Group 7: James Clark, Anthony Sommer, Saitejasree Ramala, and James Wehmueller
Dr. Yugyung Lee
CS 590BD: Big Data Analytics
June 24, 2014

Lab 2

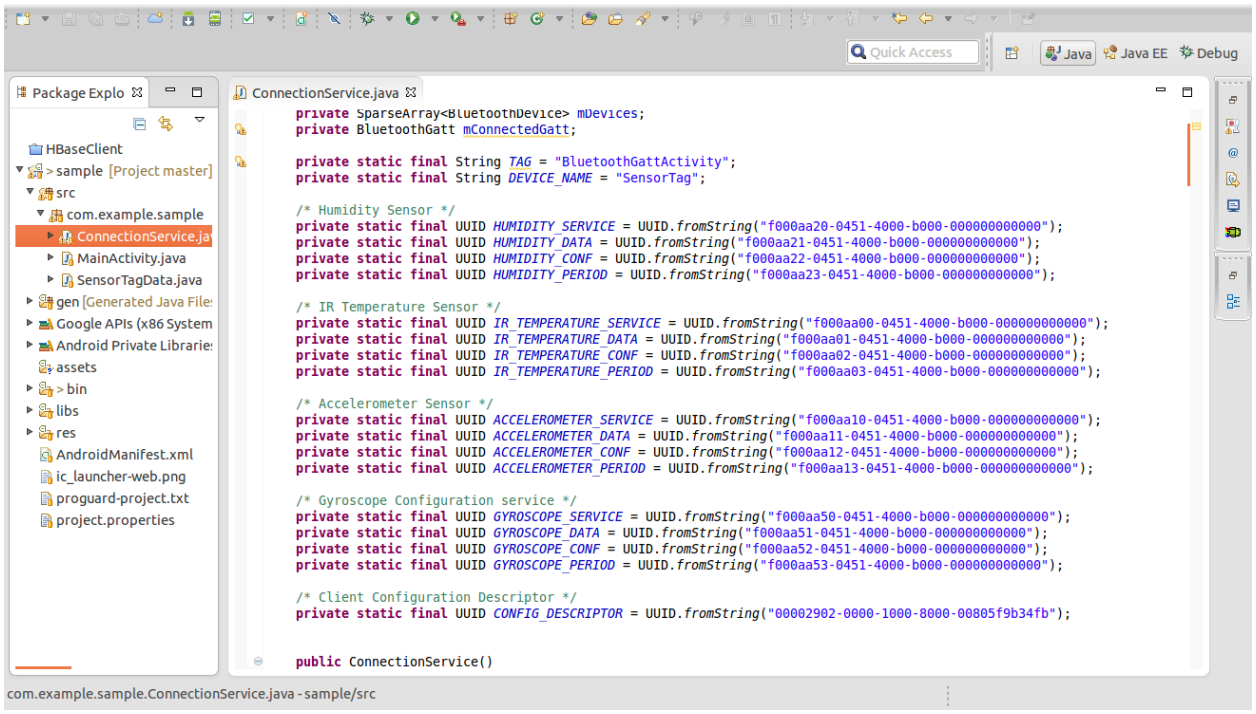
Part 1 - Android + Sensors to Text FIle

We modified the app1-app3-SensorTagGPS project sample code from Tutorial 3 to accomplish Part 1: File generation for sensor activity of at least four types of information using the TI CC2541 SensorTag.
Our application uses the IR Temperature, Humidity, Accelerometer, and Gyroscope sensors to record ambient temperature, object temperature, relative humidity, proper acceleration (x, y, z), and orientation (x, y, z) at the default one second interval for all sensors.

