



# **Visualize data with third-party applications using Digi Remote Manager®**

---

September, 2017

90002251

## Contents

Overview .....	3
Step 1: Create your dweet “thing” .....	3
Create a Push Monitor.....	3
Test the Push Monitor .....	4
Step 2: Create the Freeboard dashboard.....	5
Create a Freeboard account or login .....	5
Add a data source .....	5
Add a pane .....	5
That’s it!.....	6
Extra credit.....	6
Convert to Fahrenheit.....	6
Configure to scale .....	7
Example: Aggregate a Push Monitor for all similar sensor endpoints .....	7

## Overview

This application note describes how to create a linkage between Digi Remote Manager® and third-party applications using the Push Monitor feature. In this example, we will configure Remote Manager to push data to Freeboard.io via Dweet.io. The actual visualization is done using Freeboard.io. In this example, Dweet.io acts as a conduit between Remote Manager and Freeboard. This is necessary since Dweet.io provides an HTTP "listener" that publishes a public URL accessible to Remote Manager. Freeboard then allows you to visualize the data pushed into Dweet.io from Remote Manager.

When using the Dweet.io and Freeboard.io components for this example, we need to utilize one push monitor per sensor endpoint. In order to scale to large numbers of endpoints for a production implementation, you would configure a push monitor to aggregate data and push data for large numbers of devices. See the [Configure to scale](#) section for an example of how to use the Push Monitor feature for thousands of devices.

## Step 1: Create your dweet “thing”

For this example, we will use the free version of Dweet.io. A “thing” is basically a data stream in Dweet.io that holds about 5 recent data points.

You can POST and GET data. The free version means your thing is exposed to the world without the need for an API key to access the data. If you would like to lock your thing, you can do that; but there is a fee. You can find more information here: <https://dweet.io/locks>.

The Dweet.io API docs are here: <https://dweet.io/play/>.

To setup your Dweet thing, all you have to do is create a Push Monitor in Remote Manager. The thing must be a unique name on Dweet.io. We will use a Remote Manager stream ID, which includes the device ID, so we know it is likely to be unique.

### Create a Push Monitor

For this example, we're using a temperature sensor from a SmartPlug. The smart plug is named rpm1 and is associated with a gateway with this ID:

00000000-00000000-00409DFF-FF5C525A.

The stream ID for this sensor is:

dia/channel/00000000-00000000-00409DFF-FF5C525A/rpm1/temperature

Navigate to the Remote Manager API Explorer and create an HTTP Push Monitor similar to the following:

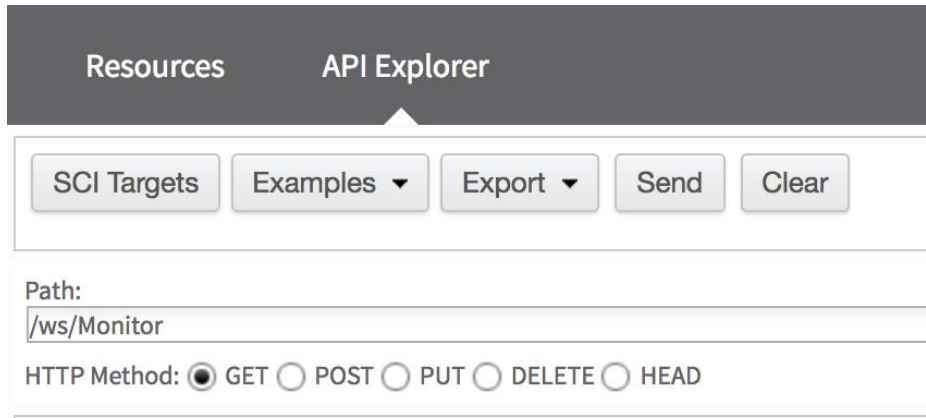
```
<Monitor>
<monTopic>DataPoint/dia/channel/00000000-00000000-00409DFF-FF5C525A/rpm1/temperature</monTopic>
<monTransportType>http</monTransportType>
<monTransportUrl>https://dweet.io/dweet/for/00000000-00000000-00409DFF-FF5C525A-rpm1-temperature</monTransportUr1>
<monFormatType>json</monFormatType>
<monBatchSize>1</monBatchSize>
<monCompression>none</monCompression>
```

```
<monBatchDuration>1</monBatchDuration>  
<monTransportMethod>POST</monTransportMethod>  
</Monitor>
```

