Accès aux fichiers en Python

Site Internet : **www.gecif.net**	Type de document : **Cours**	Intercalaire :	Date :

I – Accès à un fichier texte en lecture seule

Imaginons que nous disposions d'un fichier texte nommé `fichier.txt`, encodé avec le jeu de caractères `cp1252` (appelé **ANSI** dans le menu **Edition + Format de fichier** dans EduPython) et contenant les 2 lignes suivantes :

```
Ceci est un fichier texte contenant 2 lignes.
Il a été créé avec l'éditeur d'EduPython.
```

Pour accéder à un fichier texte en Python il faut créer un descripteur de fichier avec la fonction `open()` en précisant le nom du fichier à lire, le mode d'accès (`'r'` pour ***read***, soit ***lecture seule***), et le jeu de caractères à utiliser :

```
fic=open('fichier.txt','r',encoding='cp1252')
```

Le mode d'accès par défaut est lecture seule [`'r'`] et le jeu de caractères utilisé par défaut est `cp1252`. La ligne suivante est donc strictement équivalente, mais moins explicite : `fic=open('fichier.txt')`

On obtient alors un objet [nommé ici `fic`] appelé ***« descripteur de fichier »*** et possédant des méthodes permettant d'accéder au fichier. Pour lire l'intégralité du fichier texte on appelle la méthode `read()` de l'objet `fic` qui renvoie alors la totalité du fichier sous forme d'une chaîne de caractères avec `\n` pour les retours à la ligne :

```
>>> fic.read()
'Ceci est un fichier texte contenant 2 lignes.\nIl a été créé avec l'éditeur
d'EduPython.'
```

Lorsque Python accède à un fichier il utilise pour se repérer dans le fichier ***« un curseur »*** correspondant à la position courante de lecture dans le fichier. Juste après l'ouverture du fichier par `open()` le curseur est en position 0, c'est-à-dire au début du fichier. Après la lecture complète du fichier avec la méthode `read()` le curseur est en position finale, c'est-à-dire à la fin du fichier. Pour régler la position du curseur il faut utiliser la méthode `seek()` liée au descripteur de fichier `fic`. Par exemple pour remettre le curseur au début du fichier il faut appeler `fic.seek(0)`. Pour connaître la position courante du curseur de lecture dans un fichier sans la modifier il faut faire appel à la méthode `tell()`. La gestion du curseur permet de lire seulement une partie précise du fichier.
Par exemple si on veut lire seulement la chaîne de caractères `'est un fichier'` [contenant 14 caractères à partir de la position 5] depuis de notre fichier texte il faut positionner le curseur en position 5 grâce à `fic.seek(5)` puis demander la lecture de seulement 14 caractères grâce à `fic.read(14)`. Comme on est partie de la position 5 et qu'on a lu 14 caractères, le curseur a avancé de 14 positions : il est donc en position 19 après la lecture comme l'indique `fic.tell()`.

```
>>> fic.seek(5)
5
>>> fic.read(14)
'est un fichier'
>>> fic.tell()
19
```

A retenir : **la méthode `read(n)` renvoie n caractères à partir de la position courante du curseur**

Enfin si on a fini de travailler sur le fichier il faut le refermer en appelant la méthode `fic.close()`.

Si notre fichier texte est encodé en UTF-8 afin de pouvoir contenir n'importe quel caractères Unicode, il faut alors le préciser lors de l'ouverture du fichier dans le 3ème paramètre `encoding=` de la fonction `open()` :

```
>>> fic=open('fichier.txt','r',encoding='utf8')
>>> fic.read()
'\ufeffCeci est un fichier texte contenant 2 lignes.\nIl a été créé avec l'éditeur
d'EduPython.'
```

La lecture des caractères accentués s'effectue alors sans problème, mais on remarque que notre fichier commence par un caractère Unicode particulier de point de code **$FEFF** qui est un espace insécable indiquant que le fichier est encodé en UTF-8. Ce caractère est écrit **\ufeff** dans la chaîne de caractères renvoyée par la méthode `read()`. Et il est codé par les 3 octets **\xef\xbb\xbf** en UTF-8. Si on ne veut pas l'afficher il y a 2 solutions :

```
fic=open('fichier.txt','r',encoding='utf8')
s=fic.read()
print(s[1:])
fic.close()
```

```
fic=open('fichier.txt','r',encoding='utf8')
fic.seek(3)
print(fic.read())
fic.close()
```

A retenir : **un fichier texte encodé en UTF-8 commencera toujours par les 3 octets \xef\xbb\xbf (codage en UTF-8 du caractère Unicode de point de code U+FEFF appelé BOM et indiquant que le fichier est encodé en UTF-8) avant les caractères enregistrés dans le fichier. Pour l'éviter il faut filtrer la lecture.**

