**Acceso Local: Lectura, Escritura y ficheros TXT**

En esta unidad vas a ver diferentes maneras de leer y escribir datos desde archivos locales y, en general, volcarlos a un DataFrame, aunque lo que aprenderás te servirá para cualquier otro procesamiento que necesites. Rara vez trabajarás únicamente con los datos que genere tu programa de Python, sino que lo normal será acudir a una fuente de datos, o leer de algún archivo.

En concreto en esta sesión veremos la forma genérica de leer y escribir archivos que tiene Python y lo aplicaremos a lo que hemos llamado ficheros de texto plano.

**1. Archivos**

**Un archivo es un conjunto de datos almacenados en el ordenador en forma de bits.** Los datos se organizan en un formato específico, pudiendo ser un archivo de texto, un ejecutable, etc como comentamos en las introducciones de la unidad anterior, pero en el fondo todos esos archivos se traducen a nivel binario para el procesado del ordenador. Poniéndonos formales, los archivos se componen de:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

1. **Header**: metadatos del archivo (nombre, tamaño, tipo...)
2. **Data**: contenido del archivo
3. **End of file (EOF)**: caracter especial que indica el final del archivo.

Tú al abrir un archivo texto o un word no ves estas partes, internamente el ordenador sí las considera.

**File path**

Hay tres elementos que tenemos que conocer cuando leamos un archivo:

1. **Folder path**: en qué lugar del ordenador está el archivo.
2. **File name**
3. **Extension**: lo que va después del punto

Texto, Carta

Descripción generada automáticamente

* Si estamos trabajando en el directorio *to*, accederemos a *cats.gif* como cats.gif
* Si queremos leer *dog\_breeds.txt*, hay que ir un directorio hacia atrás, ../dog\_breeds.txt
* Y si queremos acceder a animals.csv, son dos directorios hacia atrás: ../../animals.csv

Recordarás que lo anterior son rutas relativas (no empiezan en C: o en /). Siempre podemos poner la ruta absoluta (/animals.csv, para el caso del ejemplo de la figura, claro) para el acceso a cada archivo, **aunque no es lo recomendable**.

Aquí te dejo una [buena guía para iniciarse en la lectura/escritura de archivos con Python.](https://realpython.com/read-write-files-python/)

**2. Abrir ficheros**

Aunque veremos la forma concreta que tiene pandas para abrir archivo (una de las cuales pd.read\_csv ya la has visto bastantes veces), Python tiene sus propias funciones *built-in*, para tratar con ficheros. Básicamente: open.

Para ello **usaremos la función open**, que devuelve un objeto de tipo File, con unos métodos y atributos propios empleados para obtener información de los archivos abiertos.

file\_object **=** open("filename", "mode")

El primer argumento es el nombre del archivo, mientras que en el modo tendremos que especificar si queremos leer, o escribir. Por defecto leerá, es decir, el parámetro valdrá *r*, de read. [Te dejo el enlace a la documentación para consultar el resto de modos](https://docs.python.org/3/library/functions.html#open).

Vamos a probar a leer un archivo:

with open("./data/dog\_breeds.txt","r") as open\_file:

all\_text=open\_file.read()

print(type(all\_text))

print(all\_text)

Texto, Carta

Descripción generada automáticamente

La sintaxis de línea que has visto es la recomendada, porque en algún momento se tiene que cerrar el archivo. Se abre, leemos, realizamos operaciones, y cuando acaba el with open(), se cierra el archivo.

**Leer y escribir mientras los archivos están abiertos nos dará errores**.

El método .read() nos devuelve un string con todo el texto, que no es lo ideal para tratar luego los datos.

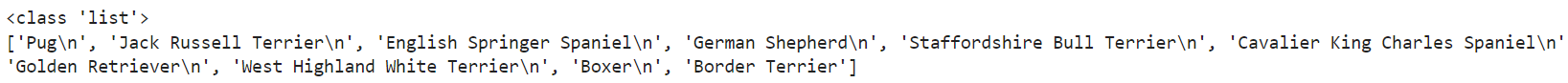
En el siguiente ejemplo vemos como también lo leemos, pero cada línea la guarda en una lista.

**with** open("./data/dog\_breeds.txt","r") **as** open\_file:

all\_text**=**open\_file.readlines()

print(type(all\_text))

print(all\_text)



Aunque a mí el estilo que me gusta es: (no por ello es mejor)

**with** open("./data/dog\_breeds.txt","r") **as** f:

lineas**=**[line.replace("\n","") **for** line **in** f]

lineas

Texto, Carta

Descripción generada automáticamente

Con el texto ya almacenado en una variable o en las que queramos podemos procesarlo. Hagamos un procesamiento simple y escribámoslo.

nuevos\_datos**=**[perrito.replace("Terrier","").replace("Spaniel","") **for** perrito **in** lineas]

nuevos\_datos

Texto

Descripción generada automáticamente

Y ahora para escribir sólo tenemos que cambiar el modo a "w" (sobrescribiremos lo que haya) o "a" (añadiremos al final)

**with** open("./data/new\_breed.txt","w") **as** g:

**for** linea **in** nuevos\_datos:

g.write(linea**+**"\n")

**with** open("./data/new\_breed.txt","r") **as** f:

**for** linea **in** f:

print(linea)

Texto

Descripción generada automáticamente

Hemos visto como abrir, leer y escribir ficheros de texto, con el modificado "b" en los argumentos del open podríamos abrir, leer y escribir ficheros binarios, es decir cualquier tipo de ficheros, pero cómo interpretar lo leído depende de cada fichero, así que por ahora trabaja con ficheros ".txt", hasta que nos adentremos en csv, json y xml en las píldoras siguientes.