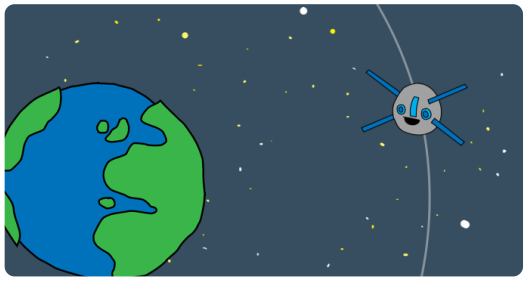
Projets

Où est la Station Spatiale ?

Trouver l'emplacement exact de l'ISS

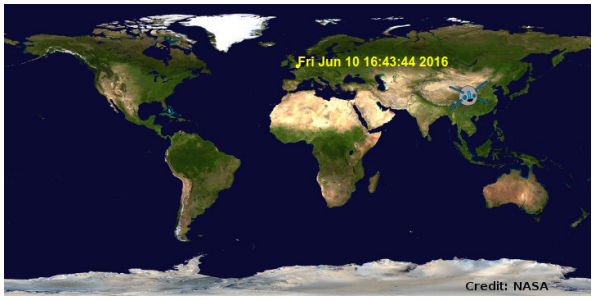


Étape 1 Introduction

Dans ce projet tu vas utiliser un web-service pour trouver la position actuelle de la Station Spatiale Internationale (ISS) et tracer sa localisation sur une carte.

Instructions

L'icône de la station spatiale indique l'emplacement actuel de l'ISS. Le texte jaune indique le prochain passage de l'ISS au-dessus du centre spatial de Houston, aux États-Unis.



Ce que tu vas apprendre

Ce projet couvre des élements des parcours suivants du **Programme de Créativité Numérique Raspberry Pi** (http://rpf.io/curriculum) :

- **Combine des notions de programmation pour résoudre un problème.** (https://www.raspberrypi.org/curriculum/programming/builder)

Étape 2 Ce dont tu auras besoin

Matériel informatique

- Un ordinateur avec une connexion Internet

Logiciel

- Éditeur en ligne **Trinket** (https://trinket.io/)

Ressources supplémentaires

- Projet de démarrage - **rpf.io/iss-on** (http://rpf.io/iss-on)
- Une version complète de ce projet - **trinket.io/python/09efa79ecb** (https://trinket.io/python/09efa79ecb)
- Ouvrir les services Web de notification - **api.open-notify.org** (http://api.open-notify.org/)

Étape 3 Qui est dans l'espace ?

Tu vas utiliser un service web qui fournit des informations en direct sur l'espace. D'abord, découvrons qui est actuellement dans l'espace.

Un service web a une adresse (URL) comme un site web. Au lieu de renvoyer du HTML pour une page Web, il renvoie des données.

- Ouvre **le service web** (http://api.open-notify.org/astros.json) dans un navigateur web.

Tu dois voir quelque chose comme ça :

```
{
  "message": "success",
  "number": 3,
  "people": [
    {
      "craft": "ISS",
      "name": "Yuri Malenchenko"
    },
    {
      "craft": "ISS",
      "name": "Timothy Kopra"
    },
    {
      "craft": "ISS",
      "name": "Timothy Peake"
    }
  ]
}
```

Les données sont en direct, donc tu vois probablement un résultat légèrement différent. Le format de données est appelé `JSON` (prononcé comme 'Jason').

Qu'est ce que JSON ?

JSON est un format de stockage et de partage de données. JSON (prononcé "Jason") signifie JavaScript Object Notation, mais il n'est pas utilisé uniquement avec JavaScript.

JSON est un format de texte qui peut être utilisé du code et est assez facile à lire.

```
{
  "nom": "Ogre",
  "taille": 90,
  "puissance": 86,
  "intelligence": 12,
  "magie": 0
}
```

Un **objet** JSON est une liste de paires clé-valeur entre accolades `{}`.

