adafruit learning system

NeoPixel Ring Clock

Created by Becky Stern

Last updated on 2014-02-12 06:30:07 PM EST

Guide Contents

Overview

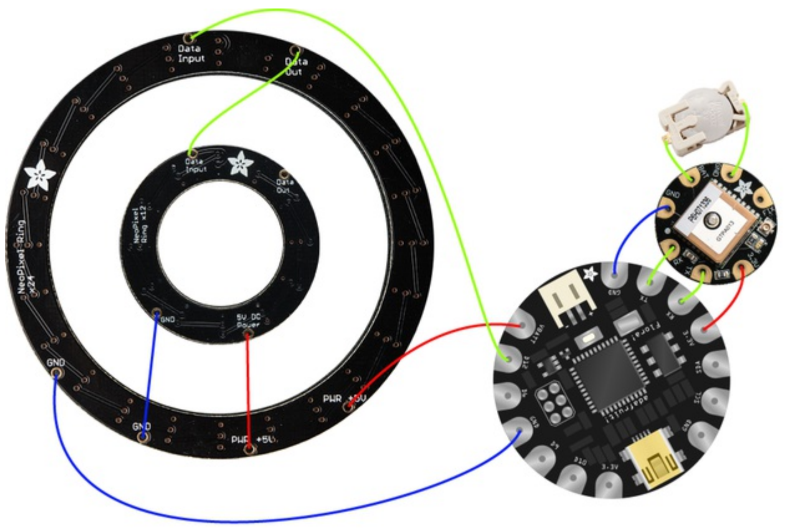
Build a clock with NeoPixel rings and FLORA! The FLORA GPS provides accurate timekeeping, and the clock's circular motif make it a handsome addition to your desk, wall, or even around your neck.

To build this project you will need:

- FLORA main board (http://adafru.it/659) and USB cable (http://adafru.it/260)
- FLORA GPS (http://adafru.it/1059)
- NeoPixel Ring 24 (http://adafru.it/1586)

- NeoPixel Ring 12 (http://adafru.it/1643)
- coincell battery holder (http://adafru.it/653) and battery (http://adafru.it/654)
- GPS antenna (http://adafru.it/960) with SMA to uFL adapter (http://adafru.it/851) (optional)
- soldering iron and solder (http://adafru.it/aTk)
- wire strippers (http://adafru.it/527)
- flush snips (http://adafru.it/152)
- pliers (http://adafru.it/146)
- multimeter (http://adafru.it/caH)
- solid core hookup wire (http://adafru.it/290)
- packing or painter's tape
- double sided tape
- third hand tool (http://adafru.it/291)

Circuit Diagram



NeoPixel rings connect as follows:

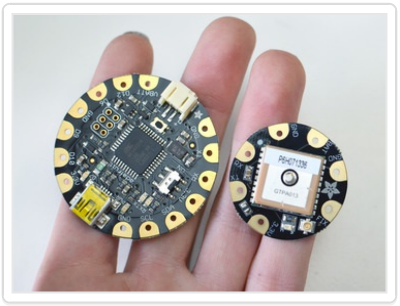
- +5V -> FLORA VBATT
- GND -> FLORA GND
- outer ring Data Input -> FLORA D12
- outer ring Data Out -> inner ring Data Input

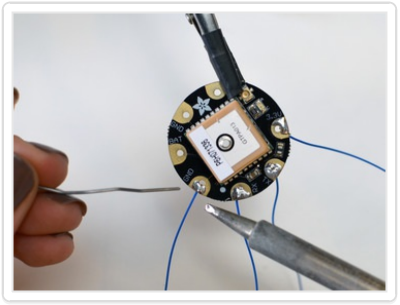
FLORA GPS connects as follows:

- GPS GND -> FLORA GND
- GPS 3.3V -> FLORA 3.3V
- GPS RX -> FLORA TX
- GPS TX -> FLORA RX
- GPS BAT -> positive side of coincell holder
- GPS GND -> negative side of coincell holder

power circuit via USB or battery connected to JST port

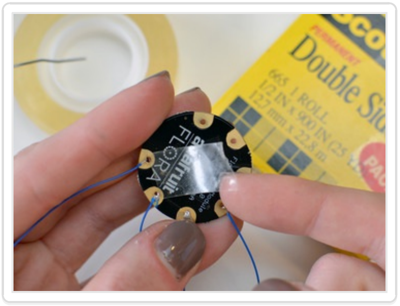
Build GPS Circuit

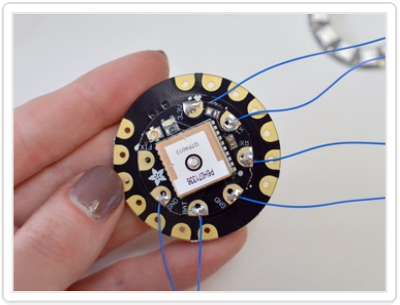




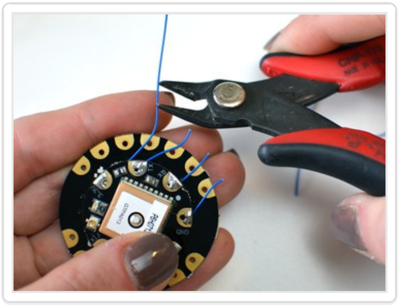
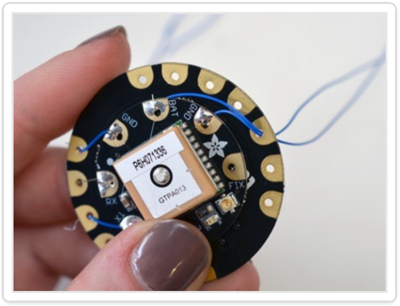
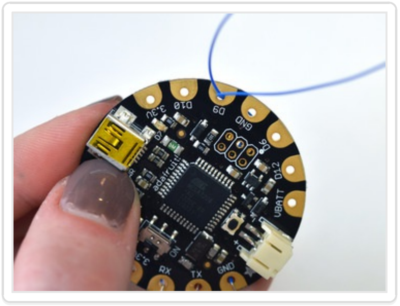
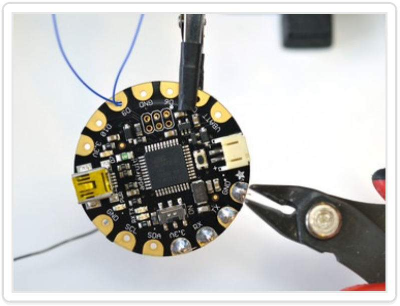
Line up your FLORA and GPS module according to the circuit diagram. You'll be attaching them back to back!

Solder wires onto the FLORA GPS' GND, TX, RX, 3.3V pins. Solder two longer wires to the GND and BAT pads on the FLORA GPS.



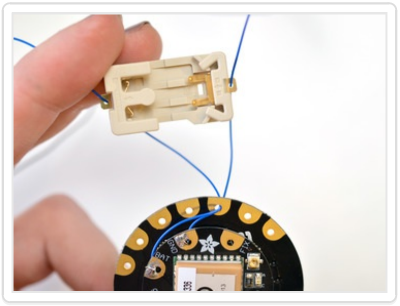
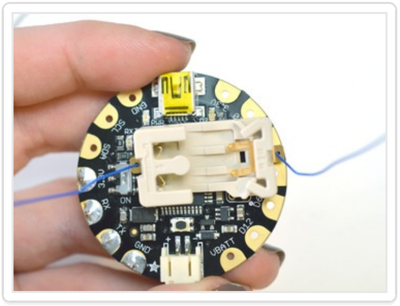
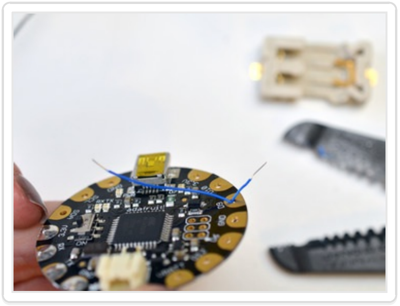
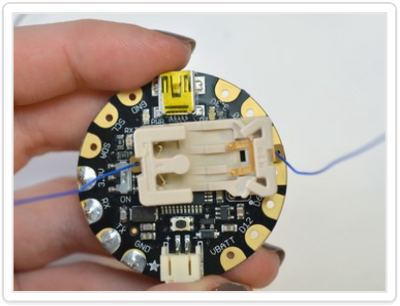
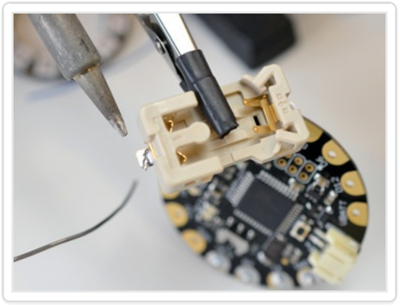


