#include <TinyGPS++.h>

#include <SoftwareSerial.h>

/\*

 This sample code demonstrates the normal use of a TinyGPS++ (TinyGPSPlus) object.

 It requires the use of SoftwareSerial, and assumes that you have a

 4800-baud serial GPS device hooked up on pins 4(rx) and 3(tx).

\*/

static const int RXPin = 4, TXPin = 3;

static const uint32\_t GPSBaud = 9600;

// The TinyGPS++ object

TinyGPSPlus gps;

// The serial connection to the GPS device

SoftwareSerial ss(RXPin, TXPin);

void setup()

{

 Serial.begin(115200);

 ss.begin(GPSBaud);

 Serial.println(F("FullExample.ino"));

 Serial.println(F("An extensive example of many interesting TinyGPS++ features"));

 Serial.print(F("Testing TinyGPS++ library v. ")); Serial.println(TinyGPSPlus::libraryVersion());

 Serial.println(F("by Mikal Hart"));

 Serial.println();

 Serial.println(F("Sats HDOP Latitude Longitude Fix Date Time Date Alt Course Speed Card Distance Course Card Chars Sentences Checksum"));

 Serial.println(F(" (deg) (deg) Age Age (m) --- from GPS ---- ---- to London ---- RX RX Fail"));

 Serial.println(F("---------------------------------------------------------------------------------------------------------------------------------------"));

}

void loop()

{

 static const double LONDON\_LAT = 51.508131, LONDON\_LON = -0.128002;

 printInt(gps.satellites.value(), gps.satellites.isValid(), 5);

 printInt(gps.hdop.value(), gps.hdop.isValid(), 5);

 printFloat(gps.location.lat(), gps.location.isValid(), 11, 6);

 printFloat(gps.location.lng(), gps.location.isValid(), 12, 6);

 printInt(gps.location.age(), gps.location.isValid(), 5);

 printDateTime(gps.date, gps.time);

 printFloat(gps.altitude.meters(), gps.altitude.isValid(), 7, 2);

 printFloat(gps.course.deg(), gps.course.isValid(), 7, 2);

 printFloat(gps.speed.kmph(), gps.speed.isValid(), 6, 2);

 printStr(gps.course.isValid() ? TinyGPSPlus::cardinal(gps.course.value()) : "\*\*\* ", 6);

 unsigned long distanceKmToLondon =

 (unsigned long)TinyGPSPlus::distanceBetween(

 gps.location.lat(),

 gps.location.lng(),

 LONDON\_LAT,

 LONDON\_LON) / 1000;

 printInt(distanceKmToLondon, gps.location.isValid(), 9);

 double courseToLondon =

 TinyGPSPlus::courseTo(

 gps.location.lat(),

 gps.location.lng(),

 LONDON\_LAT,

 LONDON\_LON);

 printFloat(courseToLondon, gps.location.isValid(), 7, 2);

 const char \*cardinalToLondon = TinyGPSPlus::cardinal(courseToLondon);

 printStr(gps.location.isValid() ? cardinalToLondon : "\*\*\* ", 6);

 printInt(gps.charsProcessed(), true, 6);

 printInt(gps.sentencesWithFix(), true, 10);

 printInt(gps.failedChecksum(), true, 9);

 Serial.println();

 smartDelay(10000);

 if (millis() > 5000 && gps.charsProcessed() < 10)

 Serial.println(F("No GPS data received: check wiring"));

}

// This custom version of delay() ensures that the gps object

// is being "fed".

static void smartDelay(unsigned long ms)

{

 unsigned long start = millis();

 do

 {

 while (ss.available())

 gps.encode(ss.read());

 } while (millis() - start < ms);

}

static void printFloat(float val, bool valid, int len, int prec)

{

 if (!valid)

 {

 while (len-- > 1)

 Serial.print('\*');

 Serial.print(' ');

 }

 else

 {

 Serial.print(val, prec);

 int vi = abs((int)val);

 int flen = prec + (val < 0.0 ? 2 : 1); // . and -

 flen += vi >= 1000 ? 4 : vi >= 100 ? 3 : vi >= 10 ? 2 : 1;

 for (int i=flen; i<len; ++i)

 Serial.print(' ');

 }

 smartDelay(0);

}

static void printInt(unsigned long val, bool valid, int len)

{

 char sz[32] = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*";

 if (valid)

 sprintf(sz, "%ld", val);

 sz[len] = 0;

 for (int i=strlen(sz); i<len; ++i)

 sz[i] = ' ';

 if (len > 0)

 sz[len-1] = ' ';

 Serial.print(sz);

 smartDelay(0);

}

static void printDateTime(TinyGPSDate &d, TinyGPSTime &t)

{

 if (!d.isValid())

 {

 Serial.print(F("\*\*\*\*\*\*\*\*\*\* "));

 }

 else

 {

 char sz[32];

 sprintf(sz, "%02d/%02d/%02d ", d.month(), d.day(), d.year());

 Serial.print(sz);

 }

 if (!t.isValid())

 {

 Serial.print(F("\*\*\*\*\*\*\*\* "));

 }

 else

 {

 char sz[32];

 sprintf(sz, "%02d:%02d:%02d ", t.hour(), t.minute(), t.second());

 Serial.print(sz);

 }

 printInt(d.age(), d.isValid(), 5);

 smartDelay(0);

}

static void printStr(const char \*str, int len)

{

 int slen = strlen(str);

 for (int i=0; i<len; ++i)

 Serial.print(i<slen ? str[i] : ' ');

 smartDelay(0);

}