Une valeur peut également être une **liste** entre crochets `[]` :

```
{
  "nom": "Ogre",
  "taille": 90,
  "puissance": 86,
  "intelligence": 12,
  "magique": 0,
  "armes": ["massue", "pierre", "os"]
}
```

Tu dois appeler le service web à partir d'un script Python, pour pouvoir utiliser les résultats.

- Ouvre cette trinket : **http://rpf.io/iss-on** (http://rpf.io/iss-on).

Les modules `urllib.request` et `json` ont déjà été importés pour toi en haut du script `main.py`.

- Ajoute le code suivant à `main.py` pour stocker l'URL du service web auquel tu as accédé en tant que variable :

```
url = 'http://api.open-notify.org/astros.json'
```

- Maintenant, appelle le service web :

```
url = 'http://api.open-notify.org/astros.json'
reponse = urllib.request.urlopen(url)
```

- Ensuite, tu dois charger la réponse JSON dans une structure de données Python :

```
url = 'http://api.open-notify.org/astros.json'
reponse = urllib.request.urlopen(url)
resultat = json.loads(reponse.read())
print(resultat)
```

Tu dois voir quelque chose comme ça :

```
{'message': 'success', 'number': 3, 'people' : [{'craft': 'ISS', 'name': 'Yuri Malenchenko'}, {'craft': 'ISS', 'name': 'Timothy Kopra'}, {'craft': 'ISS', 'name': 'Timothy Peake'}]}
```

Il s'agit d'un dictionnaire Python avec trois clés : `message`, `number`, et `people`.

Utilisation de paires clé-valeur en Python

Voici un dictionnaire des membres du groupe. La **clé** est la première partie (par exemple 'john'), et sa **valeur** associée est la seconde partie (par exemple 'guitare rythmique').

```
groupe = {
  'john' : 'guitare rythmique',
  'paul' : 'guitare basse',
  'george' : 'guitare principale',
  'ringo' : 'guitare basse'
}
```

Voici comment ajouter une paire clé-valeur au dictionnaire :

```
# Ajouter une paire clé-valeur
groupe['yoko'] = 'voix'
```

Voici comment supprimer une paire clé-valeur du dictionnaire :

```
# Supprimer une paire clé-valeur
del groupe['paul']
```

Ce `message` a la valeur `success` te dit que tu as accédé au service web. Note que tu vois différents résultats pour `number` et `people` selon qui est actuellement dans l'espace.

Maintenant, imprimons l'information de manière plus lisible.

- Tout d'abord, recherchons le nombre de personnes dans l'espace et imprimons-le:

```
url = 'http://api.open-notify.org/astros.json'
reponse = urllib.request.urlopen(url)
resultat = json.loads(reponse.read())

print('Personnes dans l''espace: ', resultat['number'])
```

`resultat['number']` affichera la valeur associée à la clé `number` dans le dictionnaire `resultat`. Dans l'exemple, il s'agit de `3`.

- La valeur associée à la clé `people` est une liste de dictionnaires ! Mettons cette valeur dans une variable pour que tu puisses l'utiliser:

```
print('Personne dans l''espace: ', resultat['number'])

personne = resultat['people']
print(personne)
```

Tu devrais voir quelque chose comme ça :

```
[{'craft': 'ISS', 'name': 'Yuri Malenchenko'}, {'craft': 'ISS', 'name': 'Timothy Kopra'}, {'craft': 'ISS', 'name': 'Timothy Peake'}]
```

- Maintenant tu dois imprimer une ligne pour chaque astronaute. Tu peux utiliser une boucle Python `for` pour ce faire.

Boucle for avec une liste en Python

Cette boucle **for** imprimera chaque élément dans la liste `animaux`.

```
animaux = ["renard", "loup", "panda", "écureuil"]

for animal in animaux:
  print(animal)
```

La sortie est la suivante :

```
renard
loup
panda
écureuil
```

Note que la ligne de code `print` est légèrement plus à droite. Ceci est appelé **indentation** - la ligne est **indentée** pour montrer qu'elle est à l'intérieur de la boucle. Toutes les lignes de code à l'intérieur de la boucle seront répétées.