Use a piece of double-sided tape to position the GPS in the center back of FLORA, with the wires pointing towards the appropriate FLORA pads.

Clip wires to length, strip the ends, and solder the four connections to FLORA.

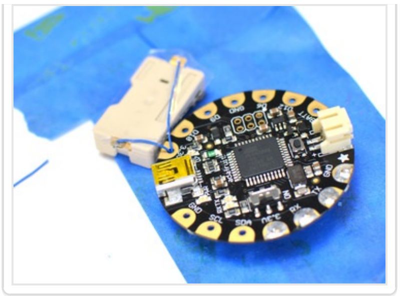
Route the two long BAT/GND wires through an unused hold on FLORA (like D9).

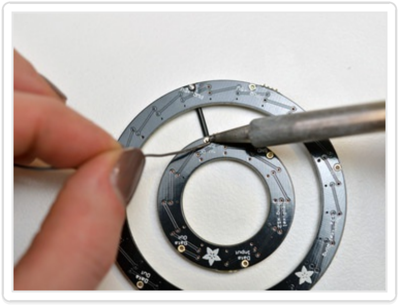
Thread on a coincell battery holder. BAT goes to + and GND goes to - on the holder. Adjust the wire lengths to position the battery holder on the FLORA as shown (we'll use foam tape to stick it down later).

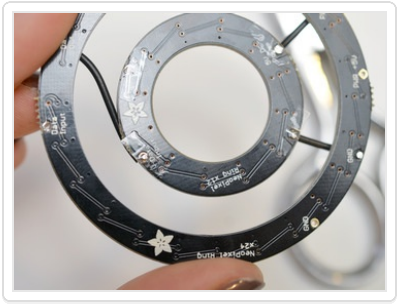
Solder these connections and insulate the back of the holder with tape.

Plug in the FLORA via USB and test the GPS circuit according to the FLORA GPS guide. (http://adafru.it/aRP)



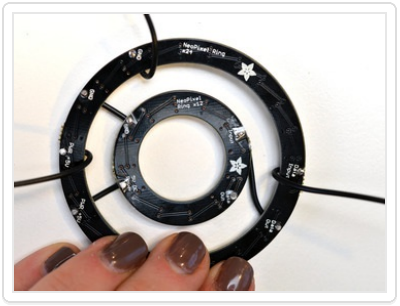
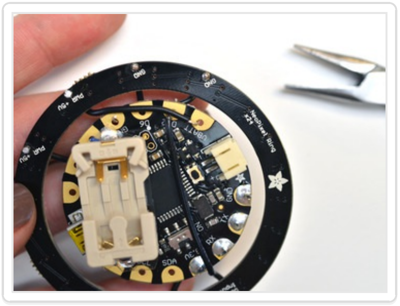
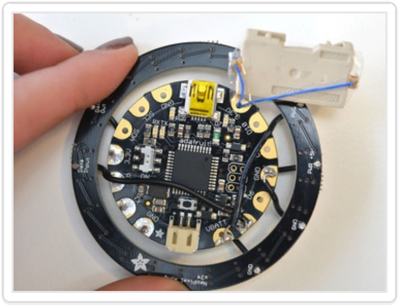
Build NeoPixel Ring Circuit





Arrange the two NeoPixel rings face down on your work surface. It can help to use tape, sticky side up, on your desk to help keep the rings aligned while you solder.

