[**HBase Java API类介绍**](file:///F:\Html\mannual_files_all\hbase_study\peixun_notes\hbase%E7%AC%94%E8%AE%B0\HBase%20Java%20API类介绍%20-%20ggjucheng%20-%20博客园_files\HBase%20Java%20API类介绍%20-%20ggjucheng%20-%20博客园.htm)

　几个相关类与HBase数据模型之间的对应关系

|  |  |
| --- | --- |
| java类 | HBase数据模型 |
| HBaseAdmin | 数据库（DataBase) |
| HBaseConfiguration |
| HTable | 表（Table) |
| HTableDescriptor | 列族（Column Family) |
| Put | 列修饰符（Column Qualifier） |
| Get |
| Scanner |

**一、HBaseConfiguration**

关系：org.apache.hadoop.hbase.HBaseConfiguration

作用：对HBase进行配置

|  |  |  |
| --- | --- | --- |
| 返回值 | 函数 | 描述 |
| void | addResource(Path file) | 通过给定的路径所指的文件来添加资源 |
| void | clear() | 清空所有已设置的属性 |
| string | get(String name) | 获取属性名对应的值 |
| String | getBoolean(String name, boolean defaultValue) | 获取为boolean类型的属性值，如果其属性值类型不为boolean,则返回默认属性值 |
| void | set(String name, String value) | 通过属性名来设置值 |
| void | setBoolean(String name, boolean value) | 设置boolean类型的属性值 |

 用法示例：

HBaseConfiguration hconfig = new HBaseConfiguration();

hconfig.set("hbase.zookeeper.property.clientPort","2181");

　　该方法设置了"hbase.zookeeper.property.clientPort"的端口号为2181。一般情况下，HBaseConfiguration会使用构造函数进行初始化，然后在使用其他方法。

**二、HBaseAdmin**

关系：org.apache.hadoop.hbase.client.HBaseAdmin

作用：提供了一个接口来管理HBase数据库的表信息。它提供的方法包括：创建表，删除表，列出表项，使表有效或无效，以及添加或删除表列族成员等。

|  |  |  |
| --- | --- | --- |
| 返回值 | 函数 | 描述 |
| void | addColumn(String tableName, HColumnDescriptor column) | 向一个已经存在的表添加咧 |
| checkHBaseAvailable(HBaseConfiguration conf) | 静态函数，查看HBase是否处于运行状态 |
| createTable(HTableDescriptor desc) | 创建一个表，同步操作 |
| deleteTable(byte[] tableName) | 删除一个已经存在的表 |
| enableTable(byte[] tableName) | 使表处于有效状态 |
| disableTable(byte[] tableName) | 使表处于无效状态 |
| HTableDescriptor[] | listTables() | 列出所有用户控件表项 |
| void | modifyTable(byte[] tableName, HTableDescriptor htd) | 修改表的模式，是异步的操作，可能需要花费一定的时间 |
| boolean | tableExists(String tableName) | 检查表是否存在 |

用法示例：

HBaseAdmin admin = new HBaseAdmin(config);

admin.disableTable("tablename")

**三、HTableDescriptor**

关系：org.apache.hadoop.hbase.HTableDescriptor

作用：包含了表的名字极其对应表的列族

|  |  |  |
| --- | --- | --- |
| 返回值 | 函数 | 描述 |
| void | addFamily(HColumnDescriptor) | 添加一个列族 |
| HColumnDescriptor | removeFamily(byte[] column) | 移除一个列族 |
| byte[] | getName() | 获取表的名字 |
| byte[] | getValue(byte[] key) | 获取属性的值 |
| void | setValue(String key, String value) | 设置属性的值 |

用法示例：

HTableDescriptor htd = new HTableDescriptor(table);

htd.addFamily(new HcolumnDescriptor("family"));

在上述例子中，通过一个HColumnDescriptor实例，为HTableDescriptor添加了一个列族：family

**四、HColumnDescriptor**

关系：org.apache.hadoop.hbase.HColumnDescriptor

作用：维护着关于列族的信息，例如版本号，压缩设置等。它通常在创建表或者为表添加列族的时候使用。列族被创建后不能直接修改，只能通过删除然后重新创建的方式。列族被删除的时候，列族里面的数据也会同时被删除。

|  |  |  |
| --- | --- | --- |
| 返回值 | 函数 | 描述 |
| byte[] | getName() | 获取列族的名字 |
| byte[] | getValue(byte[] key) | 获取对应的属性的值 |
| void | setValue(String key, String value) | 设置对应属性的值 |

