[DEV-301]: HDF Developer: Java

Length : 4 Days
Delivery Method : Instructor-led (Classroom)

Course Overview

This course provides Java programmers a deep-dive into Hadoop application development. Participants will learn how to design and develop efficient and effective MapReduce applications for Hadoop using the Hortonworks Data Platform, including how to implement combiners, partitioners, secondary sorts, custom input and output formats, joining large datasets, unit testing, and developing UDFs for Pig and Hive. Labs are run on a 7-node HDP 2.1 cluster running in a virtual machine that participants can keep for use after the training.

Audience Profile

Experienced Java software engineers who need to develop Java MapReduce applications for Hadoop.

Pre-Requisites

Participants must have experience developing Java applications and using a Java IDE. Labs are completed using the Eclipse IDE and Gradle. No prior Hadoop knowledge is required.

Formats

Lecture/Discussion	50%
Hands-on Labs	50%

Course Outline

Module 1: Understanding Hadoop, the Hadoop Distributed File System (HDFS) and Map Reduce

Lessons

- Describe Hadoop 2.X and the Hadoop Distribute File System
- Describe the YARN framework
- Describe the Purpose of NameNodes and Data Nodes
- Describe the Purpose of HDFS High Availability (HA)
- Describe the Purpose of the Quorum Journal Manager
- List Common HDFS Commands
- Describe the Purpose of YARN
- List Open-Source YARN Use Cases
- List the Components of YARN
- Describe the Life Cycle of a YARN Application
- Define Map Aggregation
- Describe the Purpose of Combiners
- Describe the Purpose of In-Map Aggregation

- Describe the Purpose of Counters
- Describe the Purpose of User-Defined Counters

Labs

- Demonstration: Understanding Block Storage
- Configuring a Hadoop Development Environment
- Putting Files in HDFS with Java
- Demonstration: Understanding Map Reduce
- Word Count
- Distributed Grep
- Inverted Index
- Using a Combiner
- Computing an Average

Module 2: Partitioning, Sorting and Input/Output Formats

Lessons

- Describe the Purpose of a Partitioner
- List the Steps for Writing a Custom Partitioner
- Describe How to Create and Distribute a Partition File
- Describe the Purpose of Sorting
- Describe the Purpose of Custom Keys
- Describe How to Write a Group Comparator
- List the Built-In Input Formats
- Describe the Purpose of Input Formats
- Define a Record Reader
- Describe How to Handle Records that Span Splits
- List the Built-In Output Formats
- Describe How to Write a Custom Output Format
- Describe the Purpose of the MultipleOutputs Class

LABS & DEMONSTRATIONS

- Writing a Custom Partitioner
- Using TotalOrderPartitioner
- Custom Sorting
- Demonstration: Combining Input Files
- Processing Multiple Inputs
- Writing a Custom Input Format
- Customizing Output
- Working with a Simple Moving Average

Module 3: Optimizing MapReduce Jobs, Advanced MapReduce Features and HBase Programming

Lessons

- List Optimization Best Practices
- Describe How to Optimize the Map and Reduce Phases

- Describe the Benefits of Data Compression
- Describe the Limits of Data Compression
- Describe the Configuration of Data Compression
- Describe the Purpose of a RawComparator
- Describe the Purpose of Localization
- List Scenarios for Performing Joins in MapReduce
- Describe the Purpose of the Bloom Filter
- Describe the Purpose of MRUnit and the MRUnit API
- Describe How to Set Up a Test
- Describe How to Test a Mapper
- Describe How to Test a Reducer
- Describe the Purpose of HBase
- Define the Differences Between a Relational Database and HBase
- Describe the HBase Architecture
- Demonstrate the Basics of HBase Programming
- Describe an HBase MapReduce Applications

LABS

- Using Data Compression
- Defining a RawComparator
- Performing a Map-Side Join
- Using a Bloom Filter
- Unit Testing a MapReduce Job
- Importing Data to HBase
- Creating an HBase Mapreduce Job

Module 4: Pig and Hive Programming, Defining Workflows
Lessons

- Describe the Purpose of Apache Pig and Pig Latin
- Demonstrate the Use of the Grunt Shell
- List the Common Pig Data Types
- Describe the Purpose of the FOREACH GENERATE Operator
- Describe the Purpose of Pig User Defined Functions (UDFs)
- Describe the Purpose of Filter Functions
- Describe the Purpose of Accumulator UDFs
- Describe the Purpose of Algebraic Functions
- Describe the Purpose of Apache Hive
- Describe the Differences Between Apache Hive and SQL
- Describe Apache Hive Architecture
- Describe How to Load Data Into Hive
- Demonstrate How to Perform Queries
- Describe the Purpose of Hive User Defined Functions (UDFs)
- Write a Hive UDF
- Describe the Purpose of HCatalog

- Describe the Purpose of Apache Oozie
- Describe How to Define an Oozie Workflow
- Describe Pig and Hive Actions
- Describe How to Define an Oozie Coordinator Job

LABS

- Demonstration: Understanding Pig
- Writing a Pig UDF
- Writing a Pig Accumulator
- Writing a Apache Hive UDF
- Defining an Oozie Workflow
- Working with TF-IDF and the JobControl Class