Féidearthachtaí as Cuimse
Infinite Possibilities

Programming for Analytics

Lecture 4: Working with Files

Bojan Božić
School of Computer Science
TU Dublin, Grangegorman

bojan.bozic@tudublin.ie

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
TU DUBLIN
TECHNOLOGICAL UNIVERSITY DUBLIN

Thursday Labs moved to CQ230

Overview

- Opening and Closing Files
- Reading Files Line-by-Line
- Writing to Files
- CSV Files
- JSON Files
- Error Handling

Why Work with Files?

- Files allow data persistence beyond a single session
- Common for storing data logs, configuration, and inputs/outputs
- Text files, CSV, and JSON are standard formats in data analytics

Opening and Closing Files

- Use `open(filename, mode)` to access a file
- Modes: `'r'` = read, `'w'` = write, `'a'` = append, `'x'` = create new
- Always close files using `file.close()` or use `with` statement

Example

```
with open('file.txt', 'mode') as file:
    lines = file.readlines()
```

Activity: Read a Simple Text File

- Create a text file with 3 lines
- Write a Python script to read and print all lines using `with open(…) as f:`

Reading Files Line-by-Line

- Use `.readline()`,`.readlines()`, or iterate with a for loop
- Useful for processing logs or structured files line-by-line
- Avoid loading large files entirely into memory if possible

Example

```
with open(‘file.txt’, ‘r’) as f:
    print(f.readline())
    print(f.readline())
    …
```

Activity: Count Lines in a File

- Read a text file and count how many lines it contains

Writing to Files

- Use `‘w’` to overwrite or `‘a’` to append
- Use `file.write()` or `file.writelines()`
- Always remember to close or use `with` block

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
DUBLIN
TECHNOLOGICAL UNIVERSITY DUBLIN

Example

```
with open('untitled2.txt', 'w') as f:
    f.write('A tale of the Malazan
      Book of the Fallen.')
```

Activity: Write User Input to File

- Prompt the user for a few lines of input.
- Write them to a file, line per line.

Using `with` to Manage Files

- `With open(filename) as f:` automatically closes the file
- Cleaner and safter than manually calling `close()`
- Recommended for most use cases

Working with CSV Files

- CSV = Comma Separated Values (simple spreadsheet-like format)
- Use the built-in `csv` module to read/write CSV files
- Rows are read as lists or dictionaries

Example

```
import csv
with open(‘book_characters.csv’)
as csvfile:
  reader = csv.reader(csvfile)
```

Activity: Read a CSV and Print Rows

- Use `csv.reader` to read a file and print each row
- Hint: a for loop through the reader object will return rows one by one.

Writing CSV Files

- Use `csv.writer` for writing lists to rows
- Use `csv.DictWriter` for writing dictionaries
- Specify headers and fieldnames when using `DictWriter`

Example

```
with open(‘csvfile’, ‘w’) as
csvfile:
    csvwriter = csv.writer(csvfile)
    csvwriter.writerow(['Id’,
    'Name', 'Job'])
```

Activity: Save Grades to CSV

- Write student name and grade data into a CSV file (at least 3 students).

Working with JSON Files

- JSON – JavaScript Object Notation (lightweight data format)
- Use built-in `json` module to `load()` and `dump()` data
- Maps neatly to Python dicts and lists

Example

```
import json
student = {"name": "Alice", "age": 28, "grades": [80, 75, 82]}
with open("students.json", "w") as json_file:
    json.dump(student, json_file)
```

Activity

- Load a JSON file with student data
- Update or add a field and write it back

Error Handling with Files

- Always anticipate FileNotFoundError, IOError etc.
- Use try/except to handle errors and print user friendly messages
- Check if file exists using `os.path.exists()`

Example

```
import json, os
if not os.path.exists(“students.json”):
    print("The file does not exist.")
else:
   try:
      with open(filename, "r") as f:
      data = json.load(f)
    except IOError:
      print(”Couldn’t read the file.")
```

Activity: Open File Safely

- Prompt for a filename and handle the case where it doesn’t exist

Working with Paths

- Use `os.path.join` for cross-platform paths
- Use `pathlib.Path` for modern object-oriented path handling
- Avoid hardcoding file paths for portability

Example

```
file_path_os =
os.path.join(data_dir, filename)
base_dir = Path.cwd() / "data"
file_path_pathlib = base_dir /
filename
```

Activity: Create Folder and File

- Use `pathlib` to create a folder and write a file inside it.

Summary of File I/O Tools

- `open()`, `close()`, `with`, `read()`, `write()`
- `csv.reader/writer`, `json.load/dump`
- `try/except` for errors, `pathlib` for paths

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
DUBLIN
TECHNOLOGICAL UNIVERSITY DUBLIN

Final Practical Challenge

- Read a CSV of students, calculate average grades, and write summary to a new file.
- Store full student data in JSON with extra fields and save it.

Next Week

- Working with external data: APIs and Web Requests.
- Intro to data analysis tools (NumPy, pandas preview)

Questions?

OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH
DUBLIN
TECHNOLOGICAL UNIVERSITY DUBLIN