HBase
Java Administrative API

Originals of slides and source code for examples: http://www.coreservlets.com/hadoop-tutorial/
Also see the customized Hadoop training courses (onsite or at public venues) – http://courses.coreservlets.com/hadoop-training.html

Customized Java EE Training: http://courses.coreservlets.com/
Hadoop, Java, JSF 2, PrimeFaces, Servlets, JSP, Ajax, jQuery, Spring, Hibernate, RESTful Web Services, Android.
Developed and taught by well-known author and developer. At public venues or onsite at your location.

For live customized Hadoop training (including prep for the Cloudera certification exam), please email info@coreservlets.com
Taught by recognized Hadoop expert who spoke on Hadoop several times at JavaOne, and who uses Hadoop daily in real-world apps. Available at public venues, or customized versions can be held on-site at your organization.

Courses developed and taught by Marty Hall
- JSF 2.2, PrimeFaces, servlets/JSP, Ajax, jQuery
- Android development, Java 7 or 8 programming, custom mix of topics
- Courses available in any state or country. Maryland/DC area companies can also choose afternoon/evening courses.

Courses developed and taught by coreservlets.com experts (edited by Marty)
- Spring, Hibernate/JPA, GWT, Hadoop, HTML5, RESTful Web Services

Contact info@coreservlets.com for details
Agenda

- Create Table
- Drop Table

Java Admin API

- Just like HTable is for client API, HBaseAdmin is for administrative tasks
  - org.apache.hadoop.hbase.client.HBaseAdmin
- Recall that only Table and Family names have to be pre-defined
  - Columns can be added/deleted dynamically
  - HBase scheme roughly equals table definitions and their column families
Create Table and Column Families

1. Construct HBaseAdmin instance
2. Create Table’s schema
   – Represented by HTableDescriptor class
   – Add column families to table descriptor (HColumnDescriptor)
3. Execute create via HBaseAdmin class

1: Construct HBaseAdmin Instance

- HbaseAdmin’s constructor requires an instance of Configuration object
  – Similar to HTable
  – We already know how to do that

```java
Configuration conf = HBaseConfiguration.create();
HBaseAdmin admin = new HBaseAdmin(conf);
```
2: Create Table Descriptor

- `org.apache.hadoop.hbase.HTableDescriptor`
  - Serves as a container for table name and column families
  - Most importantly, add one or more column families
- `org.apache.hadoop.hbase.HColumnDescriptor`
- `HColumnDescriptor` serves as a container for column family name, compressions settings, number of versions, in-memory setting, and block size

```java
HTableDescriptor table = new HTableDescriptor(toBytes("Table"));
HColumnDescriptor family = new HColumnDescriptor(toBytes("f"));
table.addFamily(family);
HColumnDescriptor family1 = new HColumnDescriptor(toBytes("f1"));
table.addFamily(family);
```

3: Execute Create via HBaseAdmin

- `HBaseAdmin` creates a table via `createTable` method
  - Synchronous operation

```java
admin.createTable(table);
```
public static void main(String[] args) throws IOException {
    Configuration conf = HBaseConfiguration.create();
    HBaseAdmin admin = new HBaseAdmin(conf);

    String name = "NewTable";
    byte[] tableName = toBytes(name);
    HTableDescriptor table = new HTableDescriptor(tableName);
    HColumnDescriptor family = new HColumnDescriptor(toBytes("new_family"));
    table.addFamily(family);

    System.out.println("Table "+name+" exist: " +
                       admin.tableExists(tableName)) ;
    System.out.println("Creating "+name+" table...");
    admin.createTable(table);
    System.out.println("Table "+name+" exist: " +
                       admin.tableExists(tableName)) ;
}

$ yarn jar $PLAY_AREA/HadoopSamples.jar hbase.CreateTableExample...
Table NewTable exist: false
Creating NewTable table...
Table NewTable exist: true

$ hbase shell
hbase> describe 'NewTable'
DESCRIPTION ENABLED
{NAME => 'NewTable', FAMILIES => [{NAME => 'new_family',
BLOOMFILTER => 'NONE', REPLICATION_SCOPE => '0',
COMPRESSION => 'NONE', VERSIONS => '3', TTL =>
'2147483647', BLOCKSIZE => '65536', IN_MEMORY => 'false',
BLOCKCACHE => 'true'}]}

1 row(s) in 0.0400 seconds
Drop Table

1. Construct HBaseAdmin instance
2. Disable table
   – Table must be taken offline in order to perform any schema modifications
3. Delete table

DropTableExample.java

```java
public static void main(String[] args) throws IOException {
    Configuration conf = HBaseConfiguration.create();

    HBaseAdmin admin = new HBaseAdmin(conf);
    byte[] tableName = toBytes("NewTable");

    admin.disableTable(tableName);
    admin.deleteTable(tableName);
}
```

Bytes utility class is imported with ‘static’ keyword:
```
import static org.apache.hadoop.hbase.util.Bytes.toBytes;
```
Test DropTableExample.java

```bash
$ hbase shell
hbase> exists 'NewTable'
Table NewTable does exist
0 row(s) in 0.1420 seconds
hbase> quit

$ yarn jar $PLAY_AREA/HadoopSamples.jar hbase.DropTableExample
...
12/01/16 12:21:44 INFO client.HBaseAdmin: Started disable of NewTable
12/01/16 12:21:46 INFO client.HBaseAdmin: Disabled NewTable
12/01/16 12:21:46 INFO client.HBaseAdmin: Deleted NewTable

$ hbase shell
hbase> exists 'NewTable'
Table NewTable does not exist
0 row(s) in 0.3700 seconds
```

Wrap-Up
Summary

- **We learned how to**
  - Create new table
  - Drop an existing table