

# Tema 6. Ficheros de texto

---

Curso 2021/2022

Programación



## File handling in Python





## Indice

1. Introduccion
2. Abrir un fichero con open
3. Secuencias de lineas y el caracter newline
4. Resolver ejercicios que usan ficheros con ejemplos (abrir-usar-cerrar)
5. Lectura de un fichero de texto: tratamiento de errores
6. Otras formas de lectura
7. Modos de abrir

# 1. Introducción

- Las aplicaciones necesitan guardar información (datos) para poder compartirlos con otras aplicaciones o para reutilizarlos en la misma aplicación.
- Los ficheros son conjuntos de datos residentes en sistemas de almacenamiento secundario (disco duro o cualquier otro tipo de memoria).
- Un tipo particular de ficheros de uso muy extendido son los ficheros de texto. Estos contienen una sucesión de caracteres que podemos considerar organizados en una secuencia de líneas.

# 1. Generalidades sobre ficheros

- Sistemas de ficheros: directorios y ficheros
- Identificación de un fichero: ruta y nombre
  - Todo fichero es identificado de forma única por su ruta (path), es decir el nombre precedido de la descripción del directorio o carpeta donde se ubica. Por ejemplo, `$HOME/Disco/tanja/GTDM/fichero.py`
  - La ruta puede ser absoluta o relativa, en relación con el directorio activo; por ejemplo, si el directorio activo es GTDM, el fichero se identifica también sólo con su nombre `fichero.py`

# 1. Ficheros de texto

- Un archivo de texto se puede considerar como una secuencia de líneas
- Igual que una cadena es una secuencia de caracteres
- Ejemplo mbox.txt

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
Return-Path: <postmaster@collab.sakaiproject.org>
Date: Sat, 5 Jan 2008 09:12:18 -0500
To: source@collab.sakaiproject.org
From: stephen.marquard@uct.ac.za
Subject: [sakai] svn commit: r39772 - content/branches/
```

```
Details: http://source.sakaiproject.org/viewsvn/?view=rev&rev=39772
```

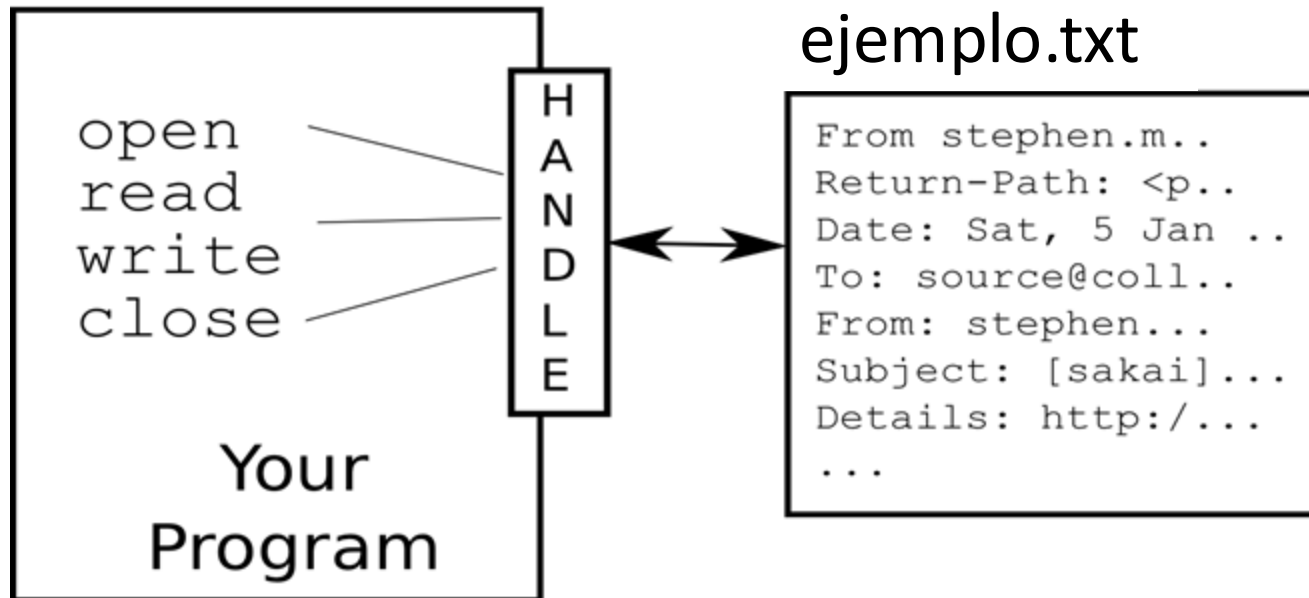
- En un programa los ficheros se manipulan por medio de variables a través de las cuales se puede escribir y/o leer información (las líneas).

## 2. Abrir un fichero: open