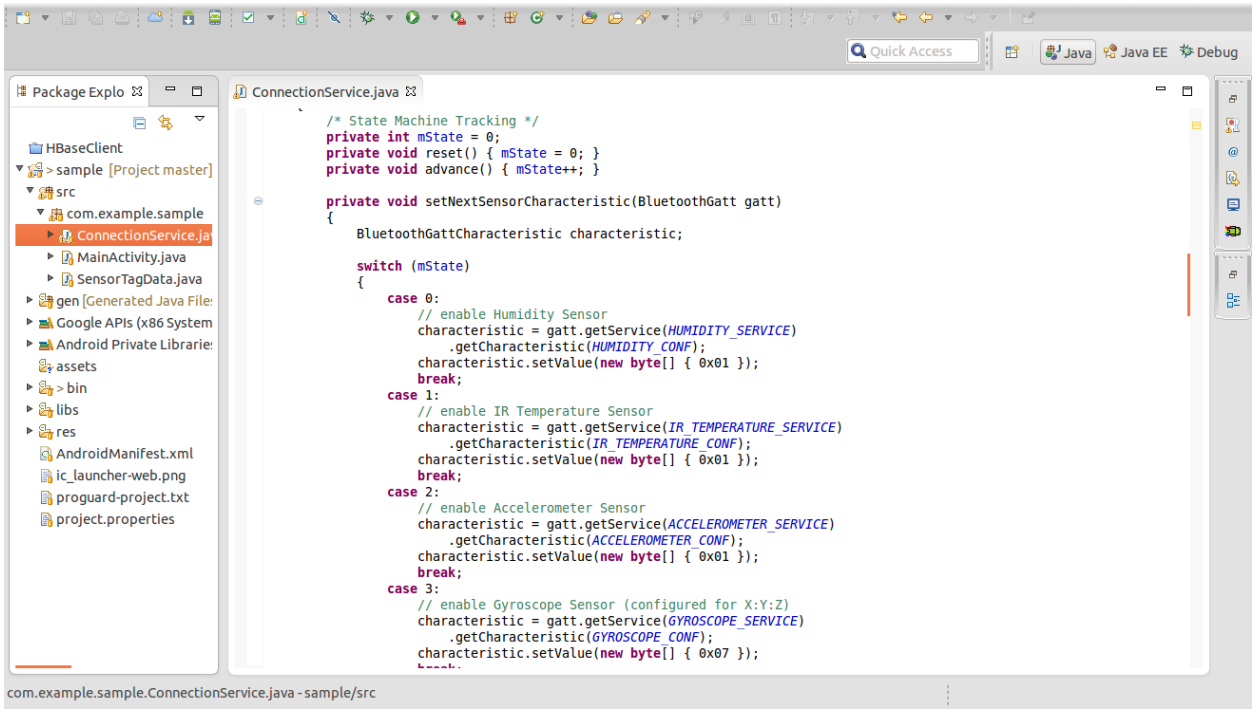
Sensors are enabled and the GATT server is set to listen for data updates for each using the GATT callback methods and a counting variable to iterate through each sensor:

1. The onServicesDiscovered() callback method resets the counter and calls setNextSensorCharacteristic().
2. setNextSensorCharacteristic() enables a sensor and invokes the onCharacteristicWrite() callback.
3. The onCharacteristicWrite() callback method calls enableNextSensorNotification().
4. enableNextSensorNotification() subscribes to data notifications for a sensor and invokes the onDescriptorWrite() callback.
5. The onDescriptorWrite() callback method advances the counter and calls setNextSensorCharacteristic() for the next sensor.