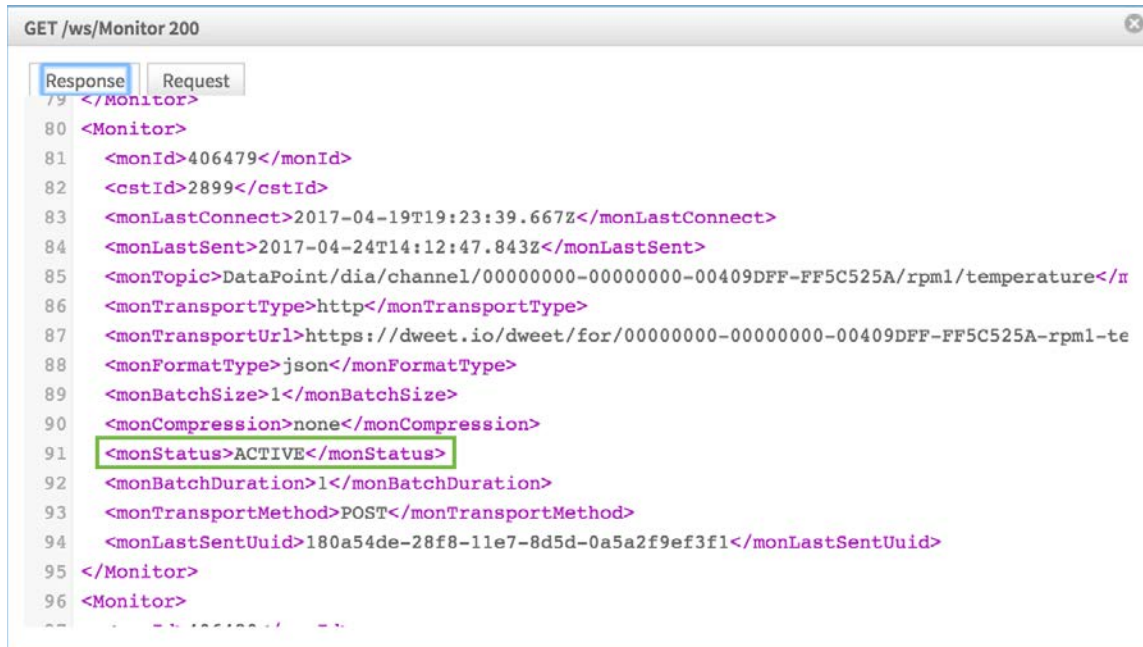
**Note:** Our Dweet "thing" is named "00000000-00000000-00409DFF-FF5C525A-rpm1-temperature".

## Test the Push Monitor

First, perform a GET against /ws/Monitor to see if your newly-created monitor is ACTIVE. From the API Explorer, ensure the path is /ws/Monitor, click the "GET" radio button and click the "Send" button.



Here are the results, showing the monitor is active:



You can confirm the push monitor is working by checking Dweet.io to ensure your data is there. Perform a GET to <https://dweet.io/get/dweets/for/00000000-00000000-00409DFF-FF5C525A-rpm1-temperature> to get the last five data points.

