Internet of Things at Bohunt School (Wokingham)

Tracking a mobile device

This practical session should be a bit of fun for you.

The objective is to track the GPS position of a mobile device (i.e. tablet or cell phone) and write this information to a database. Then by running a query on the database display the selected information on 'worldmap' in Node-RED.

Here's an image of what the displayed results could look like.

Apart from the Raspberry Pi (running Node-RED and MQTT) and your cell-phone no other hardware is needed.

These are the main steps involved.

*** Set-up a MQTT broker on the cloud**

*** Download "Owntracks - app" on your mobile device**

*** Set-up a MySQL database**

*** Write a Node-RED flow to control the system**

The overall time involved in this project is 30 to 45 minutes.

Right let's get started.

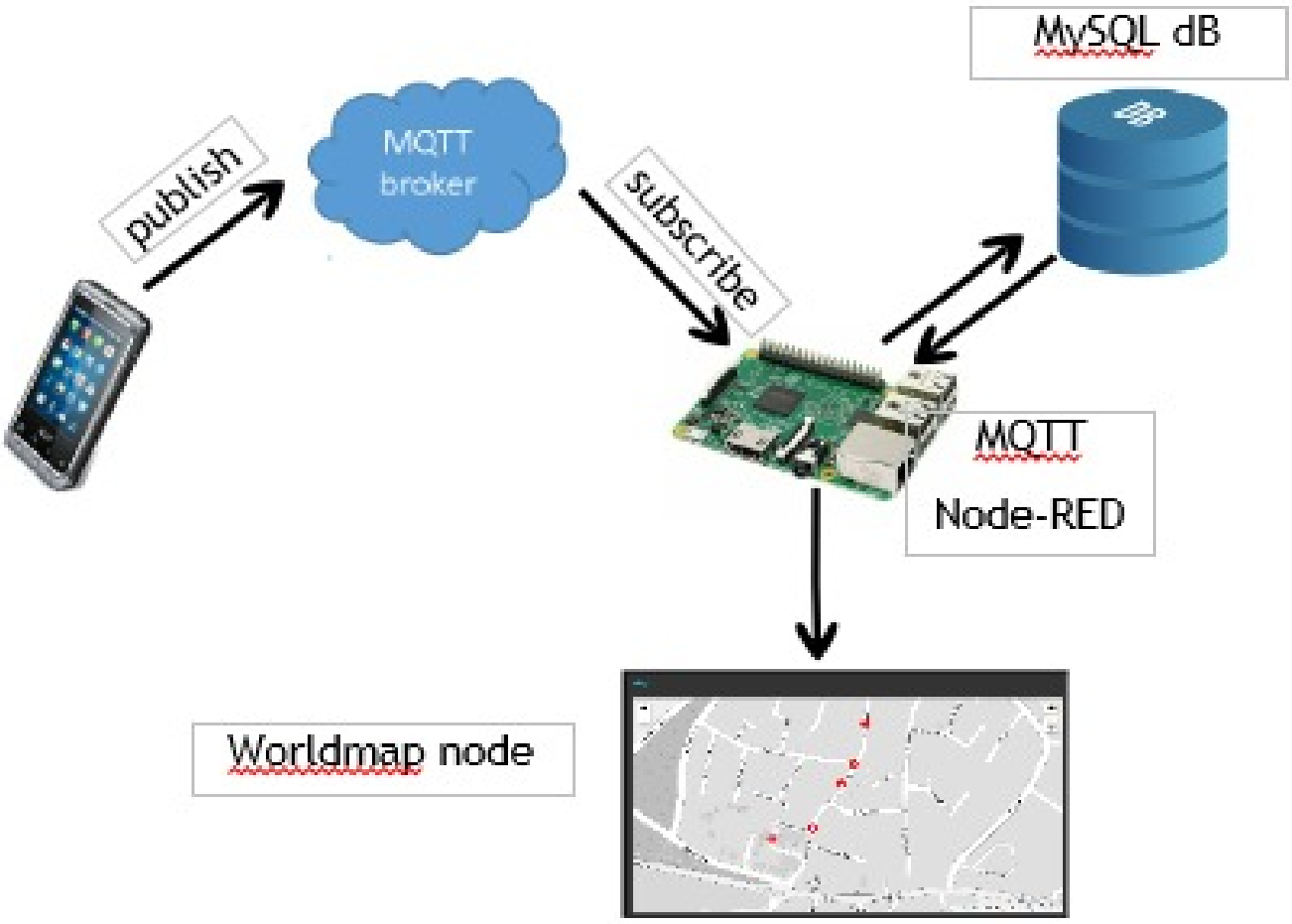
If you want to jump ahead, here's a link to the Node-RED flow file.