```
>>> fhand = open('ejemplo.txt')
>>> print(fhand)

<_io.TextIOWrapper name='ejemplo.txt' mode='r' encoding='UTF-8'>
```

Open() devuelve un «file handle», variable que se usa para realizar operaciones en el archivo.



## 2. Abrir un fichero que no existe

```
>>> fhand = open('algo.txt')
Traceback (most recent call last):
  File "<pyshell>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: 'algo.txt'
```

Luego utilizaremos try-except para poder tratarlo.

### 3. Secuencias de líneas y el carácter «newline»

- Un archivo de texto se puede considerar como una secuencia de líneas
- Se usa un carácter especial llamado «newline» para indicar cuándo acaba una línea
- En las secuencias se representa como “\n”
- Pero newline cuenta como un solo carácter (no dos)

```
>>> stuff = 'Hello\nWorld!'
>>> stuff
'Hello\nWorld!'
>>> print stuff
Hello
World!
>>> stuff = 'X\nY'
>>> print stuff
X
Y
>>> len(stuff)
3
```



## 4. Resolver ejercicios que usan ficheros: abrir – usar - cerrar

### 1. Abrir el fichero

### 2. Usar la información en el fichero (**for-in**) para hacer algo.

### 3. Cerrar el fichero

```
primer-esqueleto.py x
1 #1: Abrir el fichero en file-handle llamado fhand
2 fhand = open('ejemplo.txt')
3
4 #2: Recorrer todas las lineas
5 for line in fhand:
6     #haz algo
7
8 #3: Cerrar el fichero
9 fhand.close()
```

## 4. Ejemplo: contar cantidad de lineas

```
#1: Abrir el fichero en file-handle llamado fhand
fhand = open('ejemplo.txt')

count = 0
#2: Recorrer todas las lineas
for line in fhand:
    #incrementa el counter para contar las lineas
    count = count + 1

print('Numero de lineas: ', count)

#3: Cerrar el fichero
fhand.close()
```

## 4. Ejemplo: buscar en un fichero de texto línea a línea

```
#1: Abrir el fichero en file-handle llamado fhand
fhand = open('mbox.txt')

#2: Recorrer todas las líneas
for line in fhand:
    if (line.startswith('From:')):
        print(line)

#3: Cerrar el fichero
fhand.close()
```

Encontrar en un fichero de texto de correo,  
de quien hemos recibido correos

```
1 #1: Abrir el fichero en file-handle llamado fhand
2 fhand = open('mbox.txt')
3
4 #2: Recorrer todas las lineas
5 for line in fhand:
6     if (line.startswith('From:')):
7         print(line)
8
9 #3: Cerrar el fichero
10 fhand.close()
11
```

Shell 

```
>>> %Run search-text.py
```

```
From: stephen.marquard@uct.ac.za
```

```
From: louis@media.berkeley.edu
```

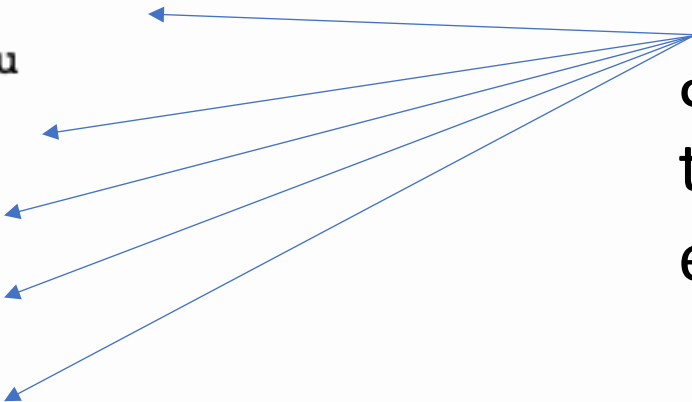
```
From: zqian@umich.edu
```