用法示例：

HTableDescriptor htd = new HTableDescriptor(tablename);

HColumnDescriptor col = new HColumnDescriptor("content:");

htd.addFamily(col);

此例添加了一个content的列族

**五、HTable**

关系：org.apache.hadoop.hbase.client.HTable

作用：可以用来和HBase表直接通信。此方法对于更新操作来说是非线程安全的。

|  |  |  |
| --- | --- | --- |
| 返回值 | 函数 | 描述 |
| void | checkAdnPut(byte[] row, byte[] family, byte[] qualifier, byte[] value, Put put | 自动的检查row/family/qualifier是否与给定的值匹配 |
| void | close() | 释放所有的资源或挂起内部缓冲区中的更新 |
| Boolean | exists(Get get) | 检查Get实例所指定的值是否存在于HTable的列中 |
| Result | get(Get get) | 获取指定行的某些单元格所对应的值 |
| byte[][] | getEndKeys() | 获取当前一打开的表每个区域的结束键值 |
| ResultScanner | getScanner(byte[] family) | 获取当前给定列族的scanner实例 |
| HTableDescriptor | getTableDescriptor() | 获取当前表的HTableDescriptor实例 |
| byte[] | getTableName() | 获取表名 |
| static boolean | isTableEnabled(HBaseConfiguration conf, String tableName) | 检查表是否有效 |
| void | put(Put put) | 向表中添加值 |

用法示例：

HTable table = new HTable(conf, Bytes.toBytes(tablename));

ResultScanner scanner = table.getScanner(family);

**六、Put**

关系：org.apache.hadoop.hbase.client.Put

作用：用来对单个行执行添加操作

|  |  |  |
| --- | --- | --- |
| 返回值 | 函数 | 描述 |
| Put | add(byte[] family, byte[] qualifier, byte[] value) | 将指定的列和对应的值添加到Put实例中 |
| Put | add(byte[] family, byte[] qualifier, long ts, byte[] value) | 将指定的列和对应的值及时间戳添加到Put实例中 |
| byte[] | getRow() | 获取Put实例的行 |
| RowLock | getRowLock() | 获取Put实例的行锁 |
| long | getTimeStamp() | 获取Put实例的时间戳 |
| boolean | isEmpty() | 检查familyMap是否为空 |
| Put | setTimeStamp(long timeStamp) | 设置Put实例的时间戳 |

用法示例：

HTable table = new HTable(conf,Bytes.toBytes(tablename));

Put p = new Put(brow);//为指定行创建一个Put操作

p.add(family,qualifier,value);

table.put(p);

**七、Get**

关系：org.apache.hadoop.hbase.client.Get

作用：用来获取单个行的相关信息

|  |  |  |
| --- | --- | --- |
| 返回值 | 函数 | 描述 |
| Get | addColumn(byte[] family, byte[] qualifier) | 获取指定列族和列修饰符对应的列 |
| Get | addFamily(byte[] family) | 通过指定的列族获取其对应列的所有列 |
| Get | setTimeRange(long minStamp,long maxStamp) | 获取指定取件的列的版本号 |
| Get | setFilter(Filter filter) | 当执行Get操作时设置服务器端的过滤器 |

用法示例：

HTable table = new HTable(conf, Bytes.toBytes(tablename));

Get g = new Get(Bytes.toBytes(row));

**八、Result**

关系：org.apache.hadoop.hbase.client.Result

作用：存储Get或者Scan操作后获取表的单行值。使用此类提供的方法可以直接获取值或者各种Map结构（key-value对）

|  |  |  |
| --- | --- | --- |
| 返回值 | 函数 | 描述 |
| boolean | containsColumn(byte[] family, byte[] qualifier) | 检查指定的列是否存在 |
| NavigableMap<byte[],byte[]> | getFamilyMap(byte[] family) | 获取对应列族所包含的修饰符与值的键值对 |
| byte[] | getValue(byte[] family, byte[] qualifier) | 获取对应列的最新值 |

**九、ResultScanner**

关系：Interface

作用：客户端获取值的接口

|  |  |  |
| --- | --- | --- |
| 返回值 | 函数 | 描述 |
| void | close() | 关闭scanner并释放分配给它的资源 |
| Result | next() | 获取下一行的值 |

转自 <http://www.cnblogs.com/NicholasLee/archive/2012/09/13/2683432.html>