|  |  |  |  |
| --- | --- | --- | --- |
| **ACADÉMIE de VERSAILLES** | *Proposition SNT* | |  |
|  | [*chdebernardi.fr*](http://chdebernardi.fr/) | *Seconde* | *4H30* |
| TD /TP *Autour de la gestion des données EXIF* | | Version 1.02 |
| *Lycée RICHELIEU – 64 rue George SAND – 92500 RUEIL MALMAISON* | | | |

*L’objectif de cette séquence est d’exploiter les informations « cachées » dans une image prise à l’aide d’un téléphone portable récent. Elle se compose principalement de petites manipulations et donc d’une suite de petits scripts en langage Python. L’objectif secondaire est d’acquérir quelques bases dans ce langage et d’appréhender le foisonnement des librairies sous Python en s’intéressant principalement aux dépôts sur Github et aux gestionnaires pip et conda.*

*------- Il faut afficher les marques pour lire ce document dans son intégralité -------*

**Note pour les enseignants :** La plupart des manipulations informatiques sont réalisées à partir de l’IDE Spyder dans un environnement sous Windows obtenu à partir de l’installation de ANACONDA disponible à l’adresse <https://www.anaconda.com/download/>.



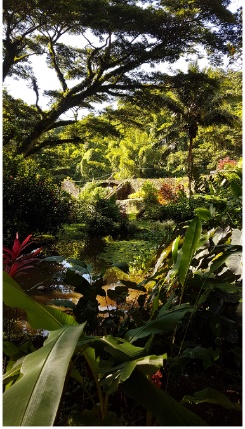
D’autres environnements de développement sont possibles.

## Activité 1 –Acquisition d’une image

**Note pour les enseignants :** L’objectif de cette activité est de faire récupérer une image ou une photo aux élèves par groupe de deux ou quatre suivant la configuration de la salle. Cette photo peut provenir de leur téléphone portable ou provenir d’un réseau social. (Note : Si elle vient d’un réseau social, il est possible/probable que les metadata soient altérées.)

1. A l’aide de votre téléphone portable, photographier un objet de la salle de TP ou choisir une image dans votre librairie d’image. Faire valider la photo par votre enseignant et insérer la dans le cadre ci-dessous. Préciser son nom.

**Note de correction :** On obtient :

Insertion de l’image OuEstCe4.jpg

* Sauvegarder l’image dans un répertoire portant votre nom.

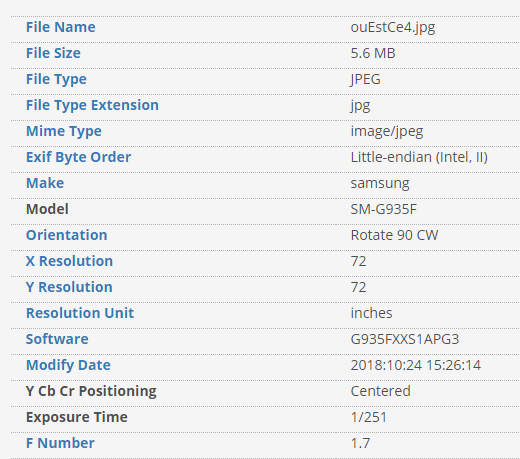
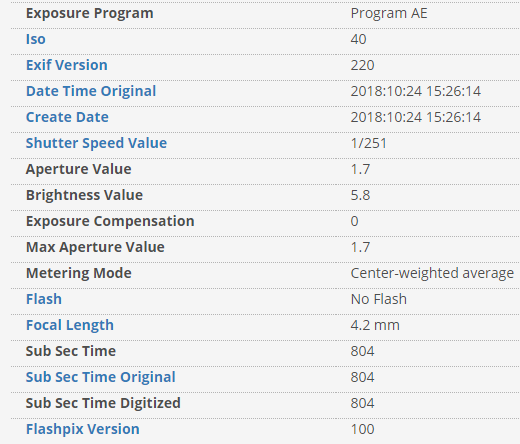
**Activité pour les enseignants :** Il est possible de vérifier que la photo ou l’image proposée par l’élève ou le groupe d’élèves contient des données de géolocalisation et d’horodatage à partir d’un site WEB comme :