- À chaque fois à travers la boucle, `p` sera défini sur un dictionnaire pour un astronaute différent.

```
print('Personne dans l''espace: ', resultat['number'])

personne = resultat['people']
print(personne)
```

- Tu peux ensuite rechercher les valeurs pour `name` et `craft`. Montrons les noms des personnes dans l'espace :

```
print('Personne dans l''espace: ', resultat['number'])

personne = resultat['people']

for p in personne:
  print(p['name'])
```

Tu devrais voir quelque chose comme ça :

```
Personnes dans l'espace : 3
Yuri Malenchenko
Timothy Kopra
Timothy Peake
```

Note: Tu utilises des données en direct, donc tes résultats dépendront du nombre de personnes actuellement dans l'espace.

Étape 4 Défi : montrer le vaisseau

En plus du nom des astronautes, le service web fournit également le vaisseau sur lequel ils sont utilisés, comme l'ISS.

- Peux-tu ajouter à ton script pour qu'il imprime également le vaisseau pour chaque astronaute ?

Exemple :

```
Personnes dans l'espace: 3
Yuri Malenchenko dans l'ISS
Timothy Kopra dans l'ISS
Timothy Peake dans l'ISS
```

J'ai besoin d'un indice

Modifie ta boucle `pour` pour qu'elle ressemble à ceci:

```
for p in personne:
  print(p['name'], ' dans ', p['craft'])
```

Étape 5 Où est l'ISS?

La Station spatiale internationale est en orbite autour de la Terre. Elle complète une orbite de la terre à peu près toutes les heures et demies, et voyage à une vitesse moyenne de 7,66 km par seconde. C'est rapide !

Nous allons utiliser un autre service web pour savoir où se trouve la Station spatiale internationale.

- Commence par ouvrir l'URL du service web dans un nouvel onglet dans ton navigateur web : **http://api.open-notify.org/iss-now.json** (http://api.open-notify.org/iss-now.json)

Tu devrais voir quelque chose comme ça :

```
{
"iss_position": {
  "latitude": 8.54938193505081,
  "longitude": 73.16560793639105
},
"message": "success",
"timestamp": 1461931913
}
```

Le résultat contient les coordonnées de l'endroit sur Terre où se trouve actuellement l'ISS.

Que sont la latitude et la longitude?

Latitude et longitude

La latitude et la longitude sont utilisées pour donner des coordonnées à des emplacements sur la surface de la Terre.

La latitude indique la position le long des axes nord-sud et peut avoir n'importe quelle valeur entre 90 et -90. 0 marque l'équateur.

La longitude indique la position le long de l'axe est-ouest, et peut être n'importe quelle valeur entre -180 et 180. 0 marque le premier méridien, qui passe par Greenwich à Londres, Royaume-Uni.

Les coordonnées sont données sous la forme **(latitude, longitude)**. Les coordonnées de l'Observatoire royal de Greenwich sont (51.48, 0). Comme vous pouvez le constater, la latitude (nord-sud) est donnée en premier.

Vous pouvez rechercher la latitude et la longitude de lieux à **latlong.net** (http://www.latlong.net/).

- Maintenant appelons le web-service à partir de Python. Ajoute le code suivant à la fin de ton script :

```
url = 'http://api.open-notify.org/iss-now.json'
reponse = urllib.request.urlopen(url)
resultat = json.loads(reponse.read())

print(resultat)
```

```
{'timestamp': 1597667587, 'message':
'success', 'iss_position': {'latitude':
'-5.7239', 'longitude': '74.8429'}}
```

- Créons des variables pour stocker la latitude et la longitude, puis les imprimer :

```
url = 'http://api.open-notify.org/iss-now.json'
reponse = urllib.request.urlopen(url)
resultat = json.loads(reponse.read())

emplacement = resultat['iss_position']
lat = float(emplacement['latitude'])
lon = float(emplacement['longitude'])
print('Latitude: ', lat)
print('Longitude: ', lon)
```

```
Latitude:  -10.9691
Longitude:  -128.9572
```