Make small pieces of solid-core wire shaped like staples to connect 5V, GND, and data between the two pixel rings according to the circuit diagram.

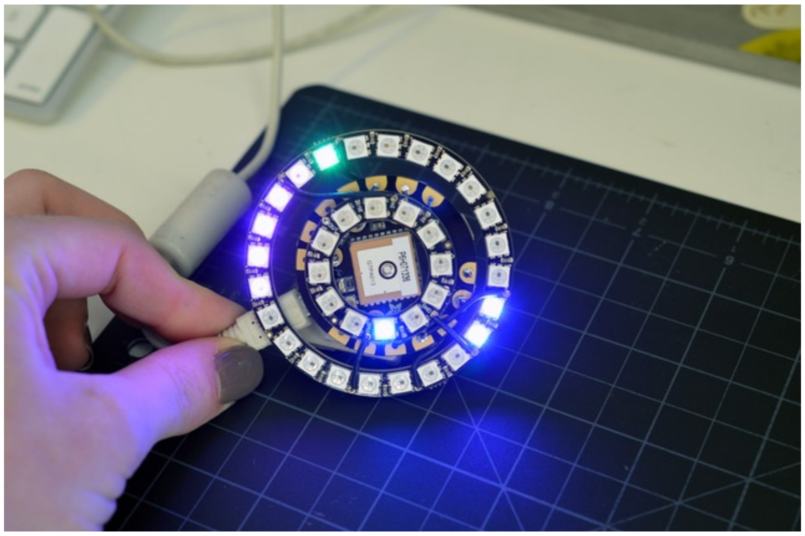




Solder longer wires as shown to the extra 5V and GND on the outer ring, as well as data input.

Place the GPS/FLORA circuit in the center and solder these longer wires to VBATT, D12, and GND on FLORA according to the circuit diagram.

Test the NeoPixel rings by downloading the NeoPixel (http://adafru.it/cEz) code library and running one of the test Arduino sketches provided.

Code and Clock Faces



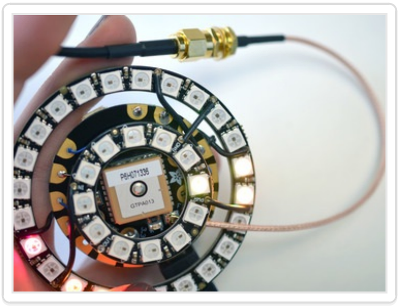
Animated clock based on zeroeth's time loop (http://adafru.it/d90), with added GPS and orientation adjustment:

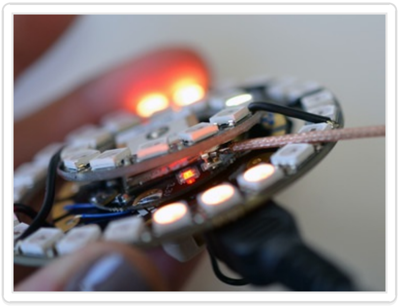
```
//NeoPixel clock with FLORA based on Kevin Alford's NeoPixel ring clock face
//https://github.com/zeroeth/time_loop
//http://www.youtube.com/watch?v=b-mROp-ZKuk
//modified by Becky Stern for Adafruit Industries
#include <Adafruit_NeoPixel.h>
#include <Time.h>
#include <Adafruit_GPS.h>
#include <SoftwareSerial.h>
#include <Wire.h>

Adafruit_GPS GPS(&Serial1);

#define PIN 12

// Set GPSECHO to 'false' to turn off echoing the GPS data to the Serial console
// Set to 'true' if you want to debug and listen to the raw GPS sentences
#define GPSECHO false

// this keeps track of whether we're using the interrupt
// off by default!
boolean usingInterrupt = false;

// From adaFruit NEOPIXEL goggles example: Gamma correction improves appearance of midrang
const uint8_t gamma[] = {
     0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,
     0,  0,  0,  0,  0,  0,  0,  0,  1,  1,  1,  1,  1,  1,  1,  1,
     1,  1,  1,  1,  2,  2,  2,  2,  2,  2,  2,  2,  3,  3,  3,  3,
     3,  3,  4,  4,  4,  4,  5,  5,  5,  5,  5,  6,  6,  6,  6,  7,
     7,  7,  8,  8,  8,  9,  9,  9, 10, 10, 10, 11, 11, 11, 12, 12,
    13, 13, 13, 14, 14, 15, 15, 16, 16, 17, 17, 18, 18, 19, 19, 20,
   20, 21, 21, 22, 22, 23, 24, 24, 25, 25, 26, 27, 27, 28, 29, 29,
   30, 31, 31, 32, 33, 34, 34, 35, 36, 37, 38, 38, 39, 40, 41, 42,
   42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57,
   58, 59, 60, 61, 62, 63, 64, 65, 66, 68, 69, 70, 71, 72, 73, 75,
   76, 77, 78, 80, 81, 82, 84, 85, 86, 88, 89, 90, 92, 93, 94, 96,
   97, 99,100,102,103,105,106,108,109,111,112,114,115,117,119,120,
  122,124,125,127,129,130,132,134,136,137,139,141,143,145,146,148,
  150,152,154,156,158,160,162,164,166,168,170,172,174,176,178,180,
  182,184,186,188,191,193,195,197,199,202,204,206,209,211,213,215,
  218,220,223,225,227,230,232,235,237,240,242,245,247,250,252,255
};

// Parameter 1 = number of pixels in strip
// Parameter 2 = pin number (most are valid)
// Parameter 3 = pixel type flags, add together as needed:
//   NEO_RGB     Pixels are wired for RGB bitstream
//   NEO_GRB     Pixels are wired for GRB bitstream
//   NEO_KHZ400  400 KHz bitstream (e.g. FLORA pixels)
//   NEO_KHZ800  800 KHz bitstream (e.g. High Density LED strip)
Adafruit_NeoPixel strip = Adafruit_NeoPixel(36, PIN, NEO_GRB + NEO_KHZ800);

//--------------------------------------------------|
//Your NeoPixel rings may not line up with ours, or each other.     |
//Enter which NeoPixel leds are on top LED (0-23 and 0-13).  |
  #define outerTOP_LED                10
  #define innerTOP_LED                5
//--------------------------------------------------|

int outertopLED = outerTOP_LED;
int innertopLED = innerTOP_LED;
float fLat = 0.0;
float fLon = 0.0;

//tequila sunrise color scheme

uint32_t milli_color  = strip.Color ( 44,  21,  0); // redest orange
uint32_t second_color = strip.Color (  44,  30, 0); //slightly yellower
uint32_t hour_color   = strip.Color (  44, 42,  0); //yellow
uint32_t minute_color = strip.Color ( 43, 0, 5); //red
uint32_t off_color    = strip.Color (  0,  0,  0);


//blue, green, & purple color scheme
/*
uint32_t milli_color  = strip.Color ( 24,  0,  24); // magenta
uint32_t second_color = strip.Color (  17,  0, 44); // purple
uint32_t hour_color   = strip.Color (  0, 10,  44); // royal blue
uint32_t minute_color = strip.Color ( 0, 44, 10); // green
uint32_t off_color    = strip.Color (  0,  0,  0);
*/

// Offset hours from gps time (UTC)
//const int offset = 1;   // Central European Time
const int offset = -4;  // Eastern Daylight Time (USA)
//const int offset = -5;  // Central Daylight Time (USA)
//const int offset = -8;  // Pacific Standard Time (USA)
//const int offset = -7;  // Pacific Daylight Time (USA)
```

```
