy the mapred-site.xml.template file

```
sudo cp -v /usr/local/hadoop/hadoop-2.7.4/etc/hadoop/mapred-site.xml.template /usr/local/hadoop/hadoop-2.7.4/etc/hadoop/mapred-site.xml
```

3.10.8. Edit the hdfs-site.xml

edit the hdfs-site.xml as follows

```
sudo mkdir -p /var/hadoop/dat/hdfs/name_node
sudo mkdir -p /var/hadoop/dat/hdfs/data_node
sudo chown -Rv hduser:hadoop /var/hadoop/
find /var/hadoop

sudo diff /usr/local/hadoop/hadoop-2.7.4/etc/hadoop/hdfs-site.xml cnf/hosts/host-name/usr/local/hadoop/hadoop-2.7.4/etc/hadoop/hdfs-site.xml
```

3.10.9. format the hadoop file system

format the hadoop file system as follows.
Note that hadoop namenode -format command should be executed once before we start using Hadoop, otherwise all the data will be destroyed …

```
sudo cp -v /home/ysg/.hadoop_opts.host-name /home/hduser/
sudo cp -v /home/ysg/.profile_opts.host-name /home/hduser/
sudo su - root -c "echo 'source ~/.profile_opts.host-name ' >> /home/hduser/.bashrc"

sudo su - hduser -c "/usr/local/hadoop/hadoop-2.7.4/bin/hdfs namenode -format"

# SDOUT
# 17/09/18 14:46:48 INFO util.ExitUtil: Exiting with status 0
# 17/09/18 14:46:48 INFO namenode.NameNode: SHUTDOWN_MSG:
# /************************************************************
# SHUTDOWN_MSG: Shutting down NameNode at host-name/127.0.1.1
# ************************************************************/
```

3.10.10. start the single node cluster

After running the commands bellow point to the following url:
http://host-name:50070/dfshealth.html#tab-overview

```
cd /usr/local/hadoop/hadoop-2.7.4/sbin
bash start-dfs.sh

# 17/09/18 15:00:35 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes
# where applicable
# Starting namenodes on [localhost]
# localhost: namenode running as process 23006. Stop it first.
# localhost: datanode running as process 23169. Stop it first.
# Starting secondary namenodes [0.0.0.0]
# The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.
# ECDSA key fingerprint is SHA256:2FhGsEKZO07xN4MPVcSVmSW0Clx9PKDtfANEqo3iHqg.
# Are you sure you want to continue connecting (yes/no)? yes
# 0.0.0.0: Warning: Permanently added '0.0.0.0' (ECDSA) to the list of known hosts.
# 0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop/hadoop-2.7.4/logs/hadoop-hduser-secondarynamenode-host-name.out
# 17/09/18 15:00:51 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes
# where applicable
#

start-yarn.sh
```

3.10.11. Verify that the everything is up-and-running

We can check if it's really up and running:
hduser@laptop:/usr/local/hadoop/sbin$

```
jps
23169 DataNode
24194 SecondaryNameNode
25306 ResourceManager
26429 NodeManager
23006 NameNode
26543 Jps
```

3.10.12. Create the data dir to write to in the hdfs

Create the data dir to write to in the hdfs as follows:

```
# add also yourself to the hadoop group
sudo usermod -a -G hadoop $USER
sudo su -l hduser -c "/usr/local/hadoop/hadoop-2.7.4/bin/hadoop fs -chown -R ysg:ysg /var/aspark-starter/dat"
```

3.11. Install python and pip v3.6

Install python and pip v3.5 as follows:

```
# this package will aslso be installed automatically if you run the ubuntu preq packages installer script
bash src/bash/aspark-starter/install-prerequisites-for-aspark-starter-on-ubuntu.sh
```

```
# OR
sudo apt-get install -y python-3
sudo apt-get install -y pip3
sudo apt-get install -y python3-venv
```

3.11.1. run the setup virtual environment script

run the setup virtual environment script as follows:

```
# this package will aslso be installed automatically if you run the ubuntu preq packages installer script
bash src/bash/aspark-starter/install-prerequisites-for-aspark-starter-on-ubuntu.sh

# OR
sudo apt-get install -y python-3
sudo apt-get install -y pip3
sudo apt-get install -y python3-venv
```

3.12. Install docker on Ubuntu 17.04

The docker installation on Ubuntu 17.04 has its own install script (Should your OS be a different one) , follow the install instructions on the following page:
https://docs.docker.com/engine/installation/linux/docker-ce/ubuntu/

```
# install the docker-ce package by running the install-script
bash src/bash/aspark-starter/install-docker-on-ubuntu-17.04.sh
```