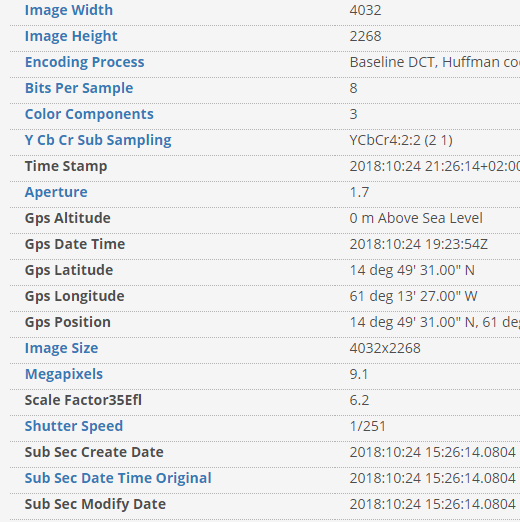
<https://www.get-metadata.com/>

<http://exif.regex.info/exif.cgi>

<https://sno.phy.queensu.ca/~phil/exiftool/>

Ou à partir de modules supplémentaires que l’on peut rajouter à un navigateur WEB.



Format EXIF : **Exchangeable image file format** ou **EXIF**

<https://fr.wikipedia.org/wiki/Exchangeable_image_file_format>

## Activité 2 – Affichage de l’image et récupération des informations globales de l’image sous Python

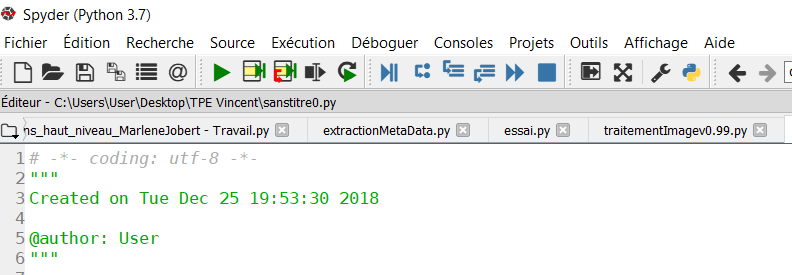
**Note pour les enseignants :** Vérifier que la librairie Pillow est présente dans le répertoire de Python ou l’installer à partir de



Sous Anaconda Prompt en mode administrateur.

### Affichage de l’image

* Lancer l’interface de programmation intégrée SPYDER (à partir d’ANACONDA) ou un autre ensemble de programmation. Attention, le lancement peut être très long (plusieurs minutes)



* Taper le programme ci-dessous :

from PIL import Image as im

nomFichierImage = "ouEstCe4.jpg" « Mettre le nom de votre image entre guillemets »

imageExif=im.open(nomFichierImage) # Ouverture du fichier

imageExif.show() # Affichage du fichier

print ('--------------------')

print(imageExif.size) # Attribut de taille : (largeur, hauteur)

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme en cliquant sur la flèche verte.

1. Que fait ce programme ? : *Il ouvre et affiche l’image*
2. Que signifie from PIL import Image as im ? : *Insertion de la classe image de la librairie PIL avec un alias.*
3. Que signifie imageExif=im.open(nomFichierImage) ? *Ouverture du fichier relatif à l’image et création d’un objet permettant de la manipuler.*
4. Que signifie imageExif.show() ? *Affichage de l’image.*
5. Que signifie print(imageExif.size)  ? *Recherche et affichage des attributs de taille de l’image.*

### Activités en autonomie

1. Sur le même principe que la dernière ligne de code du programme précédent, faire afficher les attributs de format et de mode.
2. L’image choisie est-elle de type JPG ou TIFF ? Quel attribut permet-il de le savoir ?
3. L’image choisie est-elle en couleur ? Quel attribut permet-il de le savoir ?

**Note de correction :** Il suffit de rajouter

print(imageExif.format) # Attribut de format (JPEG par exemple)

print(imageExif.mode) # Attribut de mode (RGB par exemple)

* Enregistrer votre programme sous même nom que précédemment.