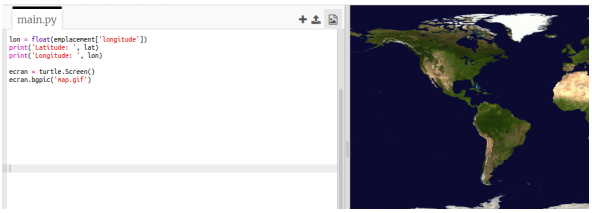
Étape 6 Tracer l'ISS sur une carte

Il serait utile de montrer la position sur une carte. Tu peux le faire en utilisant les graphiques de Python Turtle !

- D'abord, nous allons devoir importer la bibliothèque Python `turtle` :

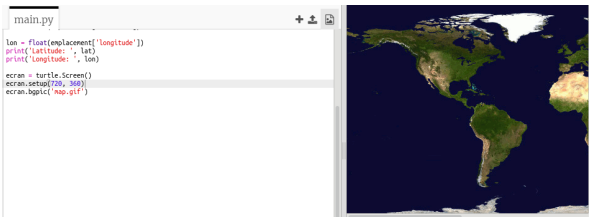
```
import json
import urllib.request
import turtle
```

- Ensuite, charger une carte du monde comme image de fond. Il y en a une déjà inclus dans votre trinket appelée 'map.gif'! La NASA a fourni cette belle carte et a donné l'autorisation pour la réutilisation.



La carte est centrée à `(0,0)` de latitude et de longitude, ce qui est exactement ce dont tu as besoin.

- Tu dois définir la taille de l'écran pour correspondre à la taille de l'image, qui est de 720 par 360 pixels. Ajouter `ecran.setup(720, 360)`:



- Tu veux pouvoir envoyer la tortue à une latitude et une longitude particulières. Pour rendre cela facile, tu peux définir l'écran pour qu'il corresponde aux coordonnées que tu utilises:

```
ecran = turtle.Screen()
ecran.setup(720, 360)
ecran.setworldcoordinates(-180, -90, 180, 90)
ecran.bgpic('map.gif')
```

Maintenant, les coordonnées correspondent aux coordonnées de latitude et de longitude que vous obtiendrez du service web.

- Créons une icône de tortue pour l'ISS. Ton trinket inclut 'iss.gif' et 'iss2.gif' — essaie-les toutes les deux et vois celle que tu préféres.

Modifier les icônes de Python Turtle

Modifier les icônes de Python Turtle

Au lieu de toujours utiliser une tortue, tu peux indiquer à l'icône Python Turtle d'utiliser une image différente. L'image doit être petite pour ne pas trop recouvrir l'écran : 50 × 50 pixels te donnera une grande icône.

- Tu dois d'abord enregistrer l'image avec la fonction `ecran` :

```
ecran = turtle.Screen()
ecran.register_shape('happy.png')
```

- Ensuite, tu peux définir la `forme` :

```
turtle.shape('happy.png')
```

- Les icônes Turtle sont en face pour commencer. Tu peux changer le titre pour que ton image soit orientée vers le haut :

```
turtle.setheading(90) # faire face vers le haut
```

Voir un exemple ici :

J'ai besoin d'un indice

Ton code devrait ressembler à ceci:

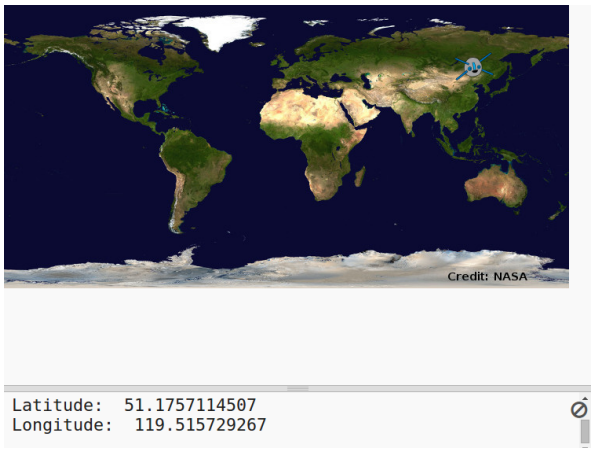
- L'ISS commence au centre de la carte, nous allons maintenant le déplacer à l'emplacement correct :

```
ecran.register_shape('iss.gif')
iss = turtle.Turtle()
iss.shape('iss.gif')
iss.setheading(90)

iss.penup()
iss.goto(lon, lat)
```