Internet of Things at Bohunt School (Wokingham)

Tracking a mobile device

Block diagram of the system

Here's a diagram to show the key parts of the system.



The 'owntracks' app on the mobile phone publishes the GPS coordinates whenever the mobile device moves a certain distance within a certain time.

The Raspberry Pi, running MQTT and Node-RED, subscribes to the MQTT cloud broker to receive the published information.

The Node-RED flow performs a number of tasks including:

- **performing a sanity check on the data received from MQTT (it ignores Last Will and Testament (LWT) detections/indications)**
- **writing the data to a MySQL database**
- **updating 'worldmap' to show the latest position of a marker**

The dashboard, in the flow, enables the user to select and display marker information for the various mobile devices.

Internet of Things at Bohunt School (Wokingham)

Tracking a mobile device

Set-up a MQTT broker on the cloud

There are many providers on the cloud that are available.

The one I selected was:- https://www.cloudmqtt.com/

I signed-up for a free account called “Cute Cat” - make a note of the User Name, Password and Port number as you will need this information later.

Download “owntracks” onto your mobile device.

Go to https://owntracks.org/ and download the app for your platform.

You need to configure the ‘app’ to publish to a Private MQTT in the Mode setting. Enter the MQTT details from above into the settings panel.

Quick check to see if things are working

Create this simple flow in Node-Red.

If you click the Report/Update button in the app you should see the track-data reported in the debug window.

owntracks/kxxaennx/lenovo : msg.payload : Object


```
object
  _type: "location"
  tid: "05"
  acc: 15
  batt: 50
  conn: "w"
  lat: 51.3461592
  lon: -0.8270807
  t: "u"
  tst: 1531070744
  _cp: true
```


Note the data reported for ‘tid’, ‘lat’, ‘lon’ and ‘tst’.

This is the device ID, latitude and longitude, and time-stamp for the detection.

Internet of Things at Bohunt School (Wokingham)

Tracking a mobile device

Set-up a MySQL database

I decided to use a MySQL database on one of my servers. If you prefer you could use a local database on your Raspberry Pi with 'sqlite' or 'mongodb'.

Here's what the schema looks like.

