|  |
| --- |
| Sciences numériques et technologieFiche ressource |

* Thème : Localisation, cartographie et mobilité (extrait du référentiel)

|  |  |
| --- | --- |
| Contenus | Capacités attendues |
| GPS, Galileo | Décrire le principe de fonctionnement de la géolocalisation. |
| Protocole NMEA | Décoder une trame NMEA pour trouver des coordonnées géographiques. |
| Confidentialité | Régler les paramètres de confidentialité d’un téléphone pour partager ou non sa position. |
| Activités |
| Connecter un récepteur GPS sur un ordinateur afin de récupérer la trame NMEA, en extraire la localisation. Situer sur une carte numérique la position récupérée. |

* **Prérequis et situation**
* Plutôt en fin d’année
* Langage python : Types, fonctions, listes
* Un outil de création de frise chronologique (sur l’ENT ou bien <http://thetimelineproj.sourceforge.net/> par exemple)

**Repères historiques**

**Durée estimée : 1H30**

**Définition de la géolocalisation**

D'après Larousse : *Technique de détermination de la situation géographique précise d'un lieu ou, à un instant donné, d'une personne, d'un véhicule, d'un objet, etc.*

Les applications liées à la géolocalisation sont devenues incontournables et ont remplacé les cartes, les boussoles et autres sextants. Avec l'arrivée des smartphones et le GPS intégré, notre position est identifiable et nous pouvons nous repérer presque partout dans le monde. De la même manière, la miniaturisation des composants électroniques a permis d'insérer des modules GPS sur de nombreux objets nous donnant leur position en temps réel.

Gardons bien en mémoire que les données en provenance d'un traceur GPS peuvent être utilisées par n'importe qui et que les données personnelles doivent être protégées.

Q1. Visionner [la vidéo](https://www.youtube.com/watch?v=ez1xZWpY9NE) (https://www.youtube.com/watch?v=ez1xZWpY9NE), chercher sur internet et associer les dates aux éléments ci-dessous :

Lancement du GPS, système américain □ □ 1973

Lancement de Galileo, système européen □ □ 1976

Lancement de Glonass, système soviétique □ □ 1993

Lancement de QZSS, système japonais □ □ 1999

Lancement de IRNSS, système indien □ □ 2002

Lancement de Beidou, système chinois □ □ 2006

Q2. Réaliser, sur l’ENT ou autre logiciel (<http://thetimelineproj.sourceforge.net/> par exemple), une frise chronologique avec les évènements précédents. On ajoutera la sortie de Google Earth et l’arrivée de OpenStreetMap.

**Etude préliminaire**

**Durée estimée : 1H30**

Le principal instrument de localisation, GPS (Global Positioning System), a été conçu par l’armée américaine dans les années soixante. Le premier satellite GPS fut lancé en 1978. Il y en a actuellement une trentaine, de sorte qu’à tout moment quatre à six satellites au moins sont visibles depuis tout point de la Terre. Couplé aux cartes numériques, le système GPS permet de se situer.

Figure 1

Il faut au minimum 3 satellites pour avoir une localisation en 2 dimensions à la surface de la Terre et 4 satellites pour accéder à l'altitude.

Pour bien comprendre ce phénomène de triangulation, prenons l’exemple simplifié suivant :

Le messager du roi de la tour 1 doit se rendre à la tour 3 au travers une forêt dense. Pour l’aider à s’orienter, car il ne dispose d’aucun appareil, les tours 1, 2 et 4 vont tirer un coup de canon toutes les heures, à 2 min d’intervalle. La première tour tire à 10 h et le messager, à l’écoute tout au long de son parcours, entend la détonation à 10 h et 30 s. Il entend la détonation de la tour 2 à 10 h 2 min et 35 s et celle de la tour 4 à 10 h 4 min et 40 s.

Q3. La vitesse du son étant de 340 m.s-1, situer notre messager dans le plan ci-dessous. Sachant que notre messager est parti de la tour 1, doit-il continuer tout droit ?



Figure 2

En réalité, nos appareils GPS utilisent les ondes radios et calculent le temps que met un signal pour faire un aller et retour vers un satellite <http://www.universcience.tv/video-galileo-fonctionnement-du-gps-europeen-11461.html>.

Les récepteurs utilisent la norme NMEA (National Marine Electronics Association) pour communiquer.

Il existe plus d'une trentaine de trames GPS différentes.

Exemple de trame : $GPGGA, 123036.000, 4850.8593, N, 00220.6853, E, 1, 9, 0.95, 68.7, M, 47.3, M, \*6F

Le type d'équipement est défini par les deux caractères qui suivent le $. Le type de trame est défini par les caractères suivants jusqu'à la virgule.

La trame GGA est très courante car elle fait partie de celles qui sont utilisées pour connaître la position courante du récepteur GPS. Nous limiterons notre étude à cette trame.

$GxGGA : Type de trame (x = A : GALLILEO ; P : GPS ; L : GLONASS ; B : BEIDOU ; N : mixte P + L)

123036.000 : Trame envoyée à 12h30m36,000s (heure UTC)

4850.8593 : Latitude 48.8476° (48+50.8593/60) décimaux ou 48°50'51.56" (48+50+0.8593\*60)

N : Nord

00220.6853 : Longitude 2.3447° décimaux (002+20.6853/60) ou 2°20'41.12" (002+20+0.6853\*60)

E : Est

1 : Type de positionnement (le 1 est un positionnement GPS)

9 : Nombre de satellites utilisés pour calculer les coordonnées

0.95 : Précision horizontale ou HDOP (Horizontal dilution of precision)

68.7 : Altitude

M : en mètres

47.3 : Correction de la hauteur du géoïde

M : en mètres

,,,,,0000 : D'autres informations peuvent être inscrites dans ces champs\*

6F : Somme de contrôle de parité



Q4. Télécharger l’application « NMEA Tools » sur votre smartphone (Android) et vérifier, dans les paramètres de vos applications et notifications, si elle est autorisée à utiliser le GPS pour obtenir sa position et à stocker des informations.

