The source code for the book is organized in Eclipse projects. There is one project per Chapter, for every chapter for which there is a source code.

**Chapter 2 project**

This project contains 3 main packages:

* com.practicalHadoop.hbase.customFilter is the package containing an example of custom HBase filter.
* com.practicalHadoop.hbase.tables is the package containing an example of programmatic HBase table creation.
* com.practicalHadoop.hbase.tables.configuration is the support package containing JAXB generated classes (from HBASETables.xsd) for HBase definition.

Both examples (filter and table creation) assumes existence of the available Hadoop cluster with HBase installed.

(Provided pom file is for CDH3 Clodera distribution - Hadoop 1. For different Hadoop distribution you will need to change pom file)

In order to run a filter example, filter has to be installed on the cluster. See http://comments.gmane.org/gmane.comp.java.hadoop.hbase.user/27131

for approaches to install custom filter.

Once the filter is installed on Hadoop cluster it can be tested using BoundinBoxFilterExample class.

This class can either run from edge node using Hadoop command or from Eclipse.

In order to run it from the edge node please comment out the line: Configuration.addDefaultResource("SandBox\_Cloud\_Config.xml"); rebuid the project and run the jar file using Hadoop command (see Chapter 3 for more details).

In order to run this class from Eclipse: In the file SandBox\_Cloud\_Config.xml replace your quorum and your port with the appropriate values and remove provided tag from pom defining Hadoop libraries.

Programmatic table creation can be tested using TableCreator class with an xml definition of tables as input.

This class can either run from edge node using Hadoop command or from Eclipse.

In order to run it from the edge node please comment out the line: Configuration.addDefaultResource("SandBox\_Cloud\_Config.xml");rebuid the project and run the jar file using Hadoop command (see Chapter 3 for more details).

In order to run this class from Eclipse: In the file SandBox\_Cloud\_Config.xml replace your quorum and your port with the appropriate values and remove provided tag from pom defining Hadoop libraries

**Chapter 4 project**

This project contains 11 main packages:

* com.practicalHadoop.inputformat.ComputeIntensive is the package containing inputFormats for compute intensive applications.
* com.practicalHadoop.inputformat.multifile is the package containing inputFormats for multi file processing.
* com.practicalHadoop.inputformat.multitable is the package containing inputFormats for multi table processing.
* com.practicalHadoop.inputformat.queue is the package containing inputFormats for for multi file processing using queue.
* com.practicalHadoop.nativecode is the package containing support for native code loading.
* com.practicalHadoop.outputformat.MultpleDirectories is the package containing outputFormats for writing data to multiple directories.
* com.practicalHadoop.partitioner is the package containing implementation of the partitioner.
* com.practicalHadoop.queue is the package containing implementation of Queue based on HBase.
* com.practicalHadoop.reader.queue is the package containing implementation of Queue Reader.
* com.practicalHadoop.reader.xml is the package containing implementation of XML Reader.
* com.practicalHadoop.writer.tar is the package containing implementation of tar writer.

The majority of code in this project is not meant to be executed by itself, but rather as part of the larger project, leveraging

some of the provided functionality.

So the project builds a jar file which can be used by any project leveraging the functionality.

Two classes provided in this project have main methods and can be implemented for testing:

* Executor class in com.practicalHadoop.outputformat.MultpleDirectories package allows to test outputFormats for writing data to multiple directories. In order to run this test you need to move resultig jar to the edge node and run it with Hadoop command (see Chapter 3 for more details)
* XMLReader class in com.practicalHadoop.reader.xml package allows to test XML processing. This class can either run from edge node using Hadoop command or from Eclipse. In order to run it from the edge node please run the jar file using Hadoop command (see Chapter 3 for more details).

In order to run this class from Eclipse: Remove provided tag from pom defining Hadoop libraries

**Chapter 5 project**

This project contains 1 package:

* com.practicalHadoop.log is the package containing a Hadoop log scraper.

