# Initiation à Python - Complément PDF
# L’encodage

Un jour ou l'autre, le développeur Python sera confronté à un message d'erreur dû à un caractère accentué. Sous Python - mais c'est le cas avec d'autres langages - la gestion des textes (chaînes de caractères) n'est vraiment pas simple...

**Le stockage des caractères en mémoire**

Le problème est lié à la façon dont les caractères sont stockés en mémoire.
Au tout début de l'informatique s'est imposé le standard *ASCII* qui définit seulement 128 caractères numérotés de 0 à 127 et codés en binaire de 0000000 à 1111111. Pour pouvoir utiliser des caractères accentués ainsi que tous les autres alphabets que l'on trouve dans le monde de nouvelles normes ou standards ont vu le jour, parmi lesquels le Standard *Unicode* (famille UTF), qui couvre plusieurs dizaines de langues avec tous leurs symboles ou glyphes. C'est un codage quasi universel des lettres ou syllabes, chiffres ou nombres, symboles divers, signes diacritiques et signes de ponctuation. D'où son nom : Unicode.
Cependant, le répertoire Unicode peut contenir plus d’un million de caractères, ce qui est bien trop grand pour être codé par un seul octet. La norme Unicode définit donc des méthodes standardisées pour coder et stocker ce répertoire sous forme de séquence d’octets : UTF-8 est l'une de ces méthodes. Cette méthode s'impose progressivement.

Un exemple : le code unicode du caractère é est U+00E9 et son codage utf-8 est c3 a9

**Les types str de Python**

Le type str de Python (chaîne de caractères) code chaque caractère sur un seul octet ce qui n'offre que 256 possibilités. Cela ne permet donc pas de décrire les dizaines de milliers de glyphes qui existent de par le monde. Python est cependant capable d'utiliser différents jeux de caractères ("charset" en anglais) selon que l'on souhaite travailler avec l'alphabet latin, chinois ou arabe, par exemple.
Python appelle ces jeux de caractères "codec" ou "encoding" ; ce sont des tables qui listent les caractères et symboles d'un alphabet ou système d'écriture et les associent à un numéro compris entre 0 et 255. Mais par défaut, le codec des chaînes de caractères dans Python est souvent l'ASCII.

Pour le vérifier :

```
>>> import sys
>>> sys.getdefaultencoding()
```

Le codec ASCII, nous l'avons dit, ne sait pas gérer les caractères accentués. La présence d'un simple accent dans une chaîne de caractères peut provoquer le plantage d'une fonction.

Comment faire ? Utiliser une variable de type str impose de garder à l'esprit que ses symboles sont associés à un codec particulier. Voyons différents cas de figure.

Cas de l'affichage écran : ???

Cas de la lecture de ses propres fichiers
Le conseil principal est de sauver vos fichiers en UTF-8. Python devrait se débrouiller pour gérer correctement la chaîne accentuée.
En cas de souci, utiliser le module codecs et sa fonction open pour manipuler directement des chaînes unicode.

```
>>> import codecs
>>> f = codecs.open("test.txt", encoding='utf-8') # on précise l'encodage à l'ouverture
>>> txt = f.read() # txt est une chaîne unicode
```

Cas de l'écriture dans ses propres fichiers
Le conseil principal est d'utiliser le module codecs et sa fonction open pour manipuler directement des chaînes unicode.

```
>>> import codecs
>>> test = codecs.open("test2.txt", "w", encoding="utf-8") # on précise l'encodage à l'ouverture
>>> test.write("Iles\n")
>>> test.write("Iles\n")
>>> ver = u"Iles où l'on ne prendra jamais terre\n"
>>> test.write(ver)
>>> test.close()
```

Cas de la lecture ou de l'écriture dans des fichiers autres
Ce cas est souvent problématique à moins de connaître précisément l'encodage qui a été mis en oeuvre. Nous renvoyons vers le lien suivant :

http://sametmax.com/lencoding-en-python-une-bonne-fois-pour-toute/

Cas des noms de fichiers
Pour éviter tout problème, il est fortement conseillé de ne pas mettre de caractères accentués dans les noms de fichier !

**Bibliographie**

Liste des caractères ASCII
http://fr.wikipedia.org/wiki/American_Standard_Code_for_Information_Interchange#Table_des_128_caract.C3.A8res_ASCII

http://www.geoinformations.developpement-durable.gouv.fr/fichier/pdf/Python_-_Gerer_les_caracteres_accentues_dans_les_textes_cle213b6c.pdf?arg=177830528&cle=31ca778b2d3

98b06d33455d7df94aa4653ea3726&file=pdf%2FPython_-_Gerer_les_caracteres_accentues_dans_les_textes_cle213b6c.pdf

https://docs.python.org/2/howto/unicode.html