Reading, managing, and exporting CSV files

JSON (JavaScript Object Notation) is a lightweight data-interchange format commonly used for storing and exchanging data. Python provides a built-in json module to handle JSON data easily.

Reading JSON in Python

- Use json.load() to read from a JSON file and convert it into Python objects (usually dictionaries or lists).
- Use json.loads() to parse a JSON string into Python objects.
  Example:

python

```python
import json

# Reading from a JSON file
with open('data.json', 'r') as file:
  data = json.load(file)
print(data)

# Parsing JSON string
json_str = '{"name": "John", "age": 30}'
parsed_data = json.loads(json_str)
print(parsed_data)
```

Writing JSON in Python

- Use json.dump() to serialize Python objects (e.g., dictionaries) into JSON format and write directly to a file.
- Use json.dumps() to convert Python objects into a JSON formatted string.
  Example:

python

```python
import json

data = {"name": "Jane", "age": 25}

# Writing to a JSON file
with open('output.json', 'w') as file:
  json.dump(data, file, indent=4)

# Convert to JSON string
json_str = json.dumps(data, indent=4)
print(json_str)
```

Key points

- Python objects and JSON types map closely: Python dict = JSON object, list/tuple = array, str = string, int/float = number, True/False = true/false, None = null.
- JSON supports nested structures.
- Indentation in dump and dumps helps make output human-readable.
- JSON is widely used in web applications, APIs, config files, and data exchange.

This makes Python's json module a powerful and straightforward tool for reading, parsing, writing, and generating JSON data

Working with Excel files in Python

Working with Excel files in Python can be done using libraries like openpyxl, pandas, and xlsxwriter. These libraries allow reading, writing, and modifying Excel spreadsheets efficiently.

Reading Excel Files

Using openpyxl:

python

```
import openpyxl
wb = openpyxl.load_workbook('file.xlsx')
sheet = wb.active
print(sheet['A1'].value)  # Read value from cell A1
```

Using pandas:

python

```
import pandas as pd
df = pd.read_excel('file.xlsx')
print(df.head())  # Display first few rows
```

Writing Excel Files

Using openpyxl to create and write:

python

```
from openpyxl import Workbook
wb = Workbook()
sheet = wb.active
sheet['A1'] = 'Hello'
sheet['B1'] = 'World'
wb.save('output.xlsx')
```

Using pandas to write DataFrame to Excel:

python

```
df.to_excel('output.xlsx', index=False)
```

Other Libraries

- xlsxwriter is used to create Excel files with advanced formatting, charts, and formulas.
- Python can also integrate with Excel using add-ins or Microsoft's new Python integration in Excel (enabling Python code inside Excel cells).

Summary

- openpyxl is suited for reading and editing Excel (.xlsx) files at cell level.
- pandas offer powerful tools for handling Excel files as structured DataFrames for analysis and manipulation.
- For writing complex sheets with formatting, xlsxwriter is preferred.
- Python integration within Excel allows using Python code directly inside Excel workbooks.

These tools collectively offer versatile options for automation, data analysis, and Excel file manipulation within Python environments.

NumPy (Numerical Python)

NumPy (Numerical Python) is a fundamental Python library for numerical computing. It provides fast and efficient multi-dimensional array objects (ndarrays) and a wide range of mathematical functions essential for handling large datasets and performing complex numerical calculations.

Key Features of NumPy for Numerical Computing

- N-dimensional arrays (ndarray): A powerful data structure for homogeneous data stored in contiguous memory for fast access and operations.
- Vectorized operations: Enables element-wise mathematical computations on entire arrays without explicit loops, leading to concise and efficient code.
- Broadcasting: Allows arithmetic operations between arrays of different shapes by automatically aligning their dimensions.
- Mathematical functions: Includes a vast collection of functions such as trigonometric, statistical, exponential, linear algebra routines, and more.
- Array indexing and slicing: Similar to Python lists but offers more advanced multi-dimensional and boolean indexing.
- High performance: Implements array operations in compiled code (C/Fortran), making computations much faster than standard Python lists.

Installation

bash

```
pip install numpy
```

Importing NumPy

python

```
import numpy as np
```