4. OPERATIONS

4.1. Run the examples

You can run all the examples as follows:

```
# check the actions to run
  cat src/bash/aspark-starter/tests/run-aspark-starter-tests.lst

# STDUOT
# sbt-compile-verbose
# sbt-clean-compile
# sbt-compile
# sbt-stage
# sbt-run

bash src/bash/aspark-starter/test-aspark-starter.sh

# now the tool will start producing output

# 2017-09-14 08:26:11      START test-aspark-starter test run report
#
# result  start-time  stop-time   action-name
result  start-time  stop-time   action-name
#   ok    18:22:41 18:24:01 sbt-compile-verbose
#   ok    18:24:02 18:24:50 sbt-clean-compile
```

```
#   ok    18:24:50 18:25:00 sbt-compile
#   ok    18:25:00 18:25:20 sbt-stage
#   ok    18:25:21 18:26:39 run-local-app
```

4.2. Docker image building and containers handling

First some theory :

An image is an ordered collection of root filesystem changes and the corresponding execution parameters for use within a container runtime. Images are read-only.

https://docs.docker.com/engine/reference/glossary/#image

A docker container is an active (or inactive if exited) stateful instantiation of an image.

4.2.1. Create a base image

Use the following sr (adapt for zesty):

https://docs.docker.com/engine/userguide/eng-image/baseimages/#create-a-full-image-using-tar

Chk also:

https://linux.die.net/man/8/debootstrap

```
sudo debootstrap zesty zesty > /dev/null
sudo tar -C zesty -c . | docker import - zesty
# sha256:f4b12ba0096b89a2d591a855778a071420e171852bc4809ec680bfcdeb0532b7

docker run zesty cat /etc/lsb-release
# DISTRIB_ID=Ubuntu
# DISTRIB_RELEASE=17.04
# DISTRIB_CODENAME=zesty
# DISTRIB_DESCRIPTION="Ubuntu 17.04"
```

4.2.2. Push a basic image to the docker cloud

```
docker commit -m "Added latest zesty" -a "NAME" 50942fd04f50 298fd226fcbc40d0b2a3a39258abc/aspark-starter:latest
# STDOUT
# sha256:fa2c3cede57001bf5632017919ee62648432ad9101a1cedc1bbaef45b531e6d7

# login to the docker cloud
docker login

# username:
# 298fd226fcbc40d0b2a3a39258abc
# type pw
# should see Login Succeeded

docker push 298fd226fcbc40d0b2a3a39258abc/aspark-starter:latest
55b61fb42e68: Pushed
latest: digest: sha256:da1f258b69f49ed0a05cfe1cf66494b96e7d556828b19b28a4303836c33c1056 size: 530
```

4.2.3. Build the docker image

Go to the Dockerfile of the current environment.

Build the image by issuing the following commands:

```
cd src/docker/aspark-handler/dev

# build the image
sudo docker build .

# now you should be able to see the newly build image by:
sudo docker images
```

4.2.4. List the docker images and attach

List the docker images and attach to an already running image

```
sudo docker images --all
```

4.2.5. Start the docker image

Get the upper most id of the docker images command and run it as follows:

```
sudo docker run -i -t 3ad09ddd4f36 /bin/bash
sudo docker run --entrypoint docker-entrypoint.sh 3ad09ddd4f36 redis-server



# how-to check all the running containers
docker ps -a

# how-to attach to a running already container
docker exec -t -i f97566aa2317 /bin/bash

# how-to remove a container by id
sudo docker rm -v f5c767a66979

# if the start did not work well
# remove all the running containers
sudo docker rm $(sudo docker ps -aq)
# remove all the images
sudo docker rmi $(sudo docker images -q)
```

5. INFORMATION SOURCES

This section contains good information sources

5.1. The spark v2.2.0 official docs

The spark v2.2.0 official docs:
https://spark.apache.org/docs/2.2.0/

5.2. Mastering Apache Spark

https://jaceklaskowski.gitbooks.io/mastering-apache-spark/content/

5.3. Kiril Pavlov's spark blog

Different types of joins :
http://kirillpavlov.com/blog/2016/04/23/beyond-traditional-join-with-apache-spark/

5.4. Overall tutorials

The hadoop installation and configuration has been largely replicated and slightly adjusted from this source:
https://www.edureka.co/blog/spark-tutorial/