PySpark on Kubernetes Word Count And PageRank

Presenter: Saron Haile
ID: 20069

TABLE OF CONTENT

INTRODUCTION

Project Overview:

This project focuses on implementing Word Count and PageRank using PySpark on Apache Spark running on Kubernetes. The aim is to utilize cloud infrastructure to efficiently handle large-scale data processing tasks.

Objectives:

- Set up a Kubernetes cluster on Google Kubernetes Engine (GKE).
- Deploy Apache Spark on the cluster.
- Execute Word Count and PageRank algorithms using PySpark.

DESIGN

Approach:

1. **Kubernetes Cluster Setup:**
   - Use GKE to create a cluster.
   - Configure the cluster to support Spark jobs.
2. **Spark Deployment:**
   - Utilize Helm charts for Apache Spark deployment.
   - Set up persistent storage using NFS Server Provisioner.
3. **Application Development:**
   - Prepare the application JAR file for Spark.
   - Develop PySpark scripts for Word Count and PageRank.

DESIGN...


```
I have no name!@spark-master-0:/opt/bitnami/spark$ ls /opt/bitnami/spar
k
LICENSE  README.md  conf          examples    licenses  sbin  work
NOTICE   RELEASE    conf.default  jars        logs      tmp   yarn
R        bin        data          kubernetes  python    venv
I have no name!@spark-master-0:/opt/bitnami/spark$ /opt/bitnami/spark/b
in/spark-submit --version
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 3.5.1
      /_/

Using Scala version 2.12.18, OpenJDK 64-Bit Server VM, 17.0.11
Branch HEAD
Compiled by user heartsavior on 2024-02-15T11:24:58Z
Revision fd86f85e181fc2dc0f50a096855acf83a6cc5d9c
Url https://github.com/apache/spark
Type --help for more information.
I have no name!@spark-master-0:/opt/bitnami/spark$
```

IMPLEMENTATION and TESTING

Step 2: Spark Deployment:

1. Add the stable Helm repository and install the NFS Server Provisioner.
2. Create a persistent disk volume and a pod to use NFS.
3. Apply the YAML descriptor for the NFS setup

```
I have no name!@spark-master-0:/opt/bitnami/spark$ ls /opt/bitnami/spar
k
LICENSE   README.md  conf          examples    licenses  sbin  work
NOTICE    RELEASE    conf.default  jars        logs      tmp   yarn
R         bin        data          kubernetes  python    venv
I have no name!@spark-master-0:/opt/bitnami/spark$ /opt/bitnami/spark/b
in/spark-submit --version
Welcome to
      ____              __
     / ___/ __  ___  ____/ /__
     \ \/ _ \/ _ `/ / __/  '_/
    /___/ .__/\_,_/_/ /_/\_\   version 3.5.1
       /_/

Using Scala version 2.12.18, OpenJDK 64-Bit Server VM, 17.0.11
Branch HEAD
Compiled by user heartsavior on 2024-02-15T11:24:58Z
Revision fd86f85e181fc2dc0f50a096855acf83a6cc5d9c
Url https://github.com/apache/spark
Type --help for more information.
I have no name!@spark-master-0:/opt/bitnami/spark$
```

```
  GNU nano 6.2
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: spark-data-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 2Gi
  storageClassName: nfs
---
apiVersion: v1
kind: Pod
metadata:
  name: spark-data-pod
spec:
  volumes:
    - name: spark-data-pv
      persistentVolumeClaim:
        claimName: spark-data-pvc
  containers:
    - name: inspector
      image: bitnami/minideb
      command:
        - sleep
        - infinity
      volumeMounts:
        - mountPath: "/data"
          name: spark-data-pv
```

```
shaile32266@cloudshell:~ (cs570-project-426508)$ nano spark-pvc.yaml
shaile32266@cloudshell:~ (cs570-project-426508)$ kubectl apply -f spark-pvc.yaml
persistentvolumeclaim/spark-data-pvc created
pod/spark-data-pod created
shaile32266@cloudshell:~ (cs570-project-426508)$
```