## Step 2: Create the Freeboard dashboard

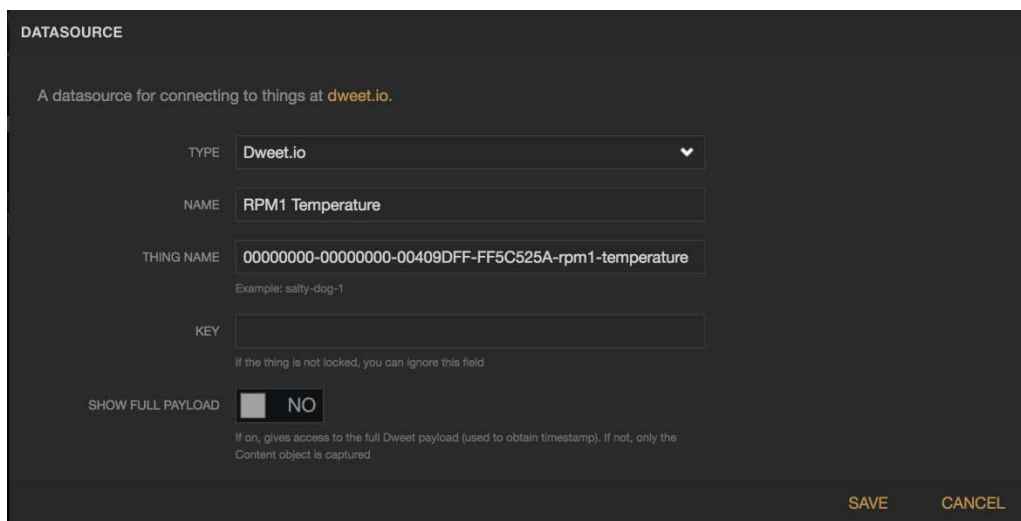
Freeboard.io is a JavaScript dashboarding tool. Similar to Dweet.io, with the free version, your dashboards are public. You can create private dashboards by paying a monthly fee. There are different tiers depending on how many private dashboards you would like to have. You can find more information here: <https://freeboard.io/#pricing>.

### Create a Freeboard account or login

If you do not yet have an account, create a new account and log in. You'll have a blank list of "My Freeboards." Pick a name (for example, "Test") and click "Create New".

### Add a data source

1. Select "Add."
2. Choose the Dweet.io data source type from the drop-down list.
3. Name the data source. For this example, we'll call it "RPM1 Temperature."
4. Input your "thing" name.