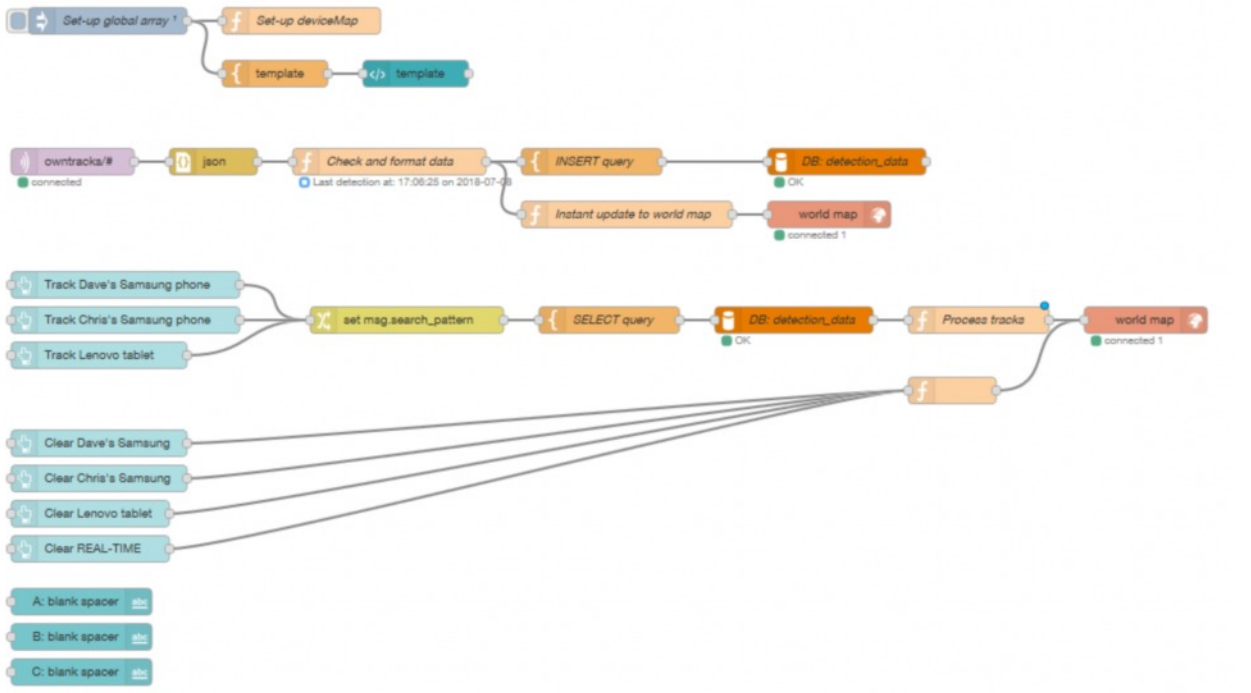
#	Name	Datatype	Length/Set	Unsigned	Allow NULL	Zerofill	Default
1	owntracks_id	MEDIUMINT	8	☑	☐	☐	AUTO_INCREMENT
2	device_id	CHAR	2	☐	☐	☐	No default
3	date	DATE		☐	☐	☐	No default
4	time	TIME		☐	☐	☐	No default
5	lat	DECIMAL	9,6	☐	☐	☐	No default
6	lon	DECIMAL	9,6	☐	☐	☐	No default

The 'device_id' is used to record which mobile device reported a track. Date, time, lat and lon are fairly obvious fields to record the track's information.

Right the next phase is to create a Node-RED flow to store and retrieve data.

Write a Node-RED flow to control the system

Here's what the complete flow looks like.



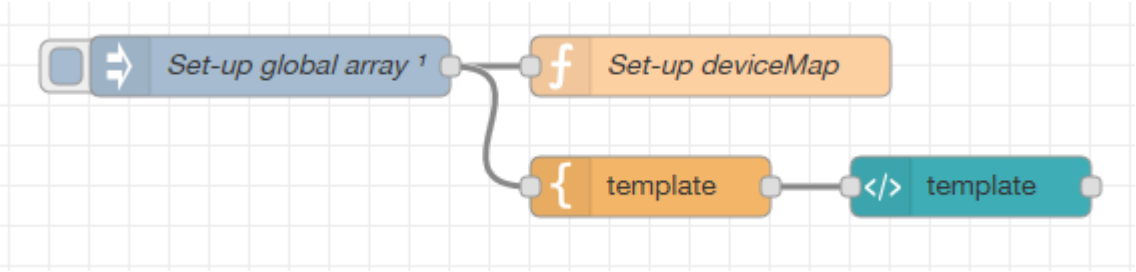
It may look complicated but when you break it down it becomes easier to read.

The first part of the flow is a 'mapping file' that maps your various mobile devices to names and identification icons (that will be used on 'worldmap').

Internet of Things at Bohunt School (Wokingham)

Tracking a mobile device

Here's the flow.



The mapping file (in the Function node) is a JavaScript object as shown below.