The project assumes existence of the available Hadoop cluster. It was tested on Hadoop 1 cluster (Cloudera 3.2)

To invoke HadoopJobLogScraper specify Job Tracker URL andjob id

**Chapter 7 project**

This project contains all the code, configuration and bash files described in Chapter 7. See Chapter itself for details on how to use the code

The actual java code is split between following 4 packages

* com.practicalHadoop.geotile, com.practicalHadoop.places and com.practicalHadoop.strand packages contains implementations (place holders) of java activities
* com.practicalHadoop.util package contains implementation (place holder) of java activity DetermineError

Cluster directory contains

* implementations of workflows, coordinators and bundle jobs (.xml files)
* implementations of the Pig activities (.pig files)
* implementations of the Hive activities (.hql files)
* Scripts for running jobs on the cluster (.bash files)

The project assumes existence of the available Hadoop cluster with the latest (3.x) version of Oozie installed. It was tested on Hadoop 2 cluster (Cloudera 4.1)

**Chapter 8 project**

This project contains 9 main packages:

* com.practicalHadoop.javaApi is the package containing an implementation of Oozie java APIs.
* com.practicalHadoop.oozie.ftp is the package containing an implementation of support classes for FTP custom Oozie node.
* com.practicalHadoop.oozieEmail is the package containing JAXB classes for Oozie email node.
* com.practicalHadoop.oozieext is the package containing an implementation of FTP custom Oozie node.
* com.practicalHadoop.ooziewf is the package containing JAXB classes for Oozie workflow.
* com.practicalHadoop.ooziewf.builder is the package containing implementation of dynamic Oozie workflow builder.
* com.practicalHadoop.uber is the package containing implementation of the uber jar support for Oozie action node.
* com.practicalHadoop.uber is the package containing implementation of the uber jar support for Oozie action node.
* com.practicalHadoop.uber.util is the package containing utilities implementation for the uber jar support for Oozie action node.

An additional package net.jcip.annotations containing implementation of the class, field, and method level annotations for describing thread-safety policies.

The project assumes existence of the available Hadoop cluster with the latest (3.x) version of Oozie installed. It was tested on Hadoop 2 cluster (Cloudera 4.1)

**Chapter 9 project**

This project contains 11 main packages:

* com.practicalHadoop.lucene.cache is the package containing cache implementation for Lucene with HNase back end.
* com.practicalHadoop.lucene.document is the package containing document processing implementation for Lucene with HNase back end.
* com.practicalHadoop.lucene.indexing is the package containing index reader/writer implementation for Lucene with HNase back end.
* com.practicalHadoop.lucene.indexing.support is the package containing implementation of the index reader/writer support for Lucene with HNase back end.
* com.practicalHadoop.lucene.initializer is the package containing initializer implementation for Lucene with HNase back end.
* com.practicalHadoop.lucene.spatial is the package containing spatial support implementation for Lucene with HNase back end.
* com.practicalHadoop.lucene.spatial.geometry is the package containing geometry calculations for spatial support implementation for Lucene with HNase back end.
* com.practicalHadoop.lucene.spatial.geometry.filters is the package containing geometry filters for spatial support implementation for Lucene with HNase back end.
* com.practicalHadoop.lucene.tables is the package containing HBase tables operations implementation for Lucene with HNase back end.
* com.practicalHadoop.lucene.tables.configuration is the package containing HBase tables configurations implementation for Lucene with HNase back end.
* com.practicalHadoop.photo is the package containing photo management system implementation.

Additionally, com.practicalHadoop.lucene package in the test directory contains several unit tests for Lucene with HNase back end.

The project assumes existence of the available Hadoop cluster with HBase installed. It was tested on Hadoop 2 cluster (Cloudera 4.1)

**Chapter 11 project**

This project contains 7 main packages:

* com.practicalHadoop.aws is the package containing support classes used by all AWS implementations.
* com.practicalHadoop.emr is the package containing an example implementation of thfe EMR commands.
* com.practicalHadoop.emr.builders is the package containing builder classes for an example implementation of the EMR commands.
* com.practicalHadoop.emr.utils is the package containing utility classes for an example implementation of the EMR commands.
* com.practicalHadoop.s3client is the package containing an example implementation of the S3 client.
* com.practicalHadoop.s3client.utils is the package containing utility classes for an example implementation of the S3 client.
* com.practicalHadoop.s3copytool is the package containing an example implementation of the S3 copy tool. This implementation relies on Chapter 4 project.

In order to run code in this package, make sure that you have a valid AWS account and store you access and secret keys in the AwsCredentials.properties file.

This project provides support for the following:

JobStatus and JobKill classes in com.practicalHadoop.emr package allow to get status and kill EMR jobs respectfully. Additionally JobInvoker class provides all of the functionality for creation of the EMR cluster, defining execution steps and executing them. This class can be used by implementation of AWS EMR command implementation.

GenericS3ClientImpl is the workhorse, implementing all S3 access operations. It can be used by any applicationrequiring S3 access.

S3DistCP is the driver for a MapReduce implementation supporting copying a set of HDFS files to S3. See S3DistCP class for command line instructions