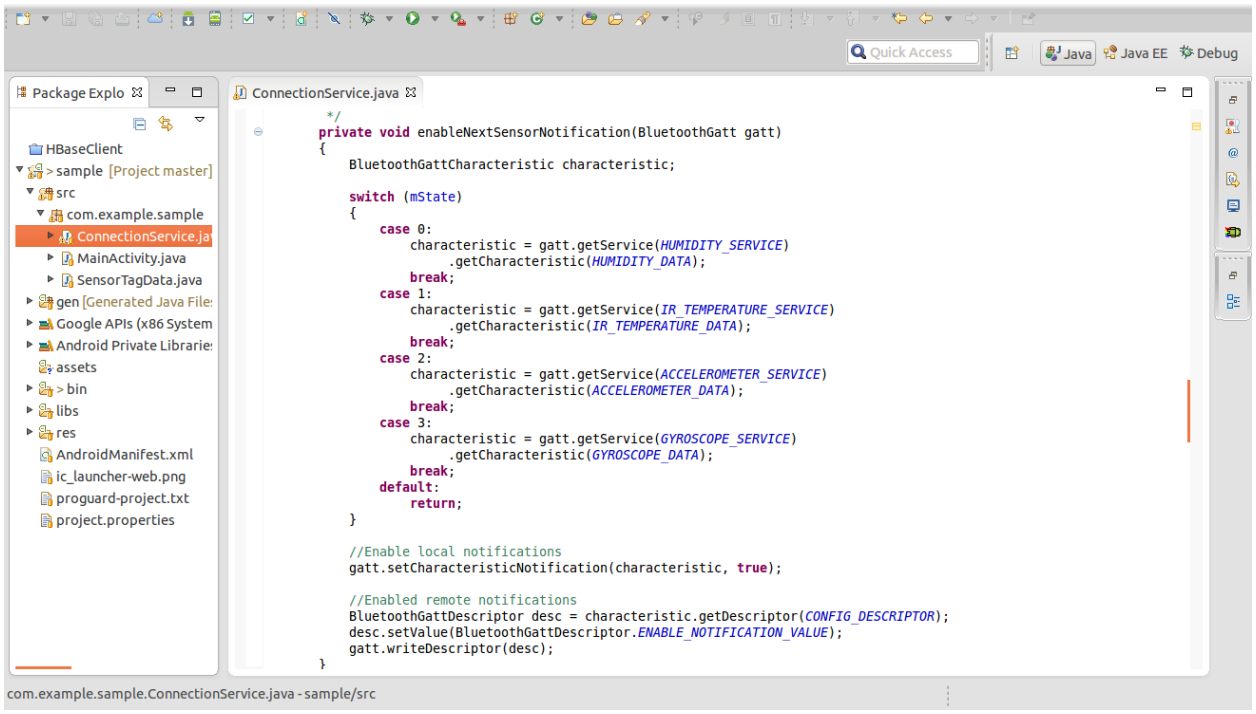
Inclusion of UUIDs for sensors:



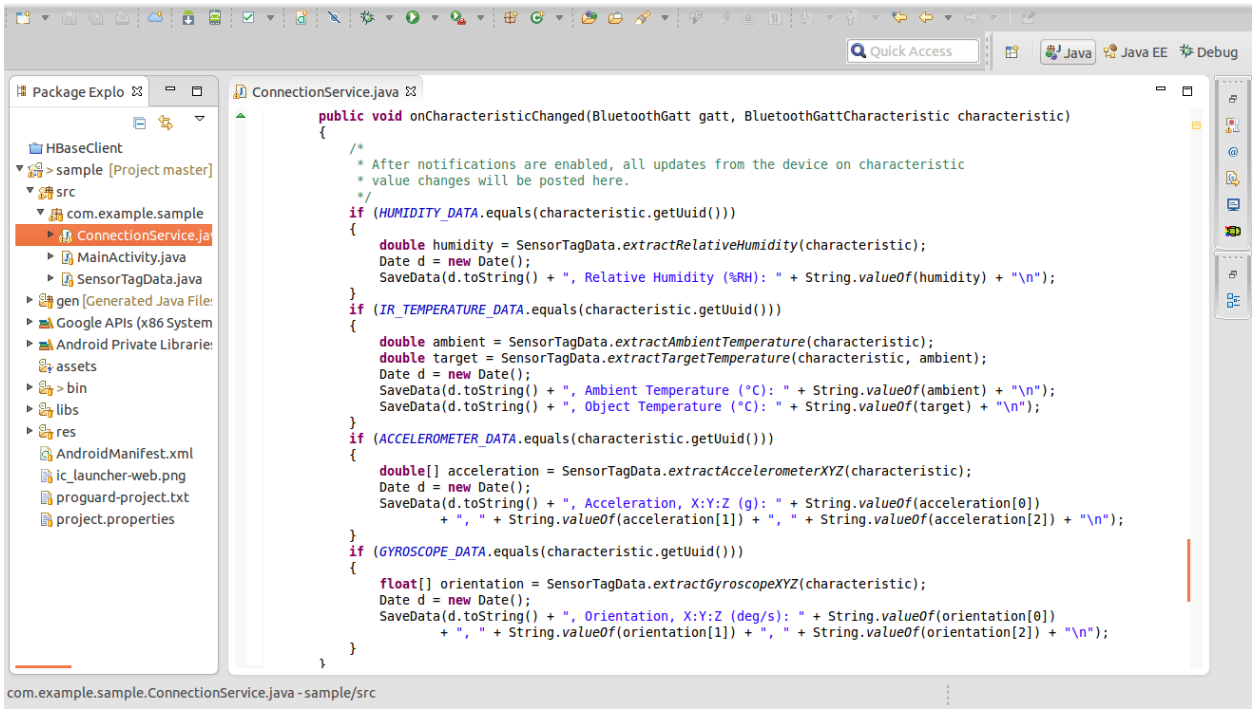
setNextSensorCharacteristic(): sensors are enabled by writing 0x01 to the Configuration (Gyroscope uses 0x07 for enabling x, y, and z)