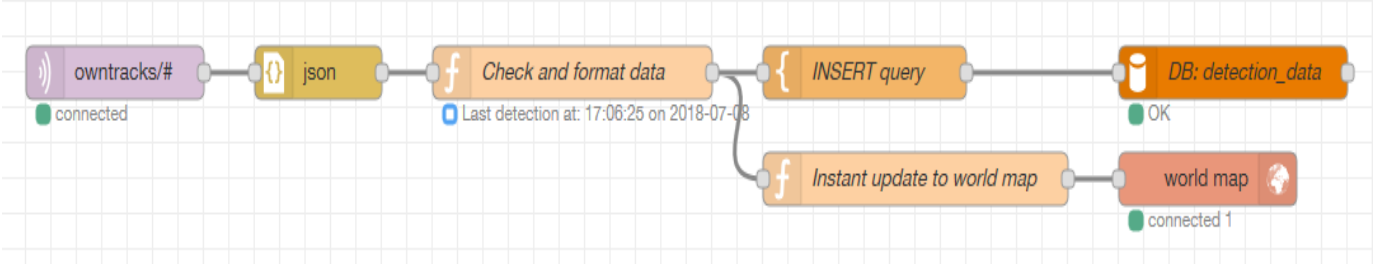
Function

```
1  global.set('deviceMap', {
2      "01" : {owner: "david", type: "Samsung", icon: "fa-map-pin", color: "blue"},
3      "02" : {owner: "chris", type: "Samsung", icon: "fa-paw", color: "green"},
4      "05" : {owner: "david", type: "Lenovo",  icon: "fa-map-pin", color: "red"},
5      "99" : {owner: "unknown", type: "Unknown",  icon: "fa-question-circle", color: "red"}
6  });
7
8  global.set("operational", "yes");
9
```

Just edit this file to match the name(s) of your mobile devices and your name. Entry "99" is used to catch any tracks that don't match known devices. More about this later.

The template node links the URL for 'worldmap' to the template UI.

Next part of the flow is the capture routine.



This routine converts the json data (received from Owntracks) to a JavaScript object which is then checked and formatted for the database. If a LWT (last will and testament) response has been received - then this is discarded.

The device id (msg.payload.tid) is checked to make sure it is in the mapping file. If it's not found, then the device id is changed to "99".

The resultant data is written to the database and sent to 'worldmap' to create a layer named 'realtime' and to instantly update the map.

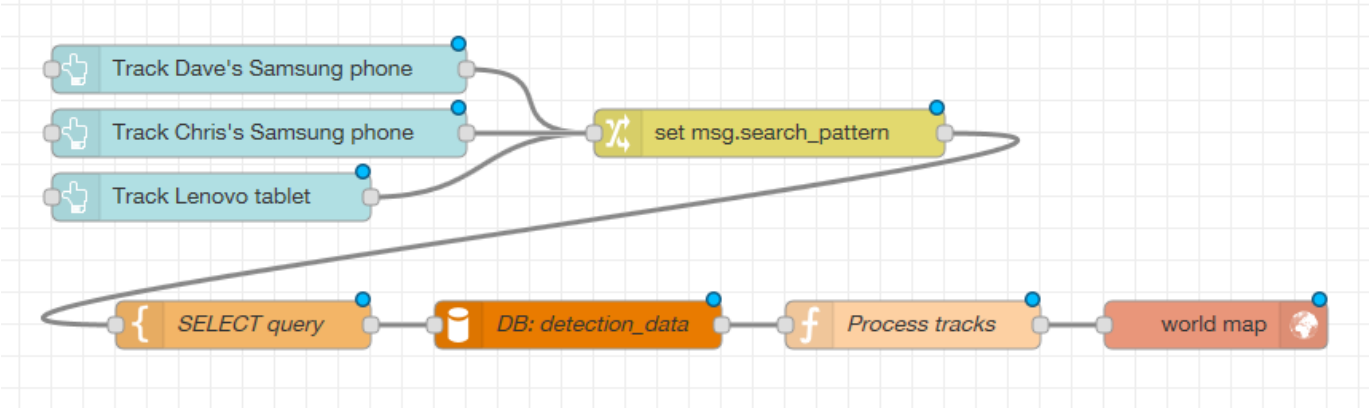
Once you've written this part of the flow you can check that markers appear on 'worldmap' whenever you click the update/report button on the app.

Internet of Things at Bohunt School (Wokingham)

Tracking a mobile device

Querying the database

This is the fun part as it enables you to plot markers for each mobile device.



The three UI buttons input the reference id for each mobile device. For example my mobile is reference "01". This is passed through the flow and used in the 'process tracks' function to process the results from the database.

The SELECT query uses the reference id to extract those particular data-sets.