/* CLOCK */
class ClockPositions
{
 public:

  uint8_t milli;
  uint8_t currentsecond;
  uint8_t currentminute;
  uint8_t currenthour;


  ClockPositions ();
  void update    ();
};


ClockPositions::ClockPositions()
{
  /*
  milli = 0;
  second = 0;
  minute = 30;
  hour = 6;
  */
  //DateTime(__DATE__, __TIME__);

}


void ClockPositions::update()
{

  if (GPS.fix) {
      // set the Time to the latest GPS reading
      setTime(GPS.hour, GPS.minute, GPS.seconds, GPS.day, GPS.month, GPS.year);
      delay(50);
      adjustTime(offset * SECS_PER_HOUR);

      delay(100);
    }

  currentsecond = outertopLED + map ((second() % 60), 0, 60, 0, 23);
  if (currentsecond > 24) {currentsecond = currentsecond-24;};
  milli  = map ((millis() %  1000), 0,  1000, 0, 24);
  currenthour   = innertopLED + map ((hour() % 12), 0,  12, 24, 35);
  if (currenthour > 35) { currenthour = currenthour-12;}
  //Serial.print("the hour is: ");
  //Serial.println(hour());
  currentminute = outertopLED + map (minute() % 60, 0,  60, 0, 23);
  if (currentminute > 23) {currentminute = currentminute-24;};
  //Serial.print("the minute is: ");
  //Serial.println(minute());
}



/* CLOCK VIEW */
```

```
class ClockSegments
{
 public:
  ClockPositions    &positions;
  Adafruit_NeoPixel &strip;

  ClockSegments (Adafruit_NeoPixel&, ClockPositions&);
  void draw  ();
  void clear ();
  void add_color (uint8_t position, uint32_t color);
  uint32_t blend (uint32_t color1, uint32_t color2);
};


ClockSegments::ClockSegments (Adafruit_NeoPixel& n_strip, ClockPositions& n_positions): strip (n_
{
}


void ClockSegments::draw()
{
  clear();

  add_color (positions.currentminute,  minute_color);
  //Serial.println(positions.minute);
  add_color (positions.currenthour,  hour_color  );


  add_color (positions.currentsecond     % 24, second_color);
  add_color ((positions.currentsecond+1) % 24, second_color);
  //add_color ((positions.second+2) % 24, second_color);

/*
  add_color (positions.milli     % 24,  milli_color);
  add_color ((positions.milli+1) % 24,  milli_color);
  add_color ((positions.milli+2) % 24,  milli_color);
*/
  strip.show ();
}


void ClockSegments::add_color (uint8_t position, uint32_t color)
{
  uint32_t blended_color = blend (strip.getPixelColor (position), color);

  /* Gamma mapping */
  uint8_t r,b,g;

  r = (uint8_t)(blended_color >> 16),
  g = (uint8_t)(blended_color >>  8),
  b = (uint8_t)(blended_color >>  0);

  strip.setPixelColor (position, blended_color);
}


uint32_t ClockSegments::blend (uint32_t color1, uint32_t color2)
{
  uint8_t r1,g1,b1;
  uint8_t r2,g2,b2;
  uint8_t r3,g3,b3;

  r1 = (uint8_t)(color1 >> 16),
  g1 = (uint8_t)(color1 >>  8),
  b1 = (uint8_t)(color1 >>  0);

  r2 = (uint8_t)(color2 >> 16),
  g2 = (uint8_t)(color2 >>  8),
  b2 = (uint8_t)(color2 >>  0);


  return strip.Color (constrain (r1+r2, 0, 255), constrain (g1+g2, 0, 255), constrain (b1+b2, 0, 255));
}


void ClockSegments::clear ()
{
  for(uint16_t i=0; i<strip.numPixels (); i++) {
      strip.setPixelColor (i, off_color);
  }
}




/* SIMPLE MIXER */
// add rgb and clamp






/* APP */
ClockPositions positions;
ClockSegments  segments(strip, positions);

void setup ()
{


  // connect at 115200 so we can read the GPS fast enough and echo without dropping chars
  // also spit it out
  Serial.begin(115200);
  Serial.println("Adafruit GPS library basic test!");

  // 9600 NMEA is the default baud rate for Adafruit MTK GPS's- some use 4800
  GPS.begin(9600);
  // uncomment this line to turn on RMC (recommended minimum) and GGA (fix data) including alt
  GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
  // uncomment this line to turn on only the "minimum recommended" data
  //GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCONLY);