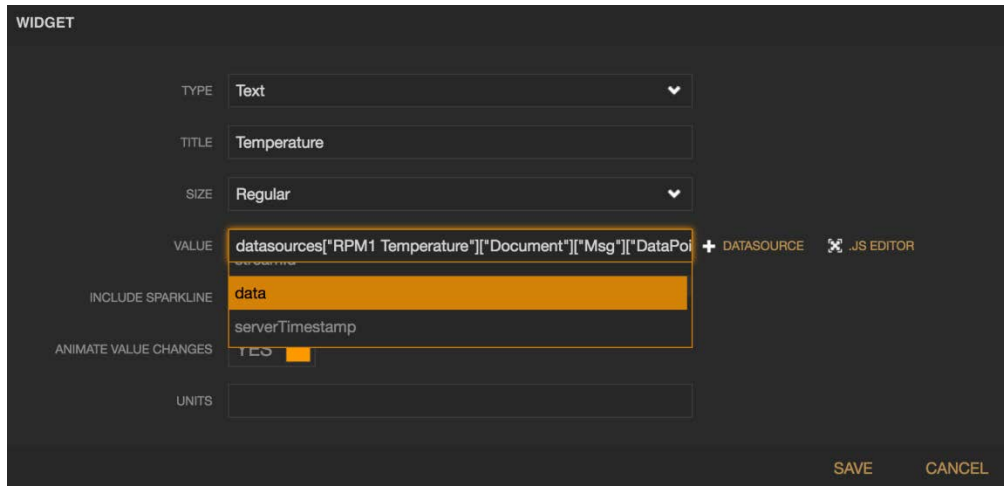


The screenshot shows a dark-themed form titled "DATASOURCE". Below the title is a subtitle: "A datasource for connecting to things at [dweet.io](https://dweet.io)". The form contains several fields: "TYPE" is a dropdown menu set to "Dweet.io"; "NAME" is a text input field containing "RPM1 Temperature"; "THING NAME" is a text input field containing a long alphanumeric string "00000000-00000000-00409DFF-FF5C525A-rpm1-temperature", with a small example "Example: salty-dog-1" below it; "KEY" is an empty text input field with a note below it: "If the thing is not locked, you can ignore this field"; "SHOW FULL PAYLOAD" is a toggle switch currently set to "NO", with a note below it: "If on, gives access to the full Dweet payload (used to obtain timestamp). If not, only the Content object is captured". At the bottom right of the form are two buttons: "SAVE" and "CANCEL".

### Add a pane

Now that you have a data source defined, you can add a pane and dashboard widget.

1. Click **Add Pane** to add a pane to the dashboard area on the page.
2. Click the **+** icon to add a dashboard widget element.
3. For this example, select the **Text** type.
4. Give it a title.
5. For the value, select from the **Datasource** drop-down and pick the data source you created in the previous step. You will need to drill down from **Thing Name > Document > Msg > DataPoint > Data**.
6. Click **Save**.



## That's it!

You should now see data being updated on your dashboard widget. This data will update at the same frequency Remote Manager is pushing data.

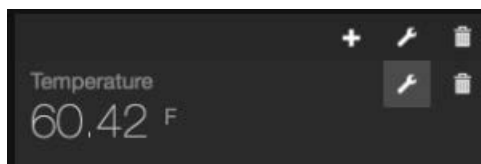


## Extra credit

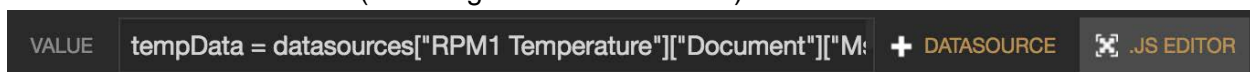
Now that you have your Freeboard dashboard linked to your Remote Manager data, you can manipulate it using JavaScript.

### Convert to Fahrenheit

1. Edit your widget by clicking the wrench icon.



2. Click the .JS Editor (to the right of the Value field).



In this case, we created a variable from the data source element and applied the C to F conversion. We also limited the decimal places to 2.

```
tempData = datasources["RPM1  
Temperature"]["Document"]["Msg"]["DataPoint"]["data"]  
return (tempData * 1.8 + 32).toFixed(2)
```

### Configure to scale

The proof-of-concept example above utilizes 1 push monitor per individual data stream. That would not be feasible in a large-scale deployment. For that, we would recommend configuring a push monitor that will represent multiple devices in aggregate. You could configure a single push monitor to push all data for all devices. Or, if you wanted to segment the data, you could configure a push monitor to aggregate per sensor endpoint or group. In addition, you could restrict the push monitor to just create, update or delete events. For more information about push monitors, see the [Monitor section](#) of the *Digi Remote Manager Programmer Guide*.

### Example: Aggregate a Push Monitor for all similar sensor endpoints

This push monitor is similar to the one illustrated above. The only difference is that it uses a wildcard in place of the device ID and sensor name, but it retains the specific sensor endpoint. This instructs the push monitor to push data from ALL devices that have the sensor endpoint "temperature."

```
<Monitor>  
<monTopic>DataPoint/dia/channel/*/*/temperature</monTopic>  
<monTransportType>http</monTransportType>  
<monTransportUrl>{URL of your enterprise application}</monTransportUrl>  
<monFormatType>json</monFormatType>  
<monBatchSize>1</monBatchSize>  
<monCompression>none</monCompression>  
<monBatchDuration>1</monBatchDuration>  
<monTransportMethod>POST</monTransportMethod>  
</Monitor>
```