Note : la latitude est normalement donnée en premier, mais nous devons d'abord donner de la longitude lors du tracé des coordonnées `(x,y)`.

- Teste ton programme en l'exécutant. L'ISS devrait se déplacer à son emplacement actuel au-dessus de la Terre.



- Attends quelques secondes et lance à nouveau ton programme pour voir où l'ISS s'est déplacé.

Étape 7 Quand l'ISS sera-t-il au-dessus?

Il y a aussi un service web que tu peux utiliser pour savoir quand l'ISS se trouvera ensuite sur un emplacement particulier.

Découvrons quand l'ISS sera sur le Centre Spatial de Houston, aux Etats-Unis, qui est à la latitude `29.5502` et à la longitude `95.097`.

- Commençons par tracer un point sur la carte à ces coordonnées :

```
iss.penup()
iss.goto(lon, lat)

# Space Center, Houston
lat = 29.5502
lon = -95.097

emplacement = turtle.Turtle()
emplacement.penup()
emplacement.color('yellow')
emplacement.goto(lon,lat)
emplacement.dot(5)
emplacement.hideturtle()
```

Maintenant nous allons obtenir la date et l'heure du prochain passage de l'ISS au-dessus de là.

- Comme avant, tu peux appeler le service web en entrant son URL dans la barre d'adresse d'un navigateur web : **api.open-notify.org/iss-pass.json** (http://api.open-notify.org/iss-pass.json)

Tu devrais voir une erreur :

```
{
  "message": "failure",
  "reason": "Latitude must be specified"
}
```

Ce service web prend la latitude et la longitude en tant qu'entrées, donc tu dois les inclure dans l'URL. Les entrées sont ajoutées après un `?` et séparées par `&`.

- Ajouter les entrées `lat` et `lon` à l'url comme indiqué : **api.open-notify.org/iss-pass.json?lat=29.55&lon=95.1** (http://api.open-notify.org/iss-pass.json?lat=29.55&lon=95.1)

```
{
  "message": "success",
  "request": {
    "altitude": 100,
    "datetime": 1465541028,
    "latitude": 29.55,
    "longitude": 95.1,
    "passes": 5
  },
  "response": [
    {
      "duration": 630,
      "risetime": 1465545197
    },
    {
      "duration": 545,
      "risetime": 1465551037
    },
    {
      "duration": 382,
      "risetime": 1465568806
    },
    {
      "duration": 625,
      "risetime": 1465574518
    }
  ]
}
```


La réponse comprend plusieurs temps de passage, et nous allons juste regarder la première. L'heure est donnée sous forme d'horodatage Unix (vous pourrez la convertir en une heure lisible dans votre script Python).

Horodatage Unix

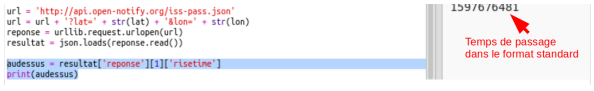
Les horodatages Unix constituent un moyen pratique de stocker une date et une heure sous la forme d'un nombre unique.

Un horodatage Unix est le nombre de secondes écoulées depuis le 1er janvier 1970 dans l'UTC (norme internationale pour l'heure). Par exemple, `1498734934` est le 29 juin 2017 à 11h15.

Tu peux trouver l'horodatage Unix actuel sur **unixtimestamp.com** (http://www.unixtimestamp.com/).