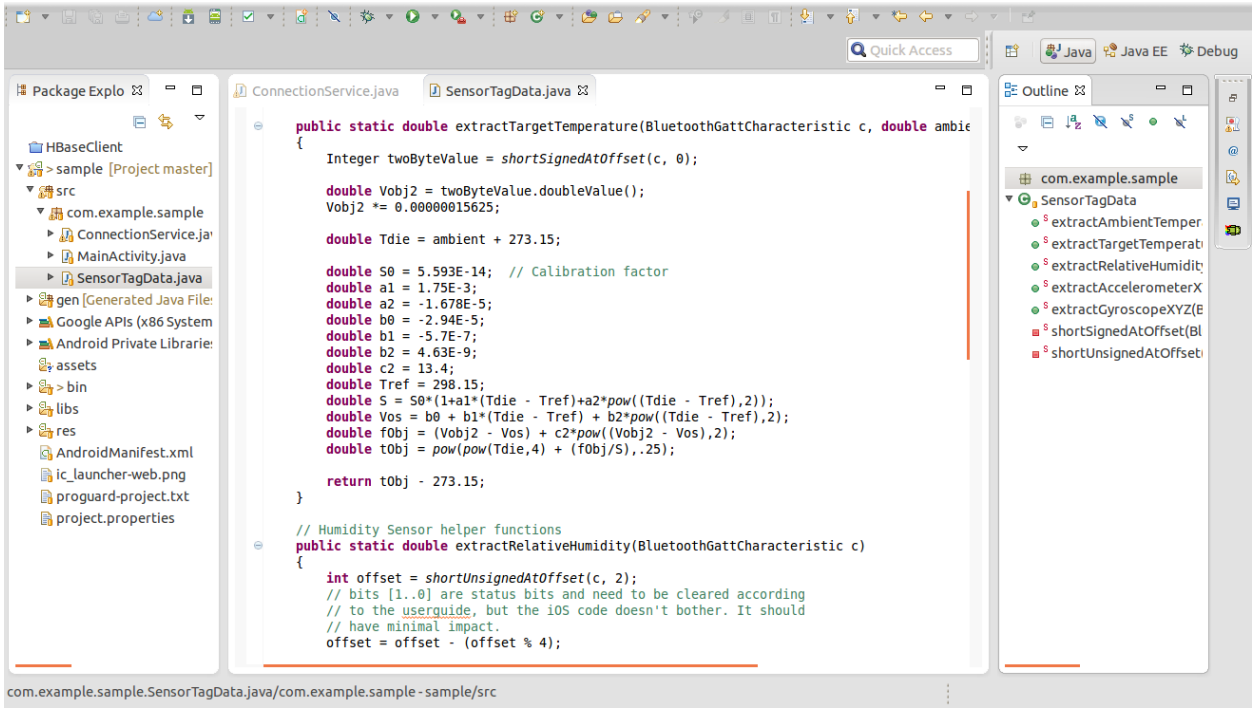
enableNextSensorNotification():



After enabling and subscribing to data notifications for a given sensor, the onCharacteristcChanged() callback method is invoked every interval for each sensor. It calls static conversion functions in a SensorTag class to extract the raw data and convert it to a useful form. It then calls the SaveData() function to write this data to a text file.



Raw data conversion functions borrowed from:
http://processors.wiki.ti.com/index.php/SensorTag_User_Guide



Example output written to phone storage:

```
...
Sat Jun 21 23:27:31 CDT 2014, Orientation, X:Y:Z (deg/s): -164.04724, 99.28894,
-215.39307
Sat Jun 21 23:27:31 CDT 2014, Relative Humidity (%RH): 29.202564239501953
Sat Jun 21 23:27:31 CDT 2014, Acceleration, X:Y:Z (g): 0.0625, -1.1875, -0.65625
Sat Jun 21 23:27:31 CDT 2014, Ambient Temperature (°C): 32.46875
Sat Jun 21 23:27:31 CDT 2014, Object Temperature (°C): 20.717300346350214
Sat Jun 21 23:27:32 CDT 2014, Orientation, X:Y:Z (deg/s): -193.34412, 214.79797,
-24.795532
Sat Jun 21 23:27:32 CDT 2014, Relative Humidity (%RH): 28.958419799804688
Sat Jun 21 23:27:32 CDT 2014, Acceleration, X:Y:Z (g): 0.203125, -0.796875, -1.1875
Sat Jun 21 23:27:32 CDT 2014, Ambient Temperature (°C): 32.40625
Sat Jun 21 23:27:32 CDT 2014, Object Temperature (°C): 16.790920976928078
...
```