## Activité 3 – Affichage brut des données EXIF

**Note pour les enseignants :** L’objectif de cette activité est de faire afficher les données EXIF sans aucun traitement et donc de montrer à l’élève que ces données sont inutilisables en l’état.

Il est possible de faire une introduction à JSON qui est un format de stockage à l’aide de balises (mais ce n’est pas obligatoire à mon avis).

* A la suite de votre programme précédent, rajouter les lignes de code ci-dessous :

print ('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

print(imageExif.info) # Attribut d'information

print ('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme

1. Que fait cette nouvelle ligne de code ? *Elle affiche les métadata en brut sans aucun traitement.*
2. Les données affichées sont-elles directement utilisables ? *Non*
3. Que peut on en conclure si on veut utiliser ces données. *Il faut traiter les données par une fonction de mise en forme.*

**Note de correction :** On obtient (affichage partiel)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

{'exif': b'Exif\x00\x00II\*\x00\x08\x00\x00\x00\x0b\x00\x0f\x01\x02\x00\x08\x00\x00\x00\xa2\x00\x00\x00\x10\x01\x02\x00\t\x00\x00\x00\xaa\x00\x00\x00\x12\x01\x03\x00\x01\x00\x00\x00\x06\x00\x00\x00\x1a\x01\x05\x00\x01\x00\x00\x00\x92\x00\x00\x00\x1b\x01\x05\x00\x01\x00\x00\x00\x9a\x00\x00\x00(\x01\x03\x00\x01\x00\x00\x00\x02\x00\x00\x001\x01\x02\x00\x0e\x00\x00\x00\xb4\x00\x00\x002\x01\x02\x00\x14\x00\x00\x00\xc2\x00\x00\x00\x13\x02\x03\x00\x01\x00\x00\x00\x01\x00\x00\x00i\x87\x04\x00\x01\x00\x00\x00\xd6\x00\x00\x00%\x88\x04\x00\x01\x00\x00\x00f\x03\x00\x004\x04\x00\x00H\x00\x00\x00\x01\x00\x00\x00H\x00\x00\x00\x01\x00\x00\x00samsung\x00SM-G935F\x00\x00G935FXXS1APG3\x002018:12:20 15:33:26\x00\x1e\x00\x9a\x82\x05\x00\x01\x00\x00\x00D\x02\x00\x00\x9d\x82\x05\x00\x01\x00\x00\x00L\x02\x00\x00"\x88\x03\x00\x01\x00\x00\x00\x02\x00\x00\x00\'\x88\x03\x00\x01\x00\x00\x00}\x00\x00\x00\x00\x90\x07\x00\x04\x00\x00\x000220\x03\x90\x02\x00\x14\x00\x00\x00T\x02\x00\x00\x04\x90\x02\x00\x14\

………..

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### Fin de la séance 1

## Activité 4 – Récupération et Exploitation des données EXIF

**Note pour les enseignants :** L’objectif de cette activité est de faire afficher les données EXIF avec un traitement permettant au moins de les rendre visibles et compréhensibles.

**Note pour les enseignants :** Vérifier que la librairie piexif est présente dans le répertoire de Python ou l’installer à partir de :



Sous Anaconda Prompt en mode administrateur.

### Affichage des metadonnées EXIF de l’image

* A la suite de votre programme précédent, rajouter les lignes de code ci-dessous :

# Affichage de toutes les données contenues dans les EXIF

import piexif as piex # ATTENTION JPEG ou TIFF exclusivement

exifDict = piex.load(nomFichierImage) # Chargement du fichier image

for ifd in ("0th", "Exif", "GPS", "1st"): # Attention il s'agit d'un zéro

for tag in exifDict[ifd]:

print(piex.TAGS[ifd][tag]["name"], exifDict[ifd][tag])

print ('////////////////////')

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme

### Affichage des metadatas

1. Que signifie import piexif as piex ? : *Insertion de la classe piexif avec un alias.*
2. Que signifie exifDict = piex.load(nomFichierImage) ? *Ouverture du fichier image et création d’un objet permettant de la manipuler pour en extraire les metadatas.*

Note : Les lignes de code ci-dessous sont plus difficiles à appréhender. On peut se limiter à une approche un peu superficielle.