II – Accès à un fichier texte en écriture

Pour accéder à un fichier en écriture afin d'enregistrer des informations, il faut l'ouvrir avec le mode d'accès `'w'` (comme ***write***, soit ***écrire***). Le jeu de caractères utilisé par défaut est toujours **ANSI** (page de code `cp1252`) :

```
fic=open('fichier.txt','w')
```

Attention : si le fichier existe déjà il sera détruit afin d'en créer un nouveau. Une fois le fichier ouvert en écriture on peut y écrire des chaînes de caractères grâce à la méthode `write()` :

```
fic.write('bonjour')
```

Si on veut sauter une ligne dans le fichier il faut insérer consciemment des caractères `\n` dans la chaîne à écrire :

```
fic.write('Cet unique\nappel de méthode\nwrite a écrit\n4 lignes dans le fichier')
```

Une fois qu'on a fini d'écrire dans le fichier il faut absolument fermer le descripteur pour que les caractères soit physiquement enregistrés sur le disque dur : `fic.close()`

Si on veut ajouter du texte à la fin d'un fichier déjà existant, il faut alors l'ouvrir avec le mode d'accès `'a'` (comme ***append***, soit ***ajouter***) :

```
fic=open('fichier.txt','a')
```

Les informations écrites dans le fichier grâce à la méthode `write()` seront alors ajoutées à la fin du fichier sans détruire les lignes déjà existantes dans le fichier texte. Le mode d'accès `'a'` permet donc de compléter un fichier.

Si on souhaite créer un fichier texte encodé en UTF-8, il faut le préciser à la fonction `open()` dès la création du fichier. Mais attention, il convient d'écrire manuellement le premier caractère **\ufeff** (le « BOM » pour l'anglais Byte Order Mark) servant de marqueur pour indiquer que le fichier est bien encodé en UTF-8 :

```
fic=open('fichier.txt','w',encoding='utf8')
fic.write('\ufeffCeci est un nouveau fichier encodé en UTF-8')
fic.close()
```

III – Accès à un fichier en mode binaire

Un fichier est avant tout un ensemble d'octets enregistrés sur le disque dur, soit un ensemble de valeur numérique codé en binaire naturel sur 8 bits. Lorsqu'un fichier est ouvert en ***mode texte*** comme précédemment, chaque valeur numérique des octets est convertit en un caractère en utilisant un jeu de caractères bien précis (ANSI par défaut).

Mais lorsqu'un fichier est ouvert en ***mode binaire***, les octets sont lus directement, sans aucune interprétation.

Pour ouvrir un fichier en lecture seule en mode binaire il faut utiliser le mode d'accès `'rb'` (**b** comme **binaire**) :

La méthode `read()` renvoie alors ***une chaîne binaire*** dans laquelle les octets sont remplacés par leur caractère ASCII s'il ont une valeur inférieure à 128, ou par des séquences de la forme **\xnn** indiquant leur valeur numérique. Par exemple si on ouvre en mode binaire notre fichier texte contenant 2 lignes et encodé en UTF-8 on obtient :

```
fic=open('fichier.txt','rb')
fic.read()
b'\xef\xbb\xbfCeci est un fichier texte contenant 2 lignes.\nIl a \xc3\xa9t\xc3\xa9
cr\xc3\xa9\xc3\xa9 avec l\xe2\x80\x99\xc3\xa9diteur d\xe2\x80\x99EduPython.'
```

On y voit en clair le codage interne en binaire sur plusieurs octets :

- du marqueur UTF-8 au début du fichier (appelé BOM) : `\xef\xbb\xbf`
- de l'apostrophe en UTF-8 : `\xe2\x80\x99`
- des e accent aigu (é) en UTF-8 : `\xc3\xa9`

Exemple d'application : en testant les 3 premiers octets d'un fichier binaire et en les comparant au BOM (l'indicateur d'ordre des octets) on peut déterminer si le fichier est encodé en UTF-8 :

```
fic=open('fichier.txt','rb')
bin=fic.read()
debut=bin[0:3]
if debut==b'\xef\xbb\xbf':
        print('Ce fichier est encodé en UTF-8')
else:
        print("Ce fichier n'est pas encodé en UTF-8 (sans doute en ANSI)")
fic.close()
```

Enfin, pour accéder en écriture à un fichier binaire faut utiliser le mode d'accès `'wb'`. Dans ce cas il faudra passer en paramètre à la méthode `write()` une chaîne binaire :

```
fic=open('fichier.txt','wb')
fic.write(b'\xef\xbb\xbfCe fichier est encod\xc3\xa9 en UTF-8')
```