```
From: rjlowe@iupui.edu
```

```
From: zqian@umich.edu
```

```
From: rjlowe@iupui.edu
```

¿Qué hacen aquí  
todas estas líneas  
en blanco?



- Cada línea del archivo tiene una **newline** al final
- La instrucción `print` añade una **newline** a cada línea

De: stephen.marquard@uct.ac.za\n

\n

De: louis@media.berkeley.edu\n

\n

De: zqian@umich.edu\n

\n

De: rjlowe@iupui.edu\n

\n

...

# rstrip()

- Podemos eliminar el espacio en blanco del lado derecho de la secuencia utilizando **rstrip()** de la biblioteca de string
- La newline se considera como un “espacio en blanco” y se elimina

```
1 #1: Abrir el fichero en file-handle llamado fhand
2 fhand = open('mbox.txt')
3
4 #2: Recorrer todas las lineas
5 for line in fhand:
6     if (line.startswith('From:')):
7         print(line.rstrip())
8
9 #3: Cerrar el fichero
10 fhand.close()
11
```

Shell

>>> %Run search-text.py

```
From: stephen.marquard@uct.ac.za
From: louis@media.berkeley.edu
From: zqian@umich.edu
From: rjlowe@iupui.edu
From: zqian@umich.edu
From: rjlowe@iupui.edu
From: cwen@iupui.edu
From: cwen@iupui.edu
From: gsilver@umich.edu
From: gsilver@umich.edu
From: zqian@umich.edu
From: gsilver@umich.edu
```

## 5. Abrir un fichero de texto: tratamiento de errores

```
contar-lineas.py ✕  
1 filename = input('Introduzca un nombre de fichero: ')  
2 try:  
3     #1: Abrir el fichero en file-handle llamado fhand  
4     fhand = open(filename)  
5  
6     count = 0  
7     #2: Recorrer todas las lineas  
8     for line in fhand:  
9         #incrementa el counter para contar las lineas  
10        count = count + 1  
11  
12    print('Numero de lineas: ', count)  
13  
14    #3: Cerrar el fichero  
15    fhand.close()  
16  
17 except FileNotFoundError:  
18    print("El fichero no existe")
```



## 5. Esqueleto 2: abrir – usar - cerrar

segundo-esqueleto.py

```
1 filename = input('Introduzca un nombre de fichero: ')
2 try:
3     #1: Abrir el fichero en file-handle llamado fhand
4     fhand = open(filename)
5
6     #2: Recorrer todas las lineas
7     for line in fhand:
8         #haz algo
9
10    #3: Cerrar el fichero
11    fhand.close()
12
13 except FileNotFoundError:
14     print("El fichero no existe")
```

## 5. Ejemplo

```
1 import random
2
3 filename = "llista.txt"
4 numero_alumnos = 50
5 try:
6     #1: Abrir el fichero en file-handle llamado fhand
7     fhand = open(filename)
8
9     contador = 0
10    num = int(random.uniform(0, numero_alumnos-1))
11
12    #2: Recorrer todas las lineas
13    for line in fhand:
14        if (contador == num):
15            print(line)
16            exit()
17        contador = contador + 1
18
19    #3: Cerrar el fichero
20    fhand.close()
21
22 except FileNotFoundError:
23     print("El fichero no existe")
```

## 6. Otra forma de lectura con **readline**

segundo-esqueleto-con-readline.py ✕

```
1 filename = input('Introduzca un nombre de fichero: ')
2 try:
3     #1: Abrir el fichero en file-handle llamado fhand
4     fhand = open(filename)
5
6     #2: Recorrer todas las lineas
7     line = fhand.readline()
8     while line != '':
9         #haz algo
10
11         #coge la siguiente linea
12         line = fhand.readline()
13
14     #3: Cerrar el fichero
15     fhand.close()
16
17 except FileNotFoundError:
18     print("El fichero no existe")
```

La función **readline** devuelve la cadena vacía al detectar el final del fichero.

## 6. Lectura de un fichero carácter a carácter con **read(i)**

```
contar-caracteres.py ✕
1 filename = input('Introduzca un nombre de fichero: ')
2 try:
3     #1: Abrir el fichero en file-handle llamado fhand
4     fhand = open(filename)
5
6     count = 0
7     ch = fhand.read(1)
8     #2: Recorrer todas las líneas
9     while ch != '':
10         #incrementa el counter para contar las caracteres
11         count = count + 1
12         ch = fhand.read(1)
13
14     print('Numero de caracteres: ', count)
15
16     #3: Cerrar el fichero
17     fhand.close()
18
19 except FileNotFoundError:
20     print("El fichero no existe")
```

**read(i)** devuelve una cadena con **i** caracteres. Devuelve la cadena vacía llegando al final del fichero.

ejercicios 1 hasta 4 del reader

## 6. Otras formas de lectura con `read()`

- La función `read()` devuelve todo el contenido del fichero como una cadena de texto.
- Y con funciones de string podemos buscar, contar, transformar.
- Pero... solo hay usar esto cuando el fichero no es muy grande.