Function

```
var search_pattern = msg.search_pattern;
var myArray = global.get('deviceMap');
var iColor  = myArray[search_pattern]['color'];
var iIcon   = myArray[search_pattern]['icon'];
var iLayer  = "device_" + search_pattern;

var tracks = msg.payload.map(function(m, i) {
    return {
            payload: {
                name: "mytrack_" + m.device_id +"_" + m.owntracks_id,
                layer: iLayer,
                lat: parseFloat(m.lat),
                lon: parseFloat(m.lon),
                time: (m.time),
                date: (m.date),
                device: (m.device_id),
                icon: iIcon,
                iconColor: iColor
            }
        };
    });
return [tracks];
```

The search_pattern (from the UI buttons) is used to map the color and icon name for the particular device, while the .map function extracts the stored values for the particular mobile device. This is sent to 'worldmap'.

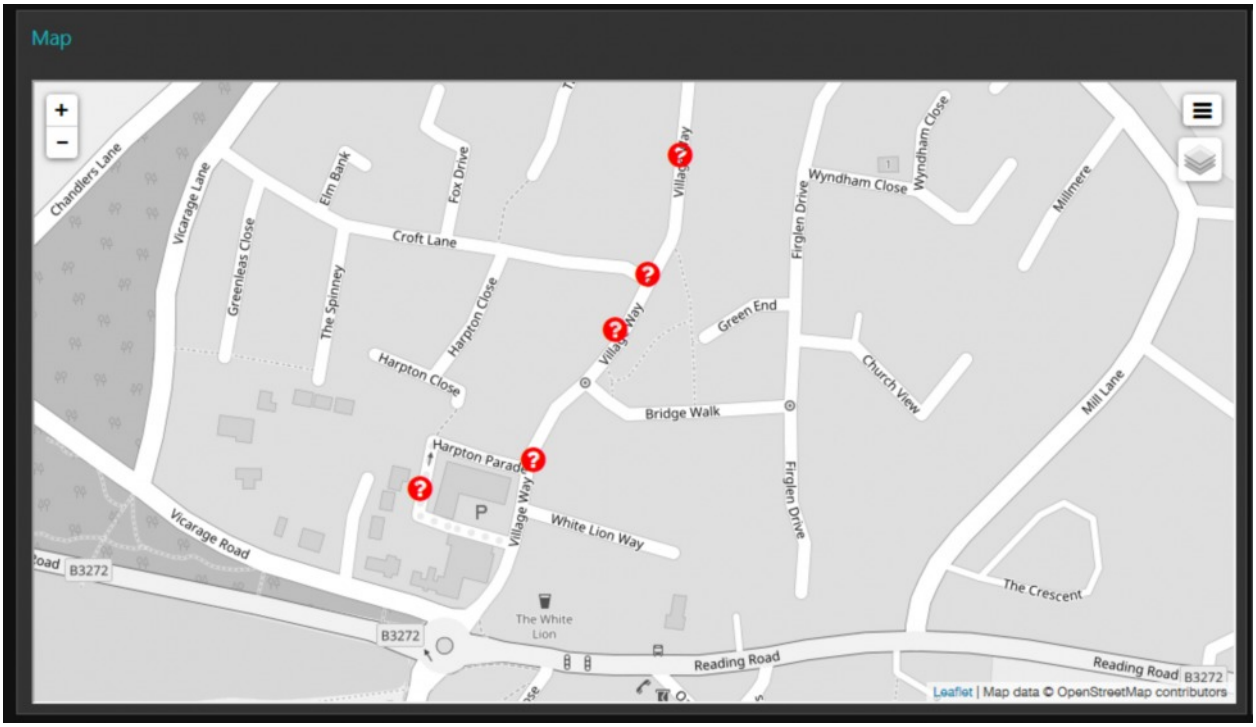
Internet of Things at Bohunt School (Wokingham)

Tracking a mobile device

Typical results obtained from querying a device

This is a screen-shot of the markers for my mobile phone (ref "01").