IMPLEMENTATION and TESTING

Step 2: Spark Deployment:

4. Prepare the application JAR file using the Bitnami Spark Docker image.

5. Add a test file with a line of words

```
shaile32266@cloudshell:~ (cs570-project-426508)$ docker run -v /tmp:/tmp -it bitnami/spark bash -c 'find /opt/bitnami/spark/examples/jars/ -name spark-examples* -exec cp {} /tmp/my.
jar \;'
Unable to find image 'bitnami/spark:latest' locally
latest: Pulling from bitnami/spark
6d10d4f6c38d: Pull complete
Digest: sha256:9e997d4f9fb5ed0ac3942e7438478739f0243921792b0ade4479d11fbfcd6f8a
Status: Downloaded newer image for bitnami/spark:latest
spark 22:03:12.06 INFO  ==>
spark 22:03:12.06 INFO  ==> Welcome to the Bitnami spark container
spark 22:03:12.06 INFO  ==> Subscribe to project updates by watching https://github.com/bitnami/containers
spark 22:03:12.06 INFO  ==> Submit issues and feature requests at https://github.com/bitnami/containers/issues
spark 22:03:12.07 INFO  ==> Upgrade to Tanzu Application Catalog for production environments to access custom-configured and pre-packaged software components. Gain enhanced features
, including Software Bill of Materials (SBOM), CVE scan result reports, and VEX documents. To learn more, visit https://bitnami.com/enterprise
spark 22:03:12.07 INFO  ==>

shaile32266@cloudshell:~ (cs570-project-426508)$
```

IMPLEMENTATION and TESTING

Step 3: Data Preparation:

- Add a test file with a line of words for the Word Count test.
- Copy the JAR file and other required files to the Persistent Volume Claim (PVC).

```
shaile32266@cloudshell:~ (cs570-project-426508)$ kubectl cp /tmp/my.jar spark-data-pod:/data/my.jar
shaile32266@cloudshell:~ (cs570-project-426508)$ kubectl cp /tmp/test.txt spark-data-pod:/data/test.txt
shaile32266@cloudshell:~ (cs570-project-426508)$
```

```
shaile32266@cloudshell:~ (cs570-project-426508)$ nano /tmp/test.txt
shaile32266@cloudshell:~ (cs570-project-426508)$ cat /tmp/test.txt
how much wood could a woodpecker chuck if a woodpecker could chuck wood
shaile32266@cloudshell:~ (cs570-project-426508)$
```

IMPLEMENTATION and TESTING

Step 1: Cluster Creation:

- Command: gcloud container clusters create spark --num-nodes=1 --machine-type=e2-highmem-2 --region=asia-southeast1
- Verify the cluster creation.

```
shaile32266@cloudshell:~ (cs570-project-426508)$ gcloud container clusters list
NAME: spark
LOCATION: asia-southeast1
MASTER_VERSION: 1.29.4-gke.1043002
MASTER_IP: 34.143.154.92
MACHINE_TYPE: e2-highmem-2
NODE_VERSION: 1.29.4-gke.1043002
NUM_NODES: 3
STATUS: RUNNING
```

IMPLEMENTATION and TESTING

Step 4: Apache Spark Deployment:

- Deploy Spark on Kubernetes using the shared volume.
- Use the Bitnami Apache Spark Helm chart for deployment.
- Get the external IP of the running pod.

```
  GNU nano 6.2
service:
  type: LoadBalancer

worker:
  replicaCount: 3
  extraVolumes:
    - name: spark-data
      persistentVolumeClaim:
        claimName: spark-data-pvc
  extraVolumeMounts:
    - name: spark-data
      mountPath: /mnt/spark-data
```