for ifd in ("0th", "Exif", "GPS", "1st"):

for tag in exifDict[ifd]:

print(piex.TAGS[ifd][tag]["name"], exifDict[ifd][tag])

1. Combien y a-t-il de boucles imbriquées ? *Il y a deux boucles imbriquées.*
2. La première boucle se fait-elle sur des valeurs numériques ? *Non, il s’agit d’une boucle réalisée à partir d’une liste de données.*
3. Que réalise print(piex.TAGS[ifd][tag]["name"], exifDict[ifd][tag])  ? *Il affiche les noms des metadata avec leurs valeurs associées.*

**Note de correction :** On obtient (affichage partiel)

Make b'samsung'

Model b'SM-G935F'

Orientation 6

XResolution (72, 1)

YResolution (72, 1)

ResolutionUnit 2

Software b'G935FXXS1APG3'

DateTime b'2018:10:24 15:26:14'

YCbCrPositioning 1

ExifTag 214

GPSTag 870

ExposureTime (1, 251)

FNumber (17, 10)

ExposureProgram 2

ISOSpeedRatings 40

ExifVersion b'0220'

DateTimeOriginal b'2018:10:24 15:26:14'

DateTimeDigitized b'2018:10:24 15:26:14'

ShutterSpeedValue (797, 100)

ApertureValue (153, 100)

BrightnessValue (580, 100)

ExposureBiasValue (0, 10)

MaxApertureValue (153, 100)

MeteringMode 2

Flash 0

FocalLength (420, 100)

MakerNote b'\x07\x00\x01\x00\x07\x00\x04\x00\x00\x000100\x02\x00\x04\x00\x01\x00\x00\x00\x00 \x01\x00\x0c\x00\x04\x00\x01\x00\x00\x00\x00\x00\x00\x00\x10\x00\x05\x00\x01\x00\x00\x00Z\x00\x00\x00@\x00\x04\x00\x01\x00\x00\x00\x00\x00\x00\x00P\x00\x04\x00\x01\x00\x00\x00\x01\x00\x00\x00\x00\x01\x03\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'

UserComment b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'

SubSecTime b'0804'

SubSecTimeOriginal b'0804'

SubSecTimeDigitized b'0804'

………..

SceneCaptureType 0

ImageUniqueID b'C12LLJB18SM C12LLJE03GM\n'

GPSVersionID (2, 2, 0, 0)

GPSLatitudeRef b'N'

GPSLatitude ((14, 1), (49, 1), (31, 1))

GPSLongitudeRef b'W'

GPSLongitude ((61, 1), (13, 1), (27, 1))

GPSAltitudeRef 1

GPSAltitude (0, 1)

GPSTimeStamp ((19, 1), (23, 1), (54, 1))

GPSDateStamp b'2018:10:24'

ImageWidth 512

ImageLength 288

Compression 6

Orientation 6

XResolution (72, 1)

YResolution (72, 1)

ResolutionUnit 2

JPEGInterchangeFormat 1206

JPEGInterchangeFormatLength 33626

1. Les données GPS sont-elles présentes dans ces informations ? *Oui, on peut les voir dans les données encadrées.*
2. Compléter les données GPS dans le cadre ci-dessous ?

**Note de correction :** On obtient

GPSVersionID (2, 2, 0, 0)

GPSLatitudeRef b'N'

GPSLatitude ((14, 1), (49, 1), (31, 1))

GPSLongitudeRef b'W'

GPSLongitude ((61, 1), (13, 1), (27, 1))

GPSAltitudeRef 1

GPSAltitude (0, 1)

GPSTimeStamp ((19, 1), (23, 1), (54, 1))

GPSDateStamp b'2018:10:24'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

## Activité 5 – Récupération et Exploitation des données GPS

**Note pour les enseignants :** L’objectif de cette activité est de faire afficher les données GPS et de traiter ces données en tenant compte du format spécifique d’enregistrement.

### Extraction Trame GPS complète

Les données GPS sont enregistrées dans des champs différents.

On peut noter la présence des lettres N ou S et W ou E dans les informations de géolocalisation.