```
>>> file = open("message.txt", "r")
      content = file.read()
      print(content)
```

```
This is a secret message
```

```
>>> |
```

```
This is a secret message
```

message.txt

Platte-tekstdocument - 24 bytes

## 4. Modos de abrir ficheros

**1. Abrir el fichero** indicando su **ruta** y el **modo** de trabajo:

- **Lectura**: sólo se leerá información del fichero, no se podrá modificar ni añadir.
- **Escritura**: sólo se podrá escribir información en el fichero. En general, este modo implica borrar el contenido previo del fichero.
- **Lectura/Escritura**: permite leer y escribir.
- **Adición**: permite añadir nueva información al fichero pero no modificar la información ya existente.

**2. Leer o escribir** la información para hacer algo.

**3. Cerrar** el fichero

## 7. Añadir texto a un fichero: modo **a**

- Programa que lee el nombre del fichero de notas (texto) del teclado y una nota y la añade al fichero.

```
nomFichero = input('Escribe el nombre del fichero de notas: ')
fichero = open(nomFichero, 'a')
nota = input('Nota a añadir: ')
fichero.write(nota + '\n')
fichero.close()
```



## 7. Escritura en un fichero de texto: modo **w** y función **write**

Programa que lee el nombre del fichero de texto del teclado y genera otro fichero con el texto codificado de forma que sólo las letras minúsculas se sustituyen por las siguientes.

codificar-f2f.py

```
1 f_input = input('Introduzca un nombre de fichero de entrada: ')
2 f_output = input('Introduzca un nombre de fichero de salida: ')
3 try:
4     #1: Abrir el fichero en file-handle llamado fhand
5     fhand_input = open(f_input, 'r')
6     fhand_output = open(f_output, 'w')
7
8     ch = fhand_input.read(1)
9     #2: Recorrer todas las lineas
10    while ch != '':
11        #haz la codificación
12        if ch >= 'a' and ch <= 'y':
13            ch = chr(ord(ch) + 1)
14        elif ch == 'z':
15            ch = 'a'
16
17        fhand_output.write(ch) #was: print(ch, end='')
18        ch = fhand_input.read(1)
19
20    #3: Cerrar el fichero
21    fhand_input.close()
22    fhand_output.close()
23
24 except FileNotFoundError:
25     print("El fichero no existe")
```

## 7. Ejemplo: generar ficheros de texto con los números del 1 al 5000 y sus cuadrados

numeros.py ✕

```
1  # Abriendo un fichero en modo write
2  # Si el fichero ya existe, todos sus datos se eliminan
3  # Si el fichero no existe, se crea una nueva
4  fichero = open("numeros.txt", 'w')
5
6  for i in range(1, 5001):
7
8      #una manera
9      fichero.write(str(i)+" ")
10     fichero.write(str(i*i)+"\n")
11
12     #otra manera
13     fichero.write(str(i) + " " + str(i*i) + "\n")
14
15     #otra forma
16     fichero.write("{0} {1}\n".format(i, i*i))
17
18 fichero.close()
19
```

## 7. Ejemplo: generar un fichero de texto con la tabla de un numero dado

tabla-n.py

```
1 #Pide un numero n al usuario entre 1 y 10, y genera un fichero de texto con
2 # nombre tabla-n.txt que contiene la tabla del numero dado
3
4 #pedir numero n
5 n = int(input('Introduce un número entero entre 1 y 10: '))
6
7 #crear el fichero con el nombre correcto
8 file_name = 'tabla-' + str(n) + '.txt'
9 f = open(file_name, 'w')
10
11 #escribir la tabla en el fichero
12 for i in range(1, 11):
13     f.write(str(n) + ' x ' + str(i) + ' = ' + str(n * i) + '\n')
14
15 #cerrar el fichero
16 f.close()
```

Shell

```
>>> %Run tabla-n.py
```

```
Introduce un número entero entre 1 y 10: 8
```

```
>>>
```

tabla-8.txt

```
8 x 1 = 8
8 x 2 = 16
8 x 3 = 24
8 x 4 = 32
8 x 5 = 40
8 x 6 = 48
8 x 7 = 56
8 x 8 = 64
8 x 9 = 72
8 x 10 = 80
```

ejercicios 5 hasta 9 del reader