Q5. Ouvrir l’application puis cliquer sur « NMEA Enregistreur » puis démarrer l’enregistrement des trames. Vous pouvez arrêter dès que la position est stabilisée et donner un nom à votre fichier. Revenir au menu principal puis cliquer sur « NMEA Viewer », sélectionner votre fichier et relever la dernière trame GGA du fichier.

QRcode NMEA Android

PS : Le professeur peut faire la démonstration si les élèves n’ont pas les moyens matériels.

Q6. Déduire de la trame précédente, les coordonnées GPS DMS (degrés, minutes, secondes) et décimales.

**Activité pratique 1**

**Durée estimée : 30 min**

Il s’agit de localiser sur une carte de France les coordonnées relevées précédemment. Nous utilisons le langage Python avec la bibliothèque « folium ».

Les fonctions nécessaires sont les suivantes :

* *m = folium.Map(location=[45.5236, -122.6750])* : pour créer une carte "m" centrée sur les coordonnées GPS. On peut ajouter l'option zoom\_start=12 pour obtenir un zoom particulier.
* *folium.Marker([45.5236, -122.6750],popup="lieu recherché").add\_to(m)* : pour ajouter un marqueur et un commentaire.
* *m* : pour afficher la carte

Q7. Implanter le programme permettant de visualiser sur une carte les coordonnées obtenues à la question 6.

Pour obtenir l’adresse correspondante aux coordonnées saisies, nous pouvons utiliser la bibliothèque « geopy.geocoders ». Le programme est le suivant :

*from geopy.geocoders import Nominatim*

*geocoder = Nominatim(user\_agent="my-application")*

*location = geocoder.reverse([lat, long])*

*print(location.address)*

Ou encore, connaitre les coordonnées géographiques :

*location = geocoder.geocode("123 rue Saint Jacques 75005 Paris")*

*print((location.latitude, location.longitude))*

Q8. Compléter le programme précédent afin d’obtenir l’adresse.

**Activité pratique 2**

**Durée estimée : 1H**

Il s’agit d’obtenir les coordonnées GPS grâce à un module relié à l’ordinateur, c’est le principe des objets connectés IOT.

Une carte Arduino reliée à un capteur GPS et à un PC (USB : COMx) envoie régulièrement la trame NMEA au PC :



Figure 3

La carte Arduino est déjà initialisée avec un programme qui envoie sans interruption, des trames NMEA au PC via la liaison série. Le programme en langage Python, doit réceptionner la trame contenant « GGA », en extraire les coordonnées et les afficher sur la carte de France. On utilise la bibliothèque « Serial » pour communiquer et les fonctions suivantes :

* *ser = serial.****Serial(****"COMx",timeout=1****)*** pour initialiser la liaison série à 9600 bauds sur le COMx (x : numéro du port de communication utilisé par la carte Arduino)
* *mes = str(ser.****readline()****)* pour lire une ligne

Nous pouvons tester si le « message » contient une chaine de caractères "GGA" ainsi que "N", "S", "E" ou "O" avec l’instruction alternative :

* *if "GGA" in pos and ("N" or "E" or "S" or "O") in pos:*

Q9. Compléter le programme qui permet de lire la trame NMEA GxGGA en provenance de la carte Arduino et de l’afficher dans la console. On donne l’algorithme suivant :

*Entrées : Donnés numériques sur liaison USB*

*Sortie : pos (chaine contenant la trame NMEA)*

*Début*

 *Initialiser la liaison série*

 *pos 🡨 ""*

 *trame 🡨 False*

 *Tant que trame = False Faire*

 *pos 🡨 lire liaison série*

 *Si pos contient « GGA » et « N », « S », « E » ou « O » Alors*

 *trame 🡨 True*

 *Sinon*

 *pos 🡨 ""*

 *Fin de Si*

 *Fin de Tant que*

 *Afficher pos*

 *Fermer la liaison série*

*Fin*

import serial

try:

 ser = serial.Serial("COMxx", 9600, timeout=1) #xx à compléter

 print(ser)

except:

 print("Pas de liaison avec la carte Arduino !")

else:

 pos = ""

 trame=False

**A compléter !**

 ser.close()

Q6 Compléter votre programme afin de localiser votre emplacement sur la carte de France.

La chaine de caractères précédente (pos) devra être convertie en liste (lpos) pour en extraire plus facilement la latitude et la longitude.

$GPGGA,123036.000,4850.8593,N,00220.6853,E,1,9,0.95,68.7,M,47.3,M,\*6F

Lpos[4]

Lpos[2]

**Annexe :**

**code de la carte Arduino**

#include <SoftwareSerial.h>

SoftwareSerial SoftSerial(2, 3); //Shield GPS sur les broches 2 et 3

unsigned char buffer[64]; //réservation mémoire

int count=0;

void setup()

{

 SoftSerial.begin(9600); //vitesse de transmission 9600 bauds

 Serial.begin(9600);

}

void loop()

{

 if (SoftSerial.available())

 {

 while(SoftSerial.available())

 {

 buffer[count++]=SoftSerial.read();

 if (count == 64) break;

 }

 Serial.write(buffer,count);

 clearBufferArray();

 count = 0;

 }

 if (Serial.available())

 {

 SoftSerial.write(Serial.read());

 }

}

void clearBufferArray()

{

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

 {

 buffer[i]=NULL;

 }

}