* N : signifie North ; S signifie South ; E signifie East et W signifie West.
* A la suite de votre programme précédent, rajouter les lignes de code ci-dessous :

print ('Extraction des coordonnées GPS')

print (exifDict["GPS"]) # Récupération totale de l'information GPS

print (exifDict["GPS"][1]) # Récupération b'N' ou b'S'

print (exifDict["GPS"][2]) # Récupération valeur totale N ou S

print (exifDict["GPS"][3]) # Récupération b'E' ou b'W'

print (exifDict["GPS"][4]) # Récupération valeur totale E ou W

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme

1. Dans le cadre ci-dessous, recopier les données affichées.

**Note de correction :** On obtient

Extraction des coordonnées GPS

{0: (2, 2, 0, 0), 1: b'N', 2: ((14, 1), (49, 1), (31, 1)), 3: b'W', 4: ((61, 1), (13, 1), (27, 1)), 5: 1, 6: (0, 1), 7: ((19, 1), (23, 1), (54, 1)), 29: b'2018:10:24'}

b'N'

((14, 1), (49, 1), (31, 1))

b'W'

((61, 1), (13, 1), (27, 1))

Les données de localisation sont enregistrées sous la forme degré, minute, seconde. Elles sont associées avec un diviseur. Le couple de données se présente donc sous la forme (valeur, diviseur). La valeur du diviseur est surtout utilisée dans les secondes car elle peut être différente de 1.

1. Dans le cadre ci-dessous, préciser quelle est la valeur du diviseur des secondes.

**Note de correction :** On obtient

Extraction des coordonnées GPS

{0: (2, 2, 0, 0), 1: b'N', 2: ((14, 1), (49, 1), (31, 1)), 3: b'W', 4: ((61, 1), (13, 1), (27, 1)), 5: 1, 6: (0, 1), 7: ((19, 1), (23, 1), (54, 1)), 29: b'2018:10:24'}

b'N'

((14, 1), (49, 1), (31, 1))

b'W'

((61, 1), (13, 1), (27, 1))

### Extraction Données GPS complètes

* A la suite de votre programme précédent, rajouter les lignes de code ci-dessous :

print (exifDict["GPS"][2][0][0]) # Récupération valeur degré N ou S (d)

print (exifDict["GPS"][2][0][1]) # Récupération diviseur degré N ou S (d)

print (exifDict["GPS"][2][1][0]) # Récupération valeur minute N ou S (m)

print (exifDict["GPS"][2][1][1]) # Récupération diviseur minute N ou S (m)

print (exifDict["GPS"][2][2][0]) # Récupération valeur seconde N ou S (s)

print (exifDict["GPS"][2][2][1]) # Récupération diviseur seconde N ou S (s)

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme

1. Que réalise la ligne de code print (exifDict["GPS"][2][0][0]) ? : *Elle extrait l’information degré en Nord ou Sud.*
2. Que réalise la ligne de code print (exifDict["GPS"][2][0][1]) ? : *Elle extrait l’information diviseur des degrés en Nord ou Sud.*
3. Sur le même modèle que les lignes de code ci-dessus, rajouter les lignes de code pour extraire les informations relatives à la position E ou W.

**Note de correction :** On obtient

print (exifDict["GPS"][4][0][0]) # Récupération valeur degré E ou W (d)

print (exifDict["GPS"][4][0][1]) # Récupération diviseur degré E ou W (d)

print (exifDict["GPS"][4][1][0]) # Récupération valeur minute E ou W (m)

print (exifDict["GPS"][4][1][1]) # Récupération diviseur minute E ou W (m)

print (exifDict["GPS"][4][2][0]) # Récupération valeur seconde E ou W (s)

print (exifDict["GPS"][4][2][1]) # Récupération diviseur seconde E ou W (s)

### Fin de la séance 2

## Activité 6 – Conversion de format

Le format des données de géolocalisation enregistrées dans les metadata est degrés, minutes, secondes associées avec leur diviseurs respectifs. La plupart des sites WEB ou des fonctions utilisées par la suite exigent des données au format décimal. Il faut donc procéder à une étape de conversion pour passer du format dms au format dd.