Part 2 - HBase

Modify /etc/hosts file and add record (Windows 7: located at c:\windows\system32\drivers\etc)
134.193.136.147 localhost.localdomain localhost

The output in console:

```
<terminated> Start [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe (Jun 23, 2014, 10:59:53 PM)
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:zookeeper.version=3.3.2-1031432, built on 11/05/20
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:host.name=JWC-PC
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:java.version=1.7.0_51
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:java.vendor=Oracle Corporation
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:java.home=C:\Program Files\Java\jre7
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:java.class.path=C:\LocalStorage\Workspaces\Android
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:java.library.path=C:\Program Files\Java\jre7\bin;C
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:java.io.tmpdir=C:\Users\JWC\AppData\Local\Temp\
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:java.compiler=<NA>
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:os.name=Windows 7
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:os.arch=amd64
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:os.version=6.1
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:user.name=JWC
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:user.home=C:\Users\JWC
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Client environment:user.dir=C:\LocalStorage\Workspaces\Android\hbase.
14/06/23 22:59:53 INFO zookeeper.ZooKeeper: Initiating client connection, connectString=134.193.136.147:2181 sess
14/06/23 22:59:54 INFO zookeeper.ClientCnxn: Opening socket connection to server /134.193.136.147:2181
14/06/23 22:59:54 INFO zookeeper.RecoverableZooKeeper: The identifier of this process is 5332@JWC-PC
14/06/23 22:59:54 INFO zookeeper.ClientCnxn: Socket connection established to localhost.localdomain/134.193.136.1
14/06/23 22:59:54 INFO zookeeper.ClientCnxn: Session establishment complete on server localhost.localdomain/134.1
14/06/23 22:59:55 INFO client.HConnectionManager$HConnectionImplementation: Closed zookeeper sessionid=0x146aa76a
14/06/23 22:59:55 INFO zookeeper.ZooKeeper: Session: 0x146aa76accc0128 closed
14/06/23 22:59:55 INFO zookeeper.ClientCnxn: EventThread shut down
14/06/23 22:59:55 INFO zookeeper.ZooKeeper: Initiating client connection, connectString=134.193.136.147:2181 sess
14/06/23 22:59:55 INFO zookeeper.ClientCnxn: Opening socket connection to server /134.193.136.147:2181
14/06/23 22:59:55 INFO zookeeper.RecoverableZooKeeper: The identifier of this process is 5332@JWC-PC
14/06/23 22:59:55 INFO zookeeper.ClientCnxn: Socket connection established to localhost.localdomain/134.193.136.1
14/06/23 22:59:55 INFO zookeeper.ClientCnxn: Session establishment complete on server localhost.localdomain/134.1
people person1:what_is_a_qualifier 1403582117202 tony
people person2:what_is_a_qualifier 1403582117281 teja
people person3:what_is_a_qualifier 1403582117352 james
people person4:what_is_a_qualifier 1403582117422 jw
```