  // For parsing data, we don't suggest using anything but either RMC only or RMC+GGA since
  // the parser doesn't care about other sentences at this time
  // Set the update rate
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ); // 1 Hz update rate
  // For the parsing code to work nicely and have time to sort thru the data, and
  // print it out we don't suggest using anything higher than 1 Hz

  // Request updates on antenna status, comment out to keep quiet
  GPS.sendCommand(PGCMD_ANTENNA);

  delay(1000);
  // Ask for firmware version
  Serial1.println(PMTK_Q_RELEASE);

  strip.begin ();
  strip.show (); // Initialize all pixels to 'off'
}

uint32_t gpsTimer = millis();
uint32_t startupTimer = millis();

void loop ()
{

  // read data from the GPS in the 'main loop'
  char c = GPS.read();
  // if you want to debug, this is a good time to do it!
  if (GPSECHO)
    //if (c) Serial.print(c);
  // if a sentence is received, we can check the checksum, parse it...
  if (GPS.newNMEAreceived()) {
    // a tricky thing here is if we print the NMEA sentence, or data
    // we end up not listening and catching other sentences!
    // so be very wary if using OUTPUT_ALLDATA and trytng to print out data
    //Serial.println(GPS.lastNMEA()); // this also sets the newNMEAreceived() flag to false
    if (!GPS.parse(GPS.lastNMEA())) // this also sets the newNMEAreceived() flag to false
      return; // we can fail to parse a sentence in which case we should just wait for another
  }

  // if millis() or timer wraps around, we'll just reset it
  if (gpsTimer > millis()) gpsTimer = millis();

    //if (GPS.fix) {
      // set the Time to the latest GPS reading
      setTime(GPS.hour, GPS.minute, GPS.seconds, GPS.day, GPS.month, GPS.year);
      delay(50);
      adjustTime(offset * SECS_PER_HOUR);
    Serial.print("\nTime: ");
    Serial.print(GPS.hour, DEC); Serial.print(':');
    Serial.print(GPS.minute, DEC); Serial.print(':');
    Serial.print(GPS.seconds, DEC); Serial.print('.');
    Serial.println(GPS.milliseconds);
      delay(100);
    //}

  // approximately every 60 seconds or so, update time
  if ((millis() - gpsTimer > 60000)) {
    gpsTimer = millis(); // reset the timer
    if (GPS.fix) {
      // set the Time to the latest GPS reading
      setTime(GPS.hour, GPS.minute, GPS.seconds, GPS.day, GPS.month, GPS.year);
      delay(50);
      adjustTime(offset * SECS_PER_HOUR);
      delay(100);
    }
  }

  positions.update ();
  segments.draw ();

}




// Fill the dots one after the other with a color
void colorWipe(uint32_t c, uint32_t wait) {
  for(uint16_t i=0; i<strip.numPixels(); i++) {
      strip.setPixelColor(i, c);
      strip.show();
      delay(wait);
  }
}

// Slightly different, this makes the rainbow equally distributed throughout
void rainbowCycle(uint8_t wait) {
  uint16_t i, j;

  for(j=0; j<256*5; j++) { // 5 cycles of all colors on wheel
    for(i=0; i< strip.numPixels(); i++) {
      strip.setPixelColor(i, Wheel(((i * 256 / strip.numPixels()) + j) & 255));
    }
    strip.show();
    delay(wait);
  }
}

// Input a value 0 to 255 to get a color value.
// The colours are a transition r - g - b - back to r.
uint32_t Wheel(byte WheelPos) {
  if(WheelPos < 85) {
   return strip.Color(WheelPos * 3, 255 - WheelPos * 3, 0);
  } else if(WheelPos < 170) {
   WheelPos -= 85;
   return strip.Color(255 - WheelPos * 3, 0, WheelPos * 3);
  } else {
   WheelPos -= 170;
   return strip.Color(0, WheelPos * 3, 255 - WheelPos * 3);
  }
}
```

Use it!





To set the time, make sure the GPS has a clear view of the sky, or attach a GPS antenna and stick it out the window.

Use a piece of wire to make a stand for your desk, hang your clock on the wall, or attach a big 'ol gold chain and jam out like 21st century Flavor Flav.