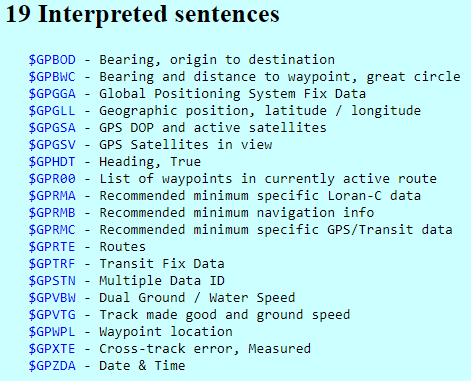
UVFit Photon Device API Datasheet

1. **Introduction**

This document will define all API’s from the Particle Photon to the web server. The Photon will interface to two sensors, the GPS and UV sensor. Each sensor has the ability to output several measurements.

1. **GPS Sensor Data Structure**

The GPS sensor outputs several types of GPS data at regular intervals. GPS data comes in a variety of what are called NEMA sentences.



However, our sensor uses only 4 of these: $GPGGA, $GPGSA, $GPRMC, $GPVTG. The datasheet for the GPS sensor recommends to only use the $GPRMC. A sentence of this structure will look something like this:



The data is separated by commas, obviously the first part is the NEMA sentence structure, for our case this is **$GPRMC**.

Following this is the Time Stamp: **194509.000**

This time is in Greenwich Mean Time (GMT) the digits are as follows

**HH:MM:SS:ms ms ms**

In our case we have the time 19:45 i.e. 7:45pm.

The next packet of data is a ‘status code’: **A for active and V for void**

The status code is void when the GPS sensor doesn’t have a link to a satellite.

Next up is the Latitude: **4042.6142,N** ------------- **DD:MM.MMMM N**

For us this is 40 degrees 42.6142 decimal minutes North

Similarly is the Longitude:**07400.4168,W ---------------- DDD:MM.MMMM W**

For us this is 074 degrees 00.4168 decimal minutes West

Next, speed in knots: **2.03 knots**

Next, is Heading Angle: **221.11 degrees**

Next, current Date: **160412 ----- 16th of April 2012**

Finally, \*XX is the checksum

1. **UV Sensor Data Structure**

The UV Sensor is very simple, it interfaces over I2C and simply outputs a UV light intensity **NOT A** **UV INDEX** like other sensors. The sensor can be configured to sample for several different lengths of time: **.5\_T, 1\_T, 2\_T, 4\_T**. The longer the sample time, the more precise the reading will be. The sensor outputs a 16-bit reading whenever it is queried, and according to the documentation they recommend calibrating this value to a known value depending on the application.

1. **API Definition**

|  |  |  |
| --- | --- | --- |
| **Resource**  **(endpoint and parameters)** | **Request Verb** | **Description** |
| /devices/register | POST | Register a new device.  Parameters (JSON):  {    "deviceId": "11f4baaef3445ff",    "email": "test@example.com"  }  Response (JSON):  Success (201):  {    "registered": true,    "message": "Device ID: 11f4baaef3445ff was registered."  }  Failure (400):  {    "registered": false,    "message": "Device ID 11f4baaef3445ff already registered."  } |
| /devices/activity | PUT | Update user activity  Parameters (JSON):  {  "deviceId": "11f4baaef3445ff",  "lat": "DDMM.MMMM",  “long”: “DDDMM.MMMM”,  “speed”: “speed(knots)”,  “start”: “start\_time(GMT)”,  “stop”: “stop\_time(GMT)”,  “uv”: “uv\_reading”  }  Response (JSON):  Success (201):  {  “logged”: true,  “message”: “Activity update logged.”  }  Failure (400):  {  “logged”: false,  “message”: “Error logging activity.”  } |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |