Gestion des fichiers en Python

La fonction clé pour travailler avec des fichiers en Python est la fonction `open()`.

Cette fonction accepte **2 paramètres** :

- le nom de fichier
- le mode de lecture

Il existe **4 méthodes** différentes pour ouvrir un fichier :

- `r` : **Read** - Valeur par défaut. Provoque une erreur si le fichier n'existe pas.
- `a` : **Append** - Ouvre un fichier pour l'ajout, crée le fichier s'il n'existe pas.
- `w` : **Write** - Ouvre un fichier en écriture, crée le fichier s'il n'existe pas.
- `x` : **Créer** - Crée le fichier spécifié, renvoie une erreur si le fichier existe.

De plus on peut spécifier si le fichier doit être traité en mode binaire ou texte :

- `t` : **Texte** - Valeur par défaut. Mode texte.
- `b` : **Binaire** - Mode binaire.

In [9]:
```
f = open("fichier_demo.txt", "x")
```

In [39]:
```
f = open("fichier_demo.txt", "a")
f.write("Bonjour ! Bienvenue dans fichier_demo.txt\nCe fichier est à des fins de te
```

Out[39]:
```
91
```

In [3]:
```
f = open("fichier-demo.txt", "a")
f.write("Ce fichier est à des fins de test.")
```

Out[3]:
```
34
```

In [4]:
```
f = open("fichier-demo.txt", "a")
f.write("Bonne chance !")
```

Out[4]:
```
14
```

Ouverture et lecture d'un fichier

On utilise la méthode `open()` qui renvoie un objet **file**, qui a une méthode ``**read()** pour lire le contenu du fichier.

In [20]:
```
f = open("fichier_demo.txt", "r")
print((f.read()))
```

```
Bonjour ! Bienvenue dans fichier_demo.txt
Ce fichier est à des fins de test.
Bonne chance !
```

Pour ouvrir un fichier situé à un emplacement différent, faites précéder le nom du fichier par son chemin (le nom du répertoire où se trouve le fichier):

In [23]:
```
f= open("C:\\Users\\utilisateur\\Desktop\\Projet_CDE\\code\\fichier_demo.txt", "r")
print(f.read())
```

```
Bonjour ! Bienvenue dans fichier_demo.txt
Ce fichier est à des fins de test.
Bonne chance !
```

Après avoir ouvert le fichier, nous pouvons lire les données du fichier en utilisant la méthode **`read()`**. Elle prend un argument facultatif en entrée pour spécifier le nombre de caractères à lire dans le fichier. Si la méthode **`read`** est invoquée sur l'objet fichier sans aucun argument, elle lit le fichier entier et le renvoie sous forme de chaîne de texte.

Lecture d'une partie du fichier

On spécifie le nombre de caractères qui s'afficheront dans la méthode **`read()`**.

In [24]:
```
f = open("fichier_demo.txt", "r")
print(f.read(5))
```

```
Bonjo
```

`readline()` : Lire le fichier ligne par ligne

In [29]:
```
f = open("fichier_demo.txt", "r")
print(f.readline())
```

```
Bonjour ! Bienvenue dans fichier_demo.txt
```

En appelant 2 fois **`readline()`**, on peut lire les deux premières lignes.

On peut également lire l'ensemble du fichier grâce à une boucle :

In [30]:
```
f = open("fichier_demo.txt", "r")
for ligne in f :
    print(ligne)
```

```
Bonjour ! Bienvenue dans fichier_demo.txt

Ce fichier est à des fins de test.

Bonne chance !
```

Fermeture d'un fichier : `close()`

In [32]:
```
f = open("fichier_demo.txt", "r")
print(f.read())
f.close()
```

```
Bonjour ! Bienvenue dans fichier_demo.txt
Ce fichier est à des fins de test.
Bonne chance !
```

Ecrire dans un fichier existant

On utilise soit **append** `"a"` soit **write** `"w"` .

Ecraser le contenu :

```
f = open("test_file.txt", "w")
f.write("Contenu écrasé.")
f.close()
```

Ajouter une ligne :

```
f = open("test_file.txt", "a")
f.write("Ajout d'une ligne.")
f.close()
```

Supprimer des fichiers

Nécessite d'importer le module **os** et sa fonction **`os.remove()`** :

```
import os
```

```
os.remove("fichier_tdemo.txt")
```

Supprimer un fichier avec son chemin absolu

In [45]:
```
os.remove(r"C:\\Users\\utilisateur\\Desktop\\Projet_CDE\\code\\fichier_demo_new.tx
```

Vérifier si le fichier existe avant de le supprimer

In [37]:
```
import os

if os.path.exists("fichier_demo.txt"):
    os.remove("fichier_demo.txt")

else :
    print("Le fichier n'existe pas")
```

Supprimer un répertoire avec `os.rmdir()`

Note : on ne peut supprimer que des dossiers vides.

In [ ]:
```
import os

os.rmdir("mon_dossier")
```

Copier des fichiers

Il existe plusieurs façons de copier des fichiers. La méthode `shutil.copy()` est utilisée pour copier le contenu du fichier source dans le fichier de destination. Il faut au préalable importer la méthode **shutil**.

In [40]:
```
import shutil

# chemin source
src_path = r"C:\\Users\\utilisateur\\Desktop\\Projet_CDE\\code\\fichier_demo.txt"
# chemin destination
destination_path = r"C:\\Users\\utilisateur\\Documents\\fichier_demo.txt"

shutil.copy(src_path, destination_path)
print("Le fichier a bien été copié.")
```

```
Le fichier a bien été copié.
```

Renommer des fichiers

Le module **os** fournit la méthode `rename()` pour spécifier le nom de fichier avec le nouveau nom.

In [43]:
```
import os

# chemin absolu d'un fichier
ancien_nom = r"C:\\Users\\utilisateur\\Desktop\\Projet_CDE\\code\\fichier_demo.txt"
nouveau_nom = r"C:\\Users\\utilisateur\\Desktop\\Projet_CDE\\code\\fichier_demo_new

# change le nom du fichier
os.rename(ancien_nom, nouveau_nom)
```