Mini-projets

Objectifs

À l'issue de la semaine 5, vous avez à présent un bagage très complet en ce qui concerne les notions usuelles du langage.

Pour les deux dernières semaines, les cours vont adresser des points plus spécifiques ou avancés. Nous vous proposons pour ces deux semaines un programme d'exercices volontairement réduit, pour vous donner le temps de vous investir dans un mini-projet.

Après avoir écrit tout d'abord de tous petits fragments de code dans les semaines 2 à 4, puis un module entier avec quelques classes en Semaine 5, il est temps pour vous de vous essayer à écrire une petite application complète.

Les sujets

Pour vous donner des idées dans ce sens, nous vous proposons de choisir un (ou plusieurs, naturellement) sujets parmi ces trois.

- `diskusage.py` est un outil orienté *Command-Line*, qui peut être utile dans la vie de tous les jours lorsqu'il s'agit de faire le ménage dans un disque trop plein. Ce sujet vous permettra de voir de plus près un exemple de `script`, comparable en gros à ce qu'on pourrait faire en shell.
- `webcrawler.py` est un prototype de robot qui aspire les pages web à partir d'une URL qui sert de point d'entrée. Son objectif et d'avoir un moyen simple d'inspecter un site Web pour, par exemple, indentifier automatiquement les liens morts.
- `meteodata.py` est un exemple de code qui extrait des données météo fournies par un service Internet, qui servent ensuite de prétexte pour faire un peu de visualisation avec `matplotlib`.

Chacun de ces sujets de mini-projet est détaillé dans un notebook séparé.

Modalités

Pour chaque sujet, l'esprit général consiste pour nous à expliquer le contexte, vous donner quelques pistes pour l'implémentation, et vous montrer les différentes ***features*** que nous avons implémentées, pour vous donner des idées. Mais naturellement tout ceci est à titre indicatif, à vous de choisir les ***features*** qui vous intéressent.

En guise de corrigés, nous publierons vers la fin du MOOC nos codes source pour chacun des trois sujets.

Il nous faut souligner que le code de ces corrigés n'est ni totalement fiable, ni validé par des

tests ; et ce pour plusieurs raisons :

- d'une part un code durci a tendance à être plus bavard et plus lourd à lire, et donc à être plus difficile d'accès ;
- d'autre part naturellement cela prend plus de temps à écrire.

Est-ce que cela est un problème pour ce projet ? Non, mais il est important dans du vrai code de production de le fiabiliser au maximum (en capturant les exceptions avec des reactions appropriées, et en testant toutes les entrées) et de le tester (en ajoutant, par exemple, des tests unitaires et des tests fonctionnels). Cela a évidement un coût très élévé en terme d'augmentation du code (on peut facilement multiplier par 2 ou 3 le nombre de lignes de code) et de temps de développement (il va falloir imaginer tous les cas à tester).

En parallèle du mini-projet à proprement parler, un objectif de l'exercice est de vous donner l'occasion de jeter un coup d'oeil à la librairie standard python, que vous pouvez parcourir à partir de ce point d'entrée

https://docs.python.org/2/library/index.html (https://docs.python.org/2/library/index.html)

Vous y retrouverez les modules dont nous avons déjà parlé, et bien d'autres.