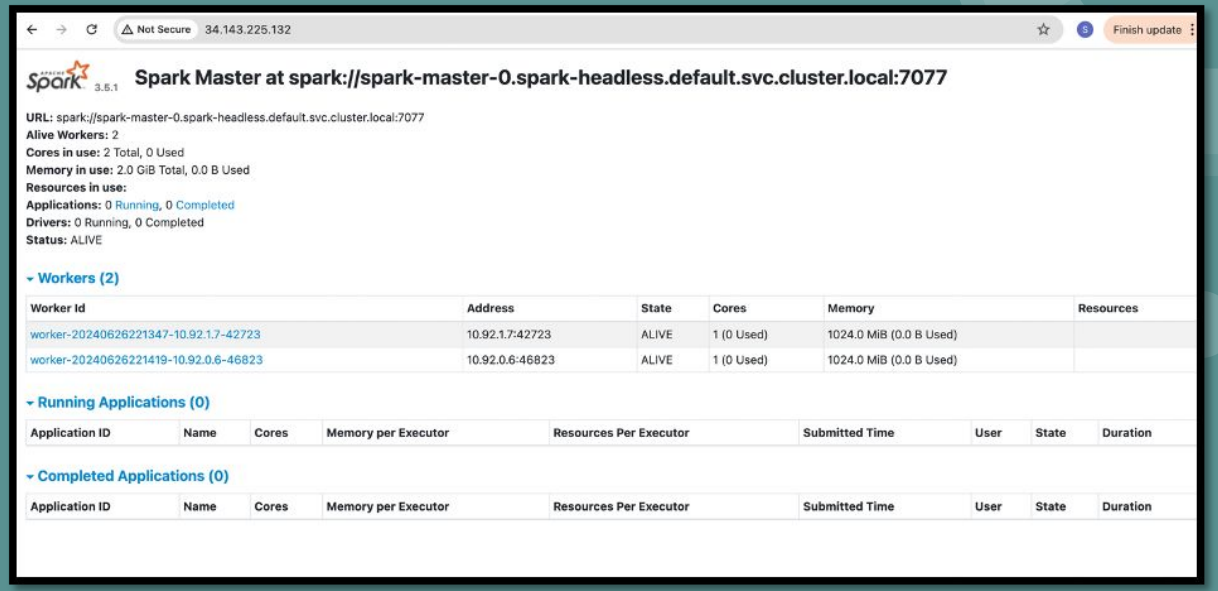
```
shaile32266@cloudshell:~ (cs570-project-426508)$ nano spark-chart.yaml
shaile32266@cloudshell:~ (cs570-project-426508)$ helm repo add bitnami https://charts.bitnami.com/bitnami
"bitnami" has been added to your repositories
shaile32266@cloudshell:~ (cs570-project-426508)$ helm install spark bitnami/spark -f spark-chart.yaml
NAME: spark
LAST DEPLOYED: Wed Jun 26 22:12:35 2024
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
CHART NAME: spark
CHART VERSION: 9.2.4
APP VERSION: 3.5.1

** Please be patient while the chart is being deployed **

1. Get the Spark master WebUI URL by running these commands:

    NOTE: It may take a few minutes for the LoadBalancer IP to be available.
    You can watch the status of by running 'kubectl get --namespace default svc -w spark-master-svc'

  export SERVICE_IP=$(kubectl get --namespace default svc spark-master-svc -o jsonpath="{.status.loadBalancer.ingress[0]['ip', 'hostname'] }")
  echo http://$SERVICE_IP:80

2. Submit an application to the cluster:

  To submit an application to the cluster the spark-submit script must be used. That script can be
  obtained at https://github.com/apache/spark/tree/master/bin. Also you can use kubectl run.

  Run the commands below to obtain the master IP and submit your application.

  export EXAMPLE_JAR=$(kubectl exec -ti --namespace default spark-worker-0 -- find examples/jars/ -name 'spark-example*\.jar' | tr -d '\r')
  export SUBMIT_IP=$(kubectl get --namespace default svc spark-master-svc -o jsonpath="{.status.loadBalancer.ingress[0]['ip', 'hostname'] }")

  kubectl run --namespace default spark-client --rm --tty -i --restart='Never' \
    --image docker.io/bitnami/spark:3.5.1-debian-12-r7 \
```

```
shaile32266@cloudshell:~ (cs570-project-426508)$ kubectl get svc -l "app.kubernetes.io/instance=spark,app.kubernetes.io/name=spark"
NAME                TYPE           CLUSTER-IP       EXTERNAL-IP       PORT(S)                        AGE
spark-headless      ClusterIP      None             <none>            <none>                         67s
spark-master-svc    LoadBalancer   34.118.233.188   34.143.225.132    7077:31235/TCP,80:30572/TCP    67s
shaile32266@cloudshell:~ (cs570-project-426508)$
```