**Note pour les enseignants :** L’objectif de cette activité est de traiter les données GPS pour les passer du format dms vers le format dd, habituellement utilisé par OpenStreetMap. Le niveau de guidance peut être très différent d’un groupe d’élèves à un autre.

### De dms vers dd

* A la suite de votre programme précédent, proposer et rajouter les lignes de code ci-dessous :

latitudeAbs=exifDict["GPS"][2][0][0]/(exifDict["GPS"][2][0][1])+\

exifDict["GPS"][2][1][0]/(60\*exifDict["GPS"][2][1][1])+\

exifDict["GPS"][2][2][0]/(3600\*exifDict["GPS"][2][2][1])

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme

1. Que signifie le caractère \ en fin de ligne ? : *Il permet d’indiquer à l’IDE que la ligne de code n’est pas terminée.*
2. Sur le même modèle que les lignes de code ci-dessus, proposer les lignes de code permettant d’extraire les informations relatives à la longitudeAbs.

**Note de correction :** On obtient

longitudeAbs=exifDict["GPS"][4][0][0]/(exifDict["GPS"][4][0][1])+\

exifDict["GPS"][4][1][0]/(60\*exifDict["GPS"][4][1][1])+\

exifDict["GPS"][4][2][0]/(3600\*exifDict["GPS"][4][2][1])

1. Proposer les lignes de code permettant de faire afficher la latitudeAbs et la longitudeAbs.

**Note de correction :** On obtient

print ('Latitude valeur absolue (format dd) = ', latitudeAbs,\

'Longitude valeur absolue (format dd) = ', longitudeAbs)

Les données liées à la latitude et à la longitude sont enregistrées uniquement à l’aide de valeurs **positives**. Les coordonnées doivent cependant être placées en positif ou en négatif suivant les informations N/S et E/W.

Les valeurs de latitude et longitude sont positives, respectivement pour N et pour E. Elles sont négatives dans les autres cas.

1. En autonomie (ou avec de la guidance), proposer les lignes de code basées sur des structures de sélection permettant de passer de latitudeAbs à latitude et de longitudeAbs à longitude suivant les valeurs contenues dans exifDict["GPS"][1] et exifDict["GPS"][3]. Faire afficher les valeurs de latitude et longitude.

**Note de correction :** On obtient

# Prise en compte de N ou S et E ou W

if (exifDict["GPS"][1]==b'N'):

latitude=latitudeAbs

else:

latitude=-latitudeAbs

if (exifDict["GPS"][3]==b'E'):

longitude=longitudeAbs

else:

longitude=-longitudeAbs

print ('Latitude (format dd) = ', latitude,\

'Longitude (format dd) = ', longitude)

Note : Il est possible de vérifier le calcul réalisé à partir d’un site web de conversion de dms vers dd comme par exemple <http://xjubier.free.fr/site_pages/DMS2DD_Converter.html>

## Activité 7 – Géolocalisation et affichage du nom et de carte de la prise de vue

### Détermination du nom du lieu de la place de vue

**Note pour les enseignants :** L’objectif de cette activité est d’obtenir les informations de localisation à partir de la connaissance des informations de latitude et longitude au format dd. Elle utilise l’API OpenStreetMap.

Il est possible que le

**Note pour les enseignants :** Vérifier que la librairie geopy est présente dans le répertoire de Python ou l’installer à partir de :



Sous Anaconda Prompt en mode administrateur.

Les informations attendues par cette API sont des chaînes de caractères et non des flottants.

* A la suite de votre programme précédent, rajouter les lignes de code ci-dessous :

from geopy.geocoders import Nominatim

geolocator = Nominatim(user\_agent="XX") # Il faut placer une chaine de

# caractères quelconque

##### ATTENTION : Il faut configurer le proxy (passerelle) au lycée

strLat=str(latitude) # Conversion en chaîne de caractères

strLong=str(longitude) # Conversion en chaîne de caractères

strLatLong=strLat + ', ' + strLong # Concaténation des deux chaînes

location = geolocator.reverse(strLatLong) # Cette fonction attend une

# chaine de caractères

print(location.address)

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme

1. Que réalise la ligne de code strLat=str(latitude) ? : *Elle transforme le « nombre » contenu dans la variable latitude en une chaine de caractères.*
2. Que réalise la ligne de code strLatLong=strLat + ', ' + strLong ? *Elle concatène les deux chaines en ajoutant une virgule au milieu.*
3. Que réalise la ligne de code location = geolocator.reverse(strLatLong) ? *Elle questionne l’API OpenStreetMap pour obtenir les informations de localisation connues à cette position GPS.*
4. Dans le cadre ci-dessous, recopier les données affichées et en particulier le lieu de prise de la photo.

