EECE796 Database and Big Data Analytics

Project 3

Create a Hadoop Project using Maven

In this project, you will build a simple Hadoop project using Maven. (You can also use Gradle to finish this project.) Apache Maven is a build automation tool used primarily for Java projects. You can download Apache Maven here: <http://maven.apache.org/download.cgi>.

Hadoop projects can be developed in either windows or Linux operating system. You can launch your Hadoop project locally or remotely to Hadoop cluster server. For this project, you only need to setup a single node cluster on Linux ubuntu server. The development environment is shown in the following figure.

Hadoop Cluster

myHadoop

Map Reduce

Eclipse Maven

Figure 1 Hadoop Development Environment

**Use Maven to build Hadoop Environment:**

1. Build a Java project using Maven. You need to set environmental variables “Path” for Maven and JDK.
2. Import maven project into eclipse. Make sure you have installed maven plugin. Install eclipse plugin for Maven: http://www.eclipse.org/m2e/
3. Add Hadoop dependency as follows:

*This is hadoop-1.2.1 version，revise eclipse：pom.xml*

//The red colored code is the added dependencies.

<projectxmlns="http://maven.apache.org/POM/4.0.0"xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0http://maven.apache.org/maven-v4\_0\_0.xsd">

  <modelVersion>4.0.0</modelVersion>

  <groupId>org.conan.myhadoop.mr</groupId>

  <artifactId>myHadoop</artifactId>

  <packaging>jar</packaging>

  <version>1.0-SNAPSHOT</version>

  <name>myHadoop</name>

  <url>http://maven.apache.org</url>

  <dependencies>

   <dependency>

       <groupId>org.apache.hadoop</groupId>

       <artifactId>hadoop-core</artifactId>

       <version>1.2.1</version>

    </dependency>

    <dependency>

      <groupId>junit</groupId>

      <artifactId>junit</artifactId>

      <version>3.8.1</version>

      <scope>test</scope>

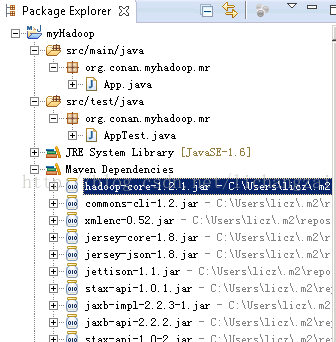
    </dependency>

  </dependencies>

</project>

1. Download the dependencies. You can see Maven dependencies as follows:

~ mvn clean install:

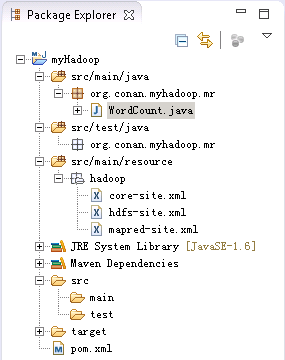


1. Download the Hadoop configuration files from cluster, and put it under src/main/resources/Hadoop directory.

core-site.xml (<value>hdfs://master :9000</value>)

hdfs-site.xml

mapred-site.xml



1. Configure localhost, and add major node master

Revise host file:

~ vi c:/Windows/System32/drivers/etc/hosts

192.168.1.210 master

1. Sample MapReduce program: WordCount.java.

*WordCount.java*

**package** org.conan.myhadoop.mr;

**import** java.io.IOException;

**import** java.util.Iterator;

**import** java.util.StringTokenizer;

**import** org.apache.hadoop.fs.Path;

**import** org.apache.hadoop.io.IntWritable;

**import** org.apache.hadoop.io.Text;

**import** org.apache.hadoop.mapred.FileInputFormat;

**import**org.apache.hadoop.mapred.FileOutputFormat;

**import**org.apache.hadoop.mapred.JobClient;

**import** org.apache.hadoop.mapred.JobConf;

**import** org.apache.hadoop.mapred.MapReduceBase;

**import** org.apache.hadoop.mapred.Mapper;

**import** org.apache.hadoop.mapred.OutputCollector;

**import** org.apache.hadoop.mapred.Reducer;

**import** org.apache.hadoop.mapred.Reporter;

**import** org.apache.hadoop.mapred.TextInputFormat;

**import** org.apache.hadoop.mapred.TextOutputFormat;

**public class** WordCount {

**public static class** WordCountMapper **extends** MapReduceBase **implements**Mapper<Object, Text, Text, IntWritable> {

**private final static** IntWritable one =**new** IntWritable(1);

**private** Textword =**new** Text();

        @Override

**public void** map(Object key, Text value,OutputCollector<Text, IntWritable> output, Reporter reporter)**throws**IOException {

            StringTokenizer itr = **new**StringTokenizer(value.toString());

**while**(itr.hasMoreTokens()) {

                word.set(itr.nextToken());

                output.collect(word,one);

            }

        }

    }

**public static class** WordCountReducer **extends** MapReduceBase **implements**Reducer<Text, IntWritable, Text, IntWritable> {

**private** IntWritableresult =**new** IntWritable();

        @Override

**public void** reduce(Text key, Iterator<IntWritable>values, OutputCollector<Text, IntWritable> output, Reporter reporter)**throws** IOException {

**int** sum = 0;

**while** (values.hasNext()){

                sum +=values.next().get();

            }

            result.set(sum);

            output.collect(key, result);

        }

    }

**public static void** main(String[] args)**throws** Exception {

        String input = "hdfs://10.1.32.91:9000/user/licz/hdfs/o\_t\_account/test.txt";

        String output = "hdfs:// 10.1.32.91:9000/user/licz/hdfs/o\_t\_account/result";

        JobConf conf = **new** JobConf(WordCount.**class**);

        conf.setJobName("WordCount");

        conf.addResource("classpath:/hadoop/core-site.xml");

        conf.addResource("classpath:/hadoop/hdfs-site.xml");

        conf.addResource("classpath:/hadoop/mapred-site.xml");

        conf.setOutputKeyClass(Text.**class**);

       conf.setOutputValueClass(IntWritable.**class**);

       conf.setMapperClass(WordCountMapper.**class**);

       conf.setCombinerClass(WordCountReducer.**class**);

       conf.setReducerClass(WordCountReducer.**class**);

        conf.setInputFormat(TextInputFormat.**class**);

       conf.setOutputFormat(TextOutputFormat.**class**);

        FileInputFormat.setInputPaths(conf,**new** Path(input));

        FileOutputFormat.setOutputPath(conf,**new** Path(output));

        JobClient.runJob(conf);

        System.exit(0);

    }

}

Reference

1. How to setup a single node cluster on Ubuntu 12.04.2 LTS 64bit Server. https://hadoop.apache.org/docs/current2/hadoop-project-dist/hadoop-common/SingleCluster.html
2. You can download the template of the code from GitHub: <https://github.com/bsspirit/maven_hadoop_template>