READING & WRITING FILES

Arne Covens & Kevin Kerselaers

INTRO

```
>>> import os
>>> os.path.join('usr', 'bin', 'spam')
'usr\\bin\\spam'
```

- Import OS
  - Nodige functies importeren
- Os.path.join(strings)
  - Strings omzetten naar een correct pad

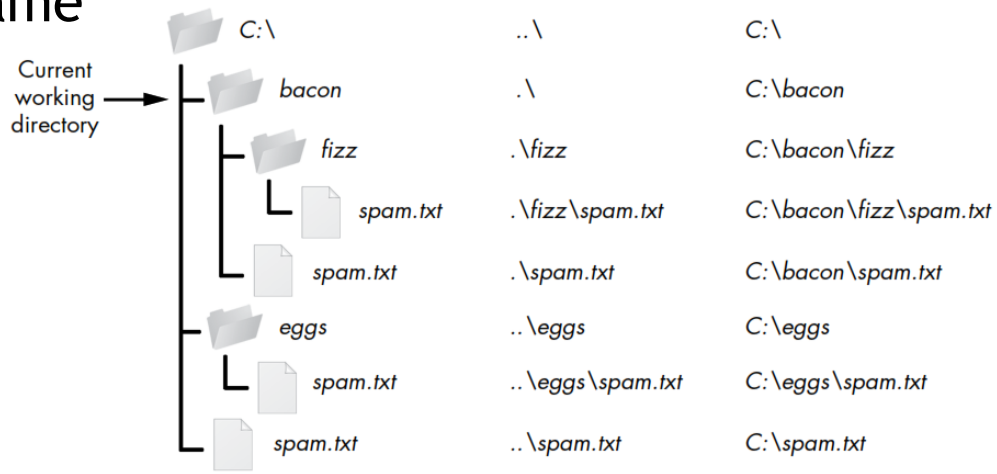
BASIC FUNCTIONS

- Os.getcwd()
  - Printen van huidige working directory
- OS.chdir(path)
  - Het veranderen van de working directory
  - Dubbele '\' om de slash de escapen

```
>>> import os
>>> os.getcwd()
'C:\\Python34'
>>> os.chdir('C:\\Windows\\System32')
>>> os.getcwd()
'C:\\Windows\\System32'
```

RELATIVE VS ABSOLUTE

- Relative
  - Current working directory
  - ./spam.txt & spam.txt same
- Absolute
  - From the root folder
- ..\ Vs .\
  - ..\ → parent folder → één folder naar boven
  - .\ → deze folder



REL VS ABS

- Os.path.abspath(path)
  - Geeft absolute path weer van het gegeven pad
- Os.path.isabs(path)
  - True als absolute, False als relative

```
>>> os.path.abspath('.')
'C:\\Python34'
>>> os.path.abspath('.\\Scripts')
'C:\\Python34\\Scripts'
>>> os.path.isabs('.')
False
>>> os.path.isabs(os.path.abspath('.'))
True
```

REL VS ABS

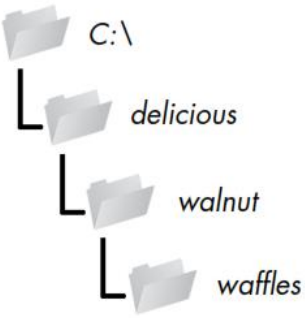
- Os.path.relpath(gotopath, wdpath)
  - Geeft het relatieve pad weer van een directory naar een directory

```
>>> os.path.relpath('C:\\Windows', 'C:\\')
'Windows'
>>> os.path.relpath('C:\\Windows', 'C:\\spam\\eggs')
'..\\..\\Windows'
>>> os.getcwd()
'C:\\Python34'
```

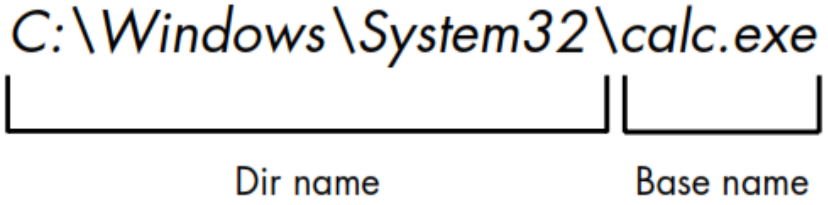
MAKE DIR

```
>>> import os
>>> os.makedirs('C:\\delicious\\walnut\\waffles')
```

- Os.makedirs(string)
  - Maken van alle nodige folders (Let op: ALLE)



BASENAME VS DIRNAME



- Os.path.basename(path)
  - Geeft alleen de basename weer
- Os.path.dirname(path)
  - Geeft alleen de dirname weer

```
>>> path = 'C:\\Windows\\System32\\calc.exe'
>>> os.path.basename(path)
'calc.exe'
>>> os.path.dirname(path)
'C:\\Windows\\System32'
```

BASENAME VS DIRNAME

- Os.path.split(path)
  - Slaagt zowel de basename als de dirname op in een tuple

```
>>> calcFilePath = 'C:\\Windows\\System32\\calc.exe'
>>> os.path.split(calcFilePath)
('C:\\Windows\\System32', 'calc.exe')
```

- Os.path.sep
  - In gebruik met .split, kan je zo het hele pad opdelen in kleine stukken

```
>>> calcFilePath.split(os.path.sep)
['C:', 'Windows', 'System32', 'calc.exe']
```

FILE SIZES, FOLDER CONTENTS

- Os.patch.getsize(path)
  - Geeft de grootte van een bestand weer in bytes
- Os.listdir(path)
  - Geeft alle bestanden weer van waar het pad naar verwijst

```
>>> os.path.getsize('C:\\Windows\\System32\\calc.exe')
776192
>>> os.listdir('C:\\Windows\\System32')
['0409', '12520437.cpx', '12520850.cpx', '5U877.ax', 'aaclient.dll',
--snip--
'xwtpdui.dll', 'xwtpw32.dll', 'zh-CN', 'zh-HK', 'zh-TW', 'zipfldr.dll']
```

IS FILE OR DIR

```
>>> os.path.exists('C:\\Windows')
True
>>> os.path.exists('C:\\some_made_up_folder')
False
>>> os.path.isdir('C:\\Windows\\System32')
True
>>> os.path.isfile('C:\\Windows\\System32')
False
>>> os.path.isdir('C:\\Windows\\System32\\calc.exe')
False
>>> os.path.isfile('C:\\Windows\\System32\\calc.exe')
True
```

- Os.path.exists(path)
  - Kijkt of het pad bestaat
- Os.path.isdir(path)
  - Kijkt of het gegeven pad een folder is
- Os.path.isfile(path)
  - Kijkt of het gegeven pad een file is

OPENING FILES

- Filepointer = Open(path)
  - Opent een file in standaard lees modus (en wijst deze aan een var)

```
>>> helloFile = open('C:\\Users\\your_home_folder\\hello.txt')
```

- Filepointer = Open(path, modus)
  - Opent een file in een gegeven modus (en wijst deze aan een var)

```
>>> baconFile = open('bacon.txt', 'w')
```

OPEN MODUSES

Modus	Uitleg
r	- Standaard modus - Alleen lezen, begin van bovenaan bestand - Error als bestand niet bestaat
r+	- Zowel lezen als schrijven - Zet cursor bovenaan de file - Error als bestand niet bestaat
w	- Schrijven naar file - Overschrijft bestaand bestand - Maakt nieuw bestand als er nog geen bestaat
W+	- Zowel lezen als schrijven - Zet cursor bovenaan de file - Overschrijft bestaand bestand
a	- Append mode - Schrijft data op het einde bij van een file
a+	- Zowel lezen als schrijven - Maakt nieuw bestand als er nog geen bestaat - Schrijft data op het einde bij van een file

READING FILES

- Testvar = filepointer.read()
  - Leest alle content van een bestand naar één variable

```
>>> helloContent = helloFile.read()
>>> helloContent
'Hello world!'
```

- Filepointer.readlines()
  - Leest alle content van een file en zet deze in een lijst
  - Elke nieuwe lijn is een is een value in de lijst

```
>>> sonnetFile = open('sonnet29.txt')
>>> sonnetFile.readlines()
[When, in disgrace with fortune and men's eyes,\n', ' I all alone beweep my
outcast state,\n', And trouble deaf heaven with my bootless cries,\n', And
look upon myself and curse my fate,']
```

READING FILES

- Filepointer.readline()
  - Leest één lijn uit een file

```
>>> f.readlines()
['First line of our text.\n', 'Second line of our text.\n', '3rd line, one line is trailing.\n']
>>> f.readline()
'First line of our text.\n'
```

WRITING FILES

- Filepointer.write(string)
  - Schrijft de string naar de file
- Filepointer.close()
  - Sluit de file sessie

```
>>> baconFile = open('bacon.txt', 'w')
>>> baconFile.write('Hello world!\n')
13
>>> baconFile.close()
>>> baconFile = open('bacon.txt', 'a')
>>> baconFile.write('Bacon is not a vegetable.')
25
>>> baconFile.close()
>>> baconFile = open('bacon.txt')
>>> content = baconFile.read()
>>> baconFile.close()
>>> print(content)
Hello world!
Bacon is not a vegetable.
```

WRITING VARIABLES TO FILE

- Om variablen bij te houden, ook nadat de sessie (shell) gesloten is,

- Twee opties
  - The shelve module (Binair wegschrijven)
  - Pretty Printing (leesbaar wegschrijven)

THE SHELVE MODULE SCHRIJVEN

- Exporteren en importeren van variable en hun values
- Import shelve
  - Importeren van shelve functies
- shelvepointer = shelve.open(‘shelvenaam’)
  - Het openen van een pointer naar een shelve bestand
- shelvepointer[‘key’] = value
  - Net zoals in een directory heb je hier key en value waarden, je schrijft deze weg naar de shelve file
- shelvepointer.close()
  - Het sluiten van de shelve file

```
>>> import shelve
>>> shelfFile = shelve.open('mydata')
>>> cats = ['Zophie', 'Pooka', 'Simon']
>>> shelfFile['cats'] = cats
>>> shelfFile.close()
```

THE SHELVE MODULE LEZEN

- Shelvepointer = shelve.open(‘shelvenaam’)
  - Het openen van de shelve
- Type(shelvepointer)
  - Het type laten zien van de shelvepointer
- Shelvepointer(‘key’)
  - De value laten zien die geassocieerd zijn met de gegeven key

```
>>> shelfFile = shelve.open('mydata')
>>> type(shelfFile)
<class 'shelve.DbfilenameShelf'>
>>> shelfFile['cats']
['Zophie', 'Pooka', 'Simon']
>>> shelfFile.close()
```

THE SHELVE MODULE

- List(shelvepointer.keys())
  - Het weergeven van alle keys die zich in de shelvefile bevinden (in lijst vorm)
- List(shelvepointer.values())
  - Het weergeven van alle values die zich in de shelvefile bevinden (in lijst vorm)

```
>>> shelfFile = shelve.open('mydata')
>>> list(shelfFile.keys())
['cats']
>>> list(shelfFile.values())
[['Zophie', 'Pooka', 'Simon']]
>>> shelfFile.close()
```

PRETTY PRINTING

- Import pprint
  - Importeren van pretty printing functies
- Var cats
  - Lijst met twee dictionaries in die je wilt wegschrijven naar een .py file
- Pprint.pformat(variable)
  - Het printen van een variable met PP

```
>>> import pprint
>>> cats = [{'name': 'Zophie', 'desc': 'chubby'}, {'name': 'Pooka', 'desc': 'fluffy'}]
>>> pprint.pformat(cats)
"[{'desc': 'chubby', 'name': 'Zophie'}, {'desc': 'fluffy', 'name': 'Pooka'}]"
>>> fileObj = open('myCats.py', 'w')
>>> fileObj.write('cats = ' + pprint.pformat(cats) + '\n')
83
>>> fileObj.close()
```

PRETTY PRINTING

- Filepointer = open(‘bestand.py’, ‘w’)
  - Een file openen in write modus, om een variable weg te schrijven gebruiken we de .py extentie, die elk python bestand gebruikt

```
>>> import pprint
>>> cats = [{'name': 'Zophie', 'desc': 'chubby'}, {'name': 'Pooka', 'desc': 'fluffy'}]
>>> pprint.pformat(cats)
"[{'desc': 'chubby', 'name': 'Zophie'}, {'desc': 'fluffy', 'name': 'Pooka'}]"
>>> fileObj = open('myCats.py', 'w')
>>> fileObj.write('cats = ' + pprint.pformat(cats) + '\n')
83
>>> fileObj.close()
```

PRETTY PRINTING

- Import ‘filename’
  - Het importeren van de file naar je huidige sessie
- Filename.variable
  - De variable afprinten die zich in het extern bestand bevind

```
>>> import myCats
>>> myCats.cats
[{'name': 'Zophie', 'desc': 'chubby'}, {'name': 'Pooka', 'desc': 'fluffy'}]
>>> myCats.cats[0]
{'name': 'Zophie', 'desc': 'chubby'}
>>> myCats.cats[0]['name']
'Zophie'
```

QUIZ

1. What is a relative path relative to ?

2. What does an absoulute path start with ?

3. What do the os.getcwd() and os.chdir() functions do?

4. What are the ‘.’ and ‘..’ folders ?

QUIZ

5. In ‘C:\bacon\eggs\spam.txt, which part is de dir name, and which part is the base name ?

6. What happens if an existing file is opened in write mode ?

7. What is the difference between the read() and the readlines() methods ?

QUIZ

8. What data structure does a shelf value resemble ?

QUESTIONS & PRACTISE

- Vragen ?

- Oefeningen te vinden op Trac