- Appelons maintenant le service Web depuis Python. Ajoute le code suivant à la fin de ton script:

- Voyons maintenant le premier temps de passage à partir du résultat. Ajoute le code suivant:



Nous aurons besoin du module Python `time` pour pouvoir l'imprimer sous une forme lisible et le convertir en heure locale. Ensuite, nous allons obtenir le script pour écrire le temps de passage par le point pour Houston.

- Ajouter une ligne `import time` en haut de votre script:

```
import json
import urllib.request
import turtle
import time
```

- La fonction `time.ctime()` convertira l'horodatage en une forme lisible que tu peux écrire sur ta carte :

```
audessus = resultat['reponse'][1]['risetime']
#print(audessus)

style = ('Arial', 6, 'bold')
emplacement.write(time.ctime(audessus), font=style)
```

(Tu peux supprimer la ligne `print` ou la transformer en un commentaire en ajoutant `#` au début pour que ton script l'ignore.)

- Si tu le souhaites, tu peux changer la couleur et le format du texte.

Écrire du texte avec Python Turtle

Tu peux utiliser une tortue pour écrire du texte.

```
turtle.write('Bonjour !')
```

Définis la couleur de la tortue pour créer un texte coloré :

```
turtle.color('deep pink')
turtle.write('Bonjour !')
```

Tu peux également modifier la police et l'alignement du texte.

```
style = ('Courier', 30, 'italic')
turtle.write('Bonjour !', font=style, align='center')
```

La police est un tuple contenant :

- Le nom de la police tel que « Arial », « Courier » ou « Times New Roman »
- La taille de la police en pixels
- Le type de police, qui peut être « normal », « gras » ou « italique »

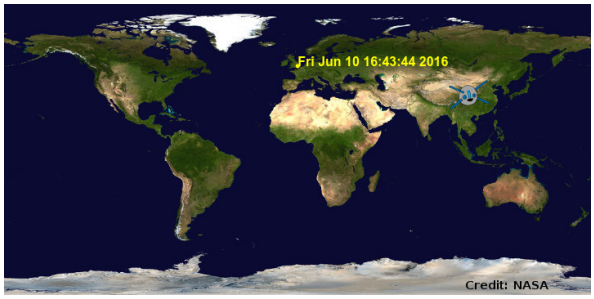
Pour définir l'alignement qui contrôle le positionnement du texte en fonction de la position de la tortue, utilise le paramètre `align`. `align` peut être défini sur l'une de ces options : 'left', 'center', 'right'

Exemple :

Étape 8 Défi : trouver plus de temps de passage

Pour rechercher la latitude et la longitude d'un emplacement qui t'intéresse, tu peux utiliser un site Web tel que **www.latlong.net/** (http://www.latlong.net/).

- Tu peux chercher et tracer les temps de passage pour plus d'endroits?



J'ai besoin d'un indice

Voici un exemple utilisant l'emplacement du Cosmodrome de Baïkonour, un port spatial dans le sud du Kazakhstan. Le code va à la fin de ton programme, après avoir tracé le temps de passage du Houston Space Center.

```
# Cosmodrome de Baïkonour
lat = 45,86
lon = 63,31

emplacement.penup ()
emplacement.color ('orange')
emplacement.goto (lon, lat)
emplacement.dot (5)
emplacement.hideturtle ()

url = 'http://api.open-notify.org/iss-pass.json?lat=' + str (lat) + '&lon =' + str (lon)
reponse = urllib.request.urlopen (url)
resultat = json.loads (reponse.read ())

#print (resultat)
audessus = resultat ['reponse'][1]['risetime']
emplacement.write (time.ctime (audessus))
```

Essaye d'ajouter plus d'emplacements !

Ce projet a été traduit par des bénévoles:

Abdul Gafur

Michel Arnols

Grâce aux bénévoles, nous pouvons donner aux gens du monde entier la chance d'apprendre dans leur propre langue. Vous pouvez nous aider à atteindre plus de personnes en vous portant volontaire pour la traduction - plus d'informations sur **rpf.io/translate** (https://rpf.io/translate).