**Changing Update Rates**

Update rates can be easily changed to 1/5/10 Hz:

Find this line in the code and replace # with 1/5/10:

GPS.sendCommand(PMTK\_SET\_NMEA\_UPDATE\_#HZ);

1 Hz is a good value for parsing and printing out data, but it can go faster - use trial and error.

For an update rate that is not 1/5/10Hz:

<http://blogs.fsfe.org/t.kandler/2013/11/17/set-gps-update-rate-on-arduino-uno-adafruit-ultimate-gps-log>ger-shield/

Machine generated alternative text: Just in case someone wants to alter the GPS update rate on a Adafruit Ultimate GPS
Logger Shield. This may come in handy if you want to reduce the power consumption of
your board. According to a datasheet of the GPS-chip the max update rate is
I 0000msIl Osec. So how do you set it?
Relatively simple: go to your Adafruit_GPS-library, open Adafruit_GPS. h and look for
these lines:
#define PMTK_SET_NMEA_UPDATE_1HZ “$PMTK22O, 100ø*1F”
#define PMTK_SET_NMEA_UPDATE_SHZ $PMTK22OJ 200*2C”
#define PMTK_SET_NMEA_UPDATE_1ØHZ “$PMTK22Ø, 1øø*2F”
So this will define the name of the method/function/whatever which will be called in your
Arduino-sketch, e.g. PM TK_SET_NMEA_UPDA TE_1HZ. PMTK22O is the chip-internal
code for update rate. So we say: Hey, I want to alter the update rafe. The value after the
comma is the update rate in milliseconds. We set it to 10000 (or whatever you like). The
value behind the * is the checksum which the chip requires. Thanks to Stevens post
about reading GPS data with bash I stumpled upon the MTK NMEA checksum
calculator. So you put in PMTK22O, 10000 and get back $PMTK22O, 10000*2F. That’s it.
Our new line would read:
#define PMTK_SET_NMEA_UPDATE_1ØSEC ‘$PMTK22Ø, løøøe*2F”
Just set PMTK_SET_NPIEA_UPDATE_lesEc in your sketch and upload it.

**RGB LED Resistors and Changing Colors**

With 22/70 Ohm Resistors for all colors:

Orange = RGB(265, 80, 0)

* RGB values will differ from standard web RGB values because we're using the same resistors for all colors even though different colors have different voltage drops.
* If the value of resistors are changed, you'll need update RGB values in the code or you'll get colors you don't want.

**Getting other NMEA Data Values**

The GPS module currently gets RMC and GGA strings. They provide the following data, (not all are saved to file):

Machine generated alternative text: GGÅ - essential fix data which provide 3D location and accuracy data.
$GPGGA,123519,4807.038,N,01131.000,E,1,08,O.9,543.4,M,46.9,M,,*47
L4here:
GGA Global Positioning System Fix Data
123S19 Fix taken at 12:3S:l9 UTC
4807.038,N Latitude 48 deg 07.038’ N
01131.000,E Longitude 11 deg 31.000’ E
1 Fix quality: 0 = invalid
1 = GPS fix (SPS)
2 = DGPS fix
3 = PPS fix
4 = Real Time Kinematic
S = Float RTK
6 = estimated (dead reckoning) (2.3 feature)
7 = Manual input mode
8 = Simulation mode
08 Number of satellites being tracked
0.9 Horizontal dilution of position
S4S.4,M Altitude, Meters, above mean sea level
46.9,M Height of geoid (mean sea level) above WG584
ellipsoid
(empty field) time in seconds since last DGPS update
(empty field) DGPS station ID number
*47 the checksum data, always begins with *

Machine generated alternative text: R1C - NNIEA has its own version of essential gps pvt (position. velocity, time) data. It is called RMC. The
Recommended Mininmm, which will look similar to:
$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,230394,003.1,W*6A
Where:
RMC Recommended Minimum sentence C
123519 Fix taken at 12:35:19 UTC
A Status A=active or V=Void.
4807.038,N Latitude 48 deg 07.038’ N
01131.000,E Longitude 11 deg 31.000’ E
022.4 Speed over the ground in knots
084.4 Track angle in degrees True
230394 Date - 23rd of March 1994
003.1,W Magnetic Variation
*6A The checksum data, always begins with *

* RMC and GGA are the NMEA strings that Adafruit's GPS library is able to parse. You'd have to create your own parser function for other NMEA strings. There are websites which will parse NMEA strings.
* Other strings: <http://www.gpsinformation.org/dale/nmea.htm>

**Code/Layout/Pinouts**

on Box

**Other Notes**

* Try to not share the power pin (3v3) that the GPS module is connected to. The GPS module requires a clean power source and ripples will throw resulting data off.
* Pin out is interchangeable between arduino nano and uno