**Note de correction :** On obtient

D 10, Le Prêcheur, Saint-Pierre, Martinique, 97250, France 

Note : Si vous avez de l’avance par rapport à vos camarades, vous pouvez essayer de connaitre le lieu de la prise de vues des photos de vos camarades.

### Obtention de la carte du lieu de la prise de vue

**Note pour les enseignants :** L’objectif de cette activité est d’obtenir l’affichage de la carte à partir de la connaissance des informations de latitude et longitude au format dd. Elle utilise l’API OpenStreetMap. Le niveau de code est plus ardu et il semble pertinent de poser des questions de haut niveau sans nécessairement entrer dans le détail de chacune des lignes.

**Note pour les enseignants :** Vérifier que la librairie smopy est présente dans le répertoire de Python ou l’installer à partir de :



Sous Anaconda Prompt en mode administrateur.

Les informations attendues par cette API sont des flottants regroupés dans un tuple et un niveau de zoom.

* A la suite de votre programme précédent, rajouter les lignes de code ci-dessous :

# Création de la carte avec un niveau de Zoom de 17

import smopy

sMap = smopy.Map((latitude, longitude), z=17) # (lat\_min, lon\_min, lat\_max, lon\_max).

x, y = sMap.to\_pixels(latitude, longitude)

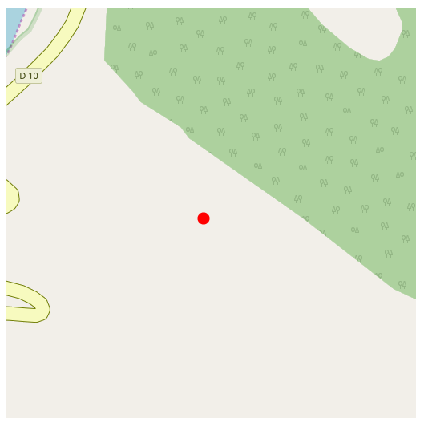
ax = sMap.show\_mpl(figsize=(8, 6))

ax.plot(x, y, 'or', ms=10, mew=2);

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme

1. Que réalise l’ensemble de ces lignes de code ? : *Elles affichent une carte, centrée autour de la position GPS, dont le niveau de zoom est élevé.*
2. Dans le cadre ci-dessous, recopier la carte obtenue à partir des coordonnées GPS.

**Note de correction :** On obtient



1. En autonomie, modifier le niveau de zoom en le passant par exemple à la valeur 6 pour obtenir une vue plus élargie et dans le cadre ci-dessous, recopier la carte obtenue à partir des coordonnées GPS. Vous pourrez au choix modifier le niveau de zoom existant ou créer une seconde vue avec le nouveau niveau de Zoom.

**Note de correction :** On obtient

# Création de la carte avec un niveau de Zoom de 6

sMap = smopy.Map((latitude, longitude), z=6) # (latMin, lgMin, latMax, lgMax)

# z : niveau de zoom

x, y = sMap.to\_pixels(latitude, longitude)

ax = sMap.show\_mpl(figsize=(8,6))

ax.plot(x, y, 'or', ms=10, mew=2);



Note : Si vous avez de l’avance par rapport à vos camarades, vous pouvez essayer de connaitre le lieu de la prise de vues des photos de vos camarades.

### Pour aller un peu plus loin – Création d’un fichier image contenant la carte

**Note pour les enseignants :** L’objectif de cette activité supplémentaire (et facultative) est de créer un fichier de type image (png) contenant la carte obtenue. Les deux éléments (photo initiale et carte) pourront par exemple être placés côte à côte dans une page Web.