IMPLEMENTATION and TESTING

Step 5: Apache Spark Deployment:

- Deploy Spark on Kubernetes using the shared volume.
- Use the Bitnami Apache Spark Helm chart for deployment.
- Get the external IP of the running pod



IMPLEMENTATION and TESTING

Step 5: Word Count Execution:

- Command: kubectl run --namespace default spark-client --rm --tty -i --restart='Never' --image docker.io/bitnami/spark:3.0.1-debian-10-r115 -- spark-submit --master spark://[EXTERNAL_IP]:7077 --deploy-mode cluster --class org.apache.spark.examples.JavaWordCount /data/my.jar /data/test.txt
- View the output on the browser.
- Identify the worker node IP address and execute the pod to see the result.

```
shaile32266@cloudshell:~ (cm570-project-426508)$ kubectl run --namespace default spark-client --rm --tty -i --restart='Never' \
  --image docker.io/bitnami/spark:3.0.1-debian-10-r115 \
  -- spark-submit --master spark://34.143.225.132:7077 \
  --deploy-mode cluster \
  --class org.apache.spark.examples.JavaWordCount \
  /data/my.jar /data/test.txt
If you don't see a command prompt, try pressing enter.
log4j:WARN No appenders could be found for logger (org.apache.hadoop.util.NativeCodeLoader).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
24/06/26 22:19:39 INFO SecurityManager: Changing view acls to: spark
24/06/26 22:19:39 INFO SecurityManager: Changing modify acls to: spark
24/06/26 22:19:39 INFO SecurityManager: Changing view acls groups to:
24/06/26 22:19:39 INFO SecurityManager: Changing modify acls groups to:
24/06/26 22:19:39 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(spark); groups with view permissions: Set(); us
ers  with modify permissions: Set(spark); groups with modify permissions: Set()
24/06/26 22:19:40 INFO Utils: Successfully started service 'driverClient' on port 46827.
24/06/26 22:19:40 INFO TransportClientFactory: Successfully created connection to /34.143.225.132:7077 after 47 ms (0 ms spent in bootstraps)
24/06/26 22:19:40 WARN TransportChannelHandler: Exception in connection from /34.143.225.132:7077
java.io.InvalidClassException: org.apache.spark.rpc.RpcEndpointRef; local class incompatible: stream classdesc serialVersionUID = -2184441956866814275, local class serialVersionUID
= -3992716321891270988
        at java.io.ObjectStreamClass.initNonProxy(ObjectStreamClass.java:699)
        at java.io.ObjectInputStream.readNonProxyDesc(ObjectInputStream.java:2003)
        at java.io.ObjectInputStream.readClassDesc(ObjectInputStream.java:1850)
        at java.io.ObjectInputStream.readNonProxyDesc(ObjectInputStream.java:2003)
        at java.io.ObjectInputStream.readClassDesc(ObjectInputStream.java:1850)
        at java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:2160)
        at java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1667)
        at java.io.ObjectInputStream.defaultReadFields(ObjectInputStream.java:2405)
        at java.io.ObjectInputStream.readSerialData(ObjectInputStream.java:2329)
        at java.io.ObjectInputStream.readOrdinaryObject(ObjectInputStream.java:2187)
        at java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1667)
        at java.io.ObjectInputStream.readObject(ObjectInputStream.java:503)
        at java.io.ObjectInputStream.readObject(ObjectInputStream.java:461)
        at org.apache.spark.serializer.JavaDeserializationStream.readObject(JavaSerializer.scala:76)
        at org.apache.spark.serializer.JavaSerializerInstance.deserialize(JavaSerializer.scala:109)
        at org.apache.spark.rpc.netty.NettyRpcEnv.$anonfun$deserialize$2(NettyRpcEnv.scala:292)
```

