Common Python Libraries

Contents

- Standard libraries
- Sockets
- NumPy and MatPlotLib
- NetCDF
- Pandas data processing
- Database access

Demo folder: **10-CommonLibraries**

Standard Library Modules

- Python has an extensive suite of standard library modules
- For example:
  - `os`
  - `sys`
  - `datetime`
  - `random`
  - etc.
- For full details, see:
  - https://docs.python.org/3.8/library

Sockets (1 of 3)

- Python has a sockets API
  - Via the socket module
  - Allows you to implement low-level client/server applications, running over TCP/IP
- For an example, see the following demo folder:
  - `Sockets`
- Open two Command Prompt windows
  - Run `server.py` in the first window
  - Run `client.py` in the other window

Sockets (2 of 3)

- Here's the code for the server application
  - See the comments in `server.py` for details

```
import socket

socket = socket.socket()
socket.bind(('localhost', 8001))

socket.listen(500)
print('Server listening on port 8001...')

while True:

    conn, addr = socket.accept()

    while True:
        databytes = conn.recv(1024).decode()
        datastr = str(databytes)
        if datastr == 'quit':
            print('Client asked to quit, so connection closed')
            conn.close()
            break
        else:
            print('Data received from client: %s' % datastr)
            conn.send(datastr.upper().encode())
```

server.py

Sockets (3 of 3)

- Here's the code for the client application
  - See the comments in `client.py` for details

```
import socket

socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
socket.connect(('localhost', 8001))

while True:
    data = input('Data to send [or "quit"]: ')
    socket.send(bytes(data, 'UTF-8'))
    if data == 'quit':
        socket.close()
        break
    else:
        print(socket.recv(1024).decode())
```

client.py

NumPy and MatPlotLib (1 of 2)

- NumPy is a numeric processing API for Python
  - Complex maths, matrix processing, etc.
- MatPlotLib is a graphical plotting API for Python
  - Similar to Matlab, allows you to plot graphs, charts, etc.
- NumPy and MatPlotLib aren't included in the core Python installation - you must install using PIP, as follows:

```
pip install matplotlib
```

- Important note: If you get errors installing matplotlib, try the following command instead

```
pip install --only-binary :all: matplotlib
```

NumPy and MatPlotLib (2 of 2)

- For examples of how to use NumPy and MatPlotLib, see the following demo folder:
  - `NumPy and Matplotlib`

- There are several examples, to illustrate simple techniques
  - `Ex1-Versions.py`
    Simply displays the library version numbers
  - `Ex2-NumPy.py`
    Shows simple examples of number crunching
  - `Ex3-Plot.py`
    Shows simple examples of how to plot charts

NetCDF

- NetCDF is a popular set of data formats widely used in the scientific data arena

- You can install the NetCDF library as follows:

```
pip install netCDF4
```

- We've provided several examples that show how to use the NetCDF library in Python
  - See the `NetCDF` demos folder

Pandas (1 of 2)

- Pandas is a data processing API for Python
  - Reading and writing CSV, Excel spreadsheets, JSON, etc.
  - Performing data cleansing, data analysis, etc.

- Pandas isn't included in the core Python installation
  - You must install several packages using PIP, as follows:

```
pip install pandas
```

```
pip install openpyxl
```

```
pip install xlrd
```

Pandas (2 of 2)

- For examples of how to use Pandas, see the following demo folder:
  - `Pandas`

- There are several examples, to illustrate simple techniques
  - `Ex1-WriteCsv.py`
  - `Ex2a-ReadOneCsv.py, Ex2b-ReadMultipleCsv.py`
  - `Ex3-PlotCsv.py`
  - `Ex4-GroupBy.py`
  - `Ex5-WriteExcel.py`
  - `Ex6-ReadExcel.py`
  - `Ex7-CleanData.py`
  - `Ex8-ManipulateColumns.py`

Database Access - MySQL (1 of 2)

- Many applications need to access relational or NoSQL databases
  - There are Python libraries for all common databases

- For example:
  - If you want to access a MySQL database, install the MySQL Connector library as follows
  - Enables a Python application to connect to MySQL databases and to execute queries

```
pip install mysql-connector-python
```

Database Access - MySQL (2 of 2)

- For examples of how to connect to a MySQL database and execute queries, see the following demo folder:
  - `Database Access`

- There are several examples:
  - `Ex1-SimpleQuery.py`
  - `Ex2-FullQuery.py`
  - `Ex3-Update.py`
  - `Ex4-Insert.py`
  - `Ex5-Delete.py`

Database Access - General

- In general, you can use the `pyodbc` package to connect to any relational database via ODBC

```
pip install pyodbc
```

- E.g. see here in the `Database Access` folder:
  - `Ex6-ConnectToSqlServer.py`

- What the example does:
  - Connects to a LocalDB database called `MySampleDatabase`
  - Executes a query and processes rows manually
  - Uses Pandas to execute a query and load data into a DataFrame
  - Also outputs the data to a CSV file

Any Questions?