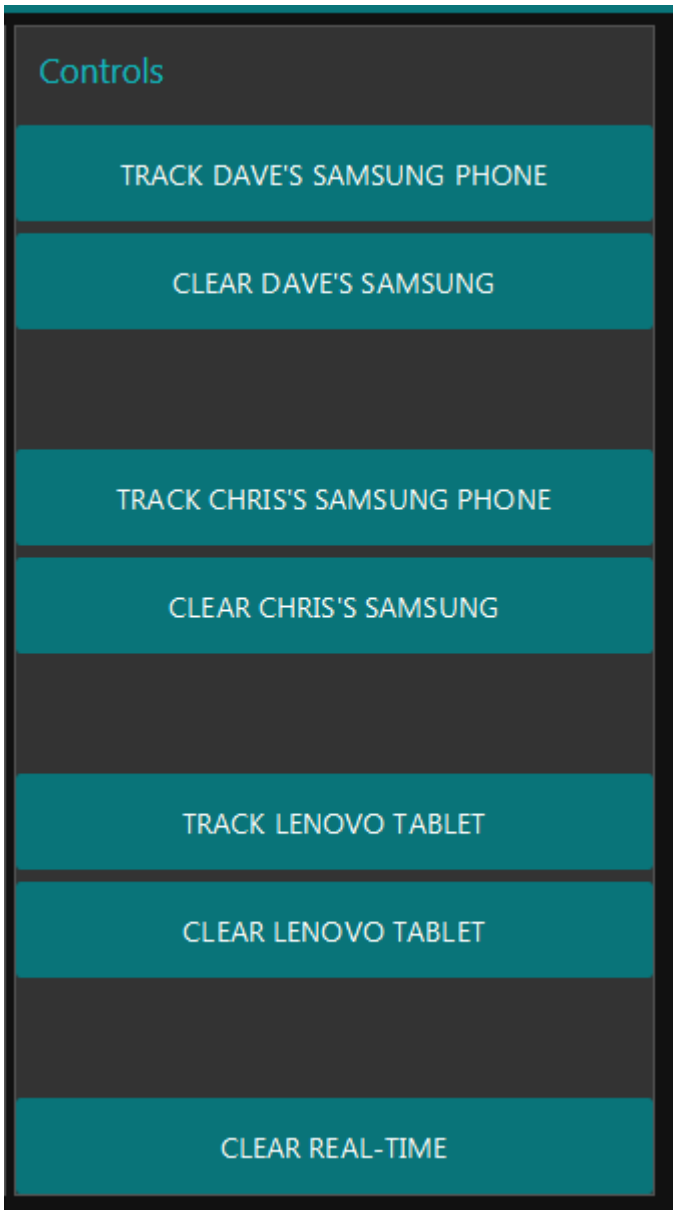
This is a screen-shot of the markers for an unknown device (ref "99").



These markers are controlled by the dashboard buttons as shown on the next page. For example, they can be selectively switched on or off.

Internet of Things at Bohunt School (Wokingham)

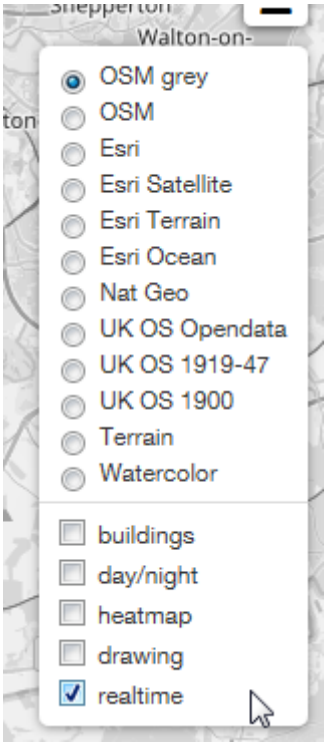
Tracking a mobile device



I know - it's a pretty boring layout.

I'm sure you could improve it.

The wording on the buttons says it all.



The above insert is a screen-shot from 'worldmap' showing the 'realtime' layer. This layer can be cleared using the 'Clear Real-Time' button opposite.

Right that's about it. I hope you have managed to get your flow working.

Acknowledgements

I need to thank a couple of people, Dave Conway-Jones (dceejay) for his help/advice with 'worldmap' and Steve Rickus for guidance on databases and the *.map function* to filter/process the database results.

Link to Node-RED file

Click this link to download the Node-RED flow.

Congratulations you can now track a mobile device.