**Note pour les enseignants :** Vérifier que la librairie matplotlib est présente dans le répertoire de Python ou l’installer à partir de :



Sous Anaconda Prompt en mode administrateur. (ou passer par pip pour d’autres distributions)

Note : cette librairie est a priori présente dans la quasi-totalité des distributions python.

* A la suite de votre programme précédent, rajouter les lignes de code ci-dessous :

# Sauvegarde de la carte avec un niveau de Zoom de 17

import matplotlib.pyplot as plt

nomCarteSauvegarde=nomFichierImage+'CarteZ17.png' # Pour repl.it

plt.savefig(nomCarteSauvegarde) ### Pour repl.it (Attention très long environ 5mn sur repl.it)

(im.open(nomCarteSauvegarde)).show() ### Pour afficher la carte comme un image

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme

1. Quel est le nom du fichier obtenu, quel est son emplacement ? Il se nomme ouEstCe4.jpgCarteZ17.png. Il se situe dans le répertoire courant de travail.
2. En autonomie, produire les lignes de code permettant de sauvegarder la carte avec un niveau de zoom de 6.

**Note de correction :** On obtient

# Sauvegarde de la carte avec un niveau de Zoom de 6

nomCarteSauvegarde=nomFichierImage+'CarteZ6.png' # Pour repl.it

plt.savefig(nomCarteSauvegarde) ### Pour repl.it (Attention très long environ 5mn sur repl.it)

(im.open(nomCarteSauvegarde)).show() ### Pour afficher la carte comme un image

### Fin de la séance 3

## Activité 8 – Pour aller un peu plus « plus » loin : Lancement d’un navigateur et création d’URL

**Note pour les enseignants :** L’objectif de cette activité supplémentaire (et facultative) est de générer un appel à un navigateur WEB pour faire afficher la position GPS sur une carte.

**Note pour les enseignants :** Vérifier que la librairie webbrowser est présente dans le répertoire de Python ou l’installer à partir de :



Sous Anaconda Prompt en mode administrateur. (ou passer par pip pour d’autres distributions)

Note : cette librairie est a priori présente dans la quasi-totalité des distributions python.

* A la suite de votre programme précédent, rajouter les lignes de code ci-dessous :

import webbrowser

# Construction de l'URL avec un niveau de Zoom de 17

### Attention sur certains navigateurs, il est possible qu’il faille //// à la place de //.

adresse = "https://www.openstreetmap.org/?mlat="\

+str(latitude)+"&mlon="+str(longitude)+"&zoom=17"

webbrowser.open(adresse) # Appel du navigateur par défaut

* Sauvegarder le programme dans le **même** répertoire que l’image de travail sous le nom extractionMetaData.py.
* Lancer le programme

1. Quel est le navigateur ouvert par défaut ? Cela dépend de l’association qui est faite sur le PC ou le MAC.

La ligne de code commençant par adresse = permet de construire une URL (Uniform Resource Locator). Cette URL contient un point d’interrogation (?) et une partie après ce caractère spécial. Une recherche rapide sur Wikipedia permet de trouver l’information suivante.

*Après le point d’interrogation, les informations suivantes correspondent à des données passées lors de l’appel de l’URL. Elles sont séparées par le caractère &.*

1. Quelles sont les données passées dans cette adresse internet ? Cette URL permet de passer les données relatives à la latitude, à la longitude et au niveau de zoom souhaités.
2. En autonomie, produire les lignes de code permettant de créer et d’ouvrir l’URL précédente avec un niveau de zoom de 6.

**Note de correction :** On obtient

# Construction de l'URL avec un niveau de Zoom de 6

adresse = "https://www.openstreetmap.org/?mlat="\

+str(latitude)+"&mlon="+str(longitude)+"&zoom=6"

webbrowser.open(adresse) # Appel du navigateur par défaut

## Activité 9 – Pour aller vraiment beaucoup plus loin : Mise à disposition du code

On peut placer son code sur des sites « participatifs » comme repl.it. Pour cela il suffit de créer un compte et de copier le code en ayant au préalable choisi son langage de développement.