Spark 3.5.1 **Spark Master at spark://spark-master-0.spark-headless.default.svc.cluster.local:7077**

URL: spark://spark-master-0.spark-headless.default.svc.cluster.local:7077
Alive Workers: 3
Cores in use: 3 Total, 0 Used
Memory in use: 3.0 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 1 Completed
Status: ALIVE

Workers (3)

Worker Id	Address	State	Cores	Memory	Resources
worker-20240626221347-10.92.1.7-42723	10.92.1.7:42723	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	
worker-20240626221419-10.92.0.6-46823	10.92.0.6:46823	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	
worker-20240626221541-10.92.2.11-35689	10.92.2.11:35689	ALIVE	1 (0 Used)	1024.0 MiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Running Drivers (0)

Submission ID	Submitted Time	Worker	State	Cores	Memory	Resources	Main Class	Duration

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Completed Drivers (1)

Submission ID	Submitted Time	Worker	State	Cores	Memory	Resources	Main Class
driver-20240626221940-0000	2024/06/26 22:19:40	worker-20240626221541-10.92.2.11-35689	ERROR	1	1024.0 MiB		org.apache.spark.examples.JavaWordCount

IMPLEMENTATION and TESTING

Step 6: PageRank Execution:

Execute the Spark master pods.

Navigate to the directory containing pagerank.py.

Run the PageRank script: $ spark-submit pagerank.py /opt 2

```
I have no name!@spark-worker-1:/opt/bitnami/spark$ cd /opt/bitnami/spar
k/work
I have no name!@spark-worker-1:/opt/bitnami/spark/work$ cat driver-2024
0626221940-0000/stdout
```

```
        at java.lang.Thread.run(Thread.java:748)
24/06/26 22:19:40 ERROR ClientEndpoint: Error connecting to master (34.143.225.132:7077).
24/06/26 22:19:40 ERROR ClientEndpoint: Cause was: java.io.InvalidClassException: org.apache.spark.rpc.RpcEndpointRef; local class incompatible: stream classc
-2184441956866814275, local class serialVersionUID = -3992716321891270988
24/06/26 22:19:40 ERROR ClientEndpoint: No master is available, exiting.
24/06/26 22:19:40 INFO ShutdownHookManager: Shutdown hook called
24/06/26 22:19:40 INFO ShutdownHookManager: Deleting directory /tmp/spark-f3885376-34e2-4dd5-b762-749137305a77
pod "spark-client" deleted
pod default/spark-client terminated (Error)
shaile32266@cloudshell:~ (cs570-project-426508)$ kubectl get pods -o wide | grep 10.92.2.11
spark-worker-2                1/1     Running   0          15m   10.92.2.11   gke-spark-default-pool-6b06556e-1gq1   <none>           <none>
shaile32266@cloudshell:~ (cs570-project-426508)$
```

```
shaile32266@cloudshell:~ (cs570-project-426508)$ kubectl exec -it spark-worker-1 -- bash
I have no name!@spark-worker-1:/opt/bitnami/spark$
```

IMPLEMENTATION and TESTING

Step 6: PageRank Execution:

- Execute the Spark master pods.
- Navigate to the directory containing pagerank.py.
- Run the PageRank script: $ spark-submit pagerank.py /opt 2

```
shaile32266@cloudshell:/opt/bitnami/spark/examples/src/main/python (cs570-project-426508)$ spark-submit pagerank.py /opt 2
```

```
        at org.apache.spark.sql.execution.datasources.DataSource.getOrInferFileFormatSchema(DataSource.scala:167)
        at org.apache.spark.sql.execution.datasources.DataSource.resolveRelation(DataSource.scala:418)
        at org.apache.spark.sql.DataFrameReader.loadV1Source(DataFrameReader.scala:326)
        at org.apache.spark.sql.DataFrameReader.$anonfun$load$3(DataFrameReader.scala:308)
        at scala.Option.getOrElse(Option.scala:189)
        at org.apache.spark.sql.DataFrameReader.load(DataFrameReader.scala:308)
        at org.apache.spark.sql.DataFrameReader.text(DataFrameReader.scala:945)
        at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
        at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
        at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
        at java.lang.reflect.Method.invoke(Method.java:498)
        at py4j.reflection.MethodInvoker.invoke(MethodInvoker.java:244)
        at py4j.reflection.ReflectionEngine.invoke(ReflectionEngine.java:357)
        at py4j.Gateway.invoke(Gateway.java:282)
        at py4j.commands.AbstractCommand.invokeMethod(AbstractCommand.java:132)
        at py4j.commands.CallCommand.execute(CallCommand.java:79)
        at py4j.GatewayConnection.run(GatewayConnection.java:238)
        at java.lang.Thread.run(Thread.java:748)
24/06/25 20:52:44 INFO SparkContext: Invoking stop() from shutdown hook
24/06/25 20:52:44 INFO SparkUI: Stopped Spark web UI at http://spark-master-0.spark-headless.default.svc.cluster.local:4040
24/06/25 20:52:44 INFO MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
24/06/25 20:52:44 INFO MemoryStore: MemoryStore cleared
24/06/25 20:52:44 INFO BlockManager: BlockManager stopped
24/06/25 20:52:44 INFO BlockManagerMaster: BlockManagerMaster stopped
24/06/25 20:52:44 INFO OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
24/06/25 20:52:44 INFO SparkContext: Successfully stopped SparkContext
24/06/25 20:52:44 INFO ShutdownHookManager: Shutdown hook called
24/06/25 20:52:44 INFO ShutdownHookManager: Deleting directory /tmp/spark-65e26f5e-996e-40ab-a11e-2f753a90b940
24/06/25 20:52:44 INFO ShutdownHookManager: Deleting directory /tmp/spark-65e26f5e-996e-40ab-a11e-2f753a90b940/pyspark-16725e8a-5068-4bdc-af9d-016c14d
24/06/25 20:52:44 INFO ShutdownHookManager: Deleting directory /tmp/spark-df67562d-97af-46a9-99df-30c6fd7da65e3
```

GITHUB LINK

https://github.com/Sharon20222/Cloud-Computing/tree/main/Kubernetes/Pyspark

ENHANCEMENT

- **Scalability:**
  - Increase the number of nodes in the Kubernetes cluster for handling larger datasets.
  - Optimize resource allocation for better performance.
- **Automation:**
  - Automate the deployment and execution process using CI/CD pipelines.
  - Implement monitoring and alerting for job status and cluster health.
- **Algorithm Improvement:**
  - Experiment with different algorithms for data processing tasks.
  - Enhance the existing Word Count and PageRank implementations for better efficiency.

CONCLUSION

- **Cloud Infrastructure:**
  - Gained hands-on experience with setting up and managing Kubernetes clusters on GKE.
  - Learned the importance of persistent storage and volume management in Kubernetes.
- **Spark and PySpark:**
  - Understood the deployment of Apache Spark on Kubernetes using Helm charts.
  - Improved skills in developing and executing PySpark scripts for data processing.
- **Troubleshooting:**
  - Overcame challenges related to networking, storage, and resource management.
  - Developed problem-solving skills through debugging and testing.

REFERENCES

https://kubernetes.io/

https://spark.apache.org/

https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html