Code to create table, insert rows, read all records:

```
package hbase.console;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.hbase.HBaseConfiguration;
import org.apache.hadoop.hbase.HColumnDescriptor;
import org.apache.hadoop.hbase.HTableDescriptor;
import org.apache.hadoop.hbase.KeyValue;
import org.apache.hadoop.hbase.client.HBaseAdmin;
import org.apache.hadoop.hbase.client.HTable;
import org.apache.hadoop.hbase.client.Put;
import org.apache.hadoop.hbase.client.Result;
import org.apache.hadoop.hbase.client.ResultScanner;
import org.apache.hadoop.hbase.client.Scan;
import org.apache.hadoop.hbase.util.Bytes;

public class HBaseConsole {
```

```
    //UMKC Cloudera
    private final String HBASE_ZOOKEEPER_QUORUM_IP = "134.193.136.147";
    private final String HBASE_ZOOKEEPER_PROPERTY_CLIENTPORT = "2181";
    private final String HBASE_MASTER = HBASE_ZOOKEEPER_QUORUM_IP + ":60010";

    public HBaseConsole() {

    }

    public void createTable(String table, String columnFamilies) throws Exception {
        HBaseAdmin hba = null;
        Configuration config = getHBaseConfiguration();
        HTableDescriptor ht = new HTableDescriptor(table);
        for (String columnFamily : columnFamilies.split(":")) {
               ht.addFamily(new HColumnDescriptor(columnFamily));
        }
        hba = new HBaseAdmin(config);
        hba.createTable(ht);
        hba.close();
    }

    public void insertRow(String table, String row, String family, String qualifier,
String value) throws Exception {
        Configuration config = getHBaseConfiguration();
        HTable ht = new HTable(config, table);
        Put put = new Put(Bytes.toBytes(row));
        put.add(Bytes.toBytes(family), Bytes.toBytes(qualifier), Bytes.toBytes(value));
        ht.put(put);
    }

    public String getRecord(String table) throws Exception {
       String line="";

       Configuration config = getHBaseConfiguration();

       HTable ht = new HTable(config, table);
       Scan s = new Scan();
       ResultScanner ss = ht.getScanner(s);
       for(Result r:ss){
              for(KeyValue kv : r.raw()){
                       line = line+ new String(kv.getRow()) + " ";
                       line = line + new String(kv.getFamily()) + ":";
                       line = line + new String(kv.getQualifier()) + " ";
                       line = line + kv.getTimestamp() + " ";
                       line = line + new String(kv.getValue());
                       line = line + "/n";
```

```
            }
        }
        return line;
    }

    private Configuration getHBaseConfiguration() {
        Configuration config = HBaseConfiguration.create();
        config.clear();
        config.set("hbase.zookeeper.quorum", HBASE_ZOOKEEPER_QUORUM_IP);
        config.set("hbase.zookeeper.property.clientPort",
HBASE_ZOOKEEPER_PROPERTY_CLIENTPORT);
        config.set("hbase.master", HBASE_MASTER);
        return config;
    }
}
```

Code that calls the previous class:

```
package hbase.console;

public class Start {

    public static void main(String[] args) {
        try {
               HBaseConsole con = new HBaseConsole();
               con.createTable("ttjj_lab2_part2", "person1:person2:person3:person4");
               con.insertRow("ttjj_lab2_part2", "people", "person1",
                     "what_is_a_qualifier", "tony");
               con.insertRow("ttjj_lab2_part2", "people", "person2",
                     "what_is_a_qualifier", "teja");
               con.insertRow("ttjj_lab2_part2", "people", "person3",
                     "what_is_a_qualifier", "james");
               con.insertRow("ttjj_lab2_part2", "people", "person4",
                     "what_is_a_qualifier", "jw");
               String s = con.getRecord("ttjj_lab2");
               System.out.println(s);

        } catch (Exception e){
               System.out.println(e.getMessage());
        }
    }
}
```

A method for parsing the Sensor.txt file

```
public void insertSensorsTxt(String table, String row, String pathToFile)
      throws Exception {
    String ambient_temp = "ambient_temp";
    String object_temp = "ambient_temp";
    String relative_humidity = "relative_humidity";
    String acceleration = "acceleration";
    String family = "";
    String qualifier = "Q"; //what's this do?
    String value = "";
    Configuration config = getHBaseConfiguration();
    HTable ht = new HTable(config, table);
    BufferedReader br = null;
    String sCurrentLine;
    br = new BufferedReader(new FileReader(pathToFile));
    int count = 1;
    while ((sCurrentLine = br.readLine()) != null) {
        System.out.println(sCurrentLine);
        Put put = new Put(Bytes.toBytes(row));
        value = sCurrentLine.split(":")[3].trim();
        switch(count % 4) {
              case 1:
                    family = ambient_temp;
                    break;
              case 2:
                    family = object_temp;
                    break;
              case 3:
                    family = relative_humidity;
                    break;
              case 4:
                    family = acceleration;
                    break;
        }
        put.add(Bytes.toBytes(family), Bytes.toBytes(qualifier), Bytes.toBytes(value));
        ht.put(put);
        count++;
    }
    if (br != null) {
        br.close();
    }
}
```