Use JavaSmap to collect data from sMap

UC Berkeley, Fall 2016

Contents

Clear everything

```
close all
clear all
clear java
```

Start and End times, using UTC in Java representation

We will grab data on August 30, 2014, from 10:00AM to 4:00PM. Use the `datenum` format to convert to a serial date, and then to the Java UTC time convention.

```
startTime = PDTtoUTC(datenum(2014,8,30,10,0,0));
endTime = PDTtoUTC(datenum(2014,8,30,16,0,0));
```

Setup to the data retrieval

We want data from Etcheverry Hall (which is stored at the openbms server). Set up the url (to get to the server) and the query (to access Etcheverry Hall information)

```
archiverUrl = 'http://new.openbms.org/backend/api/query';
query = 'Metadata/Location/Building = "Etcheverry Hall"';
```

Use sMAP to read the data

Send specific query, including start and end times, to the server

```
data = MatlabSmapRead(archiverUrl, query, startTime, endTime);
```

```
select data in (1409417999999, 1409439599999) limit 1000 streamlimit -1 where Metadata/Locati
on/Building = "Etcheverry Hall"
200 OK
```

Check class of returned variable

The returned value, `data` is a cell array. The contents of each element of the cell is a scalar struct, with two fields, `.Readings` and `.uuid`. The value of the `.Readings` field is an N-by-2 array containing the time-series (time in column 1, measured quantity in column 2). The value of the `.uuid` field is a character string, which is an identifier as to what the reading correspond to. The user must know the identifier of the physical variables they are interested in studying.

```
class(data)
```

```
ans =
cell
```

```
size(data)
```

```
ans =
     1   612
```

```
fieldnames(data{1})
```

```
ans =
    'Readings'
    'uuid'
```

Pull apart each struct in data{i} and store in separate cell arrays.

This can be done in a cleaner way, but here we use a simple FOR loop to illustrate the extraction.

```
n = numel(data);
readings = cell(n,1);
uuid = cell(n,1);
for i = 1:n
    readings{i} = data{i}.Readings;
    uuid{i} = data{i}.uuid;
end
```

Get specific Fan Data, and convert

```
FanID = '378ce506-1b39-5b3c-9cf7-d3ccdf7c2cf3';
FanIndex = find(strcmp(FanID,uuid));
FanData = readings{FanIndex};
size(FanData)
```

```
ans =
   360     2
```

Convert time (java UTC) back to PDT (in Matlab serial date format)

```
FanData(:,1) = UTCtoPDT(FanData(:,1));
```

RoomTemp

```
RoomTempID = 'a53eea71-be6c-54b9-9098-6453a2083279';
RoomTempIndex = find(strcmp(RoomTempID,uuid));
RoomTempData = readings{RoomTempIndex};
RoomTempData(:,1) = UTCtoPDT(RoomTempData(:,1));
```

SupplyTemp

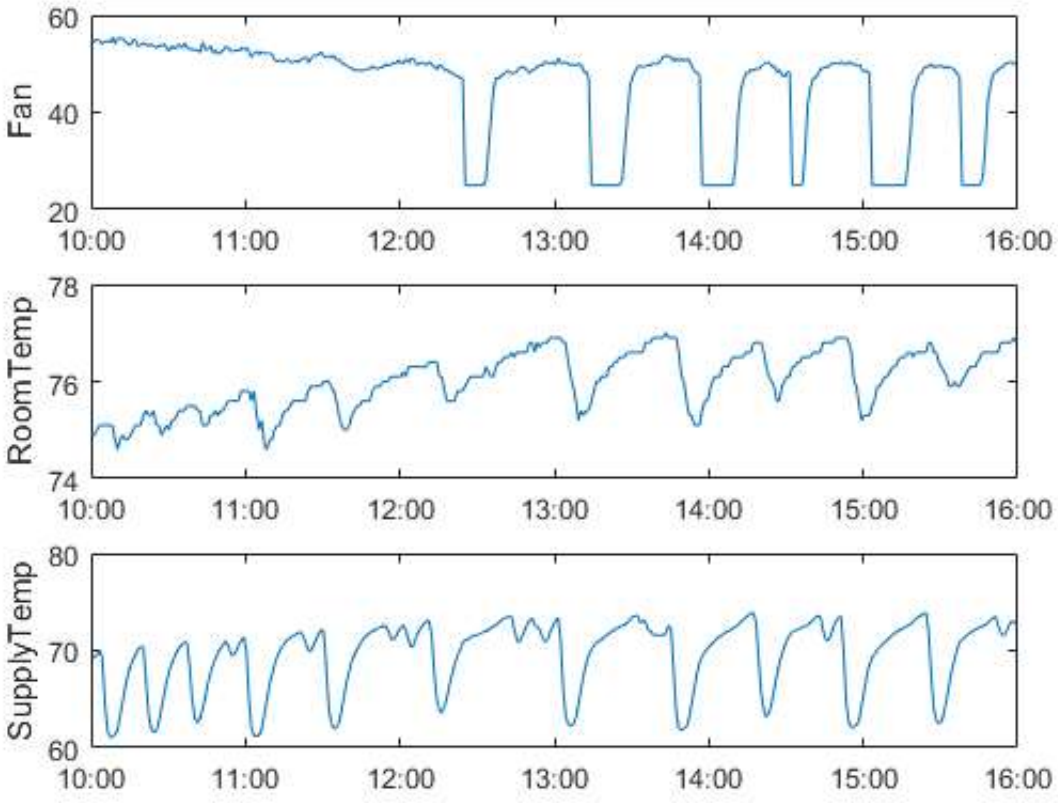
```
SupplyTempID = '5d299b4b-d59a-5ae9-8511-c7d58067d453';
SupplyTempIndex = find(strcmp(SupplyTempID,uuid));
SupplyTempData = readings{SupplyTempIndex};
SupplyTempData(:,1) = UTCtoPDT(SupplyTempData(:,1));
```

Process data here

If you had some data processing to do, here is where that code would go, acting on the time-series data that has been extracted.

Plot the Data

```
subplot(311)
plot(FanData(:,1), FanData(:,2))
ylabel('Fan')
datetick('x')
subplot(312)
plot(RoomTempData(:,1), RoomTempData(:,2))
datetick('x')
ylabel('RoomTemp')
subplot(313)
plot(SupplyTempData(:,1), SupplyTempData(:,2))
datetick('x')
ylabel('SupplyTemp')
```



Attribution

UC Berkeley, Fall 2016. ME C231A and EECS C220B. Author(s): Sarah Koehler, skoehler@berkeley.edu, Andy Packard, apackard@berkeley.edu Last Updated: 9/06/2016

Published with MATLAB® R2016a