ICT233
Data Programming

Seminar 1

Dr. Kelvin DU
zddu001@suss.edu.sg

SUSS
SINGAPORE UNIVERSITY
OF SOCIAL SCIENCES

Course Structure

13-Jan-26 to 31-Mar-26: 6 lectures (3 hrs) & 6 labs (2 hrs)

S/N	Topic	Date & Time	Venue
1	Introduction to Data Programming	7pm to 10pm, Tue, 13-Jan-26	HQ BLK C-SR.C.4.16
2	Lab	7pm to 9pm, Tue, 20-Jan-26	ICT233_JAN26_T01 -> Virtual Class -> Zoom
3	Data Management	7pm to 10pm, Tue, 27-Jan-26	HQ BLK C-SR.C.4.16
4	Lab	7pm to 9pm, Tue, 3-Feb-26	ICT233_JAN26_T01 -> Virtual Class -> Zoom
5	Data Types and Structure	7pm to 10pm, Tue, 10-Feb-26	HQ BLK C-SR.C.4.16
6	**Lab (Rescheduled due to CNY)**	**10am to 12pm, Sat, 21-Feb-26**	**ICT233_JAN26_T01 -> Virtual Class -> Zoom**
7	Data Manipulation	7pm to 10pm, Tue, 24-Feb-26	HQ BLK C-SR.C.4.16
8	Lab	7pm to 9pm, Tue, 3-Mar-26	ICT233_JAN26_T01 -> Virtual Class -> Zoom
9	Data Munging	7pm to 10pm, Tue, 10-Mar-26	HQ BLK C-SR.C.4.16
10	Lab	7pm to 9pm, Tue, 17-Mar-26	ICT233_JAN26_T01 -> Virtual Class -> Zoom
11	Data Scraping	7pm to 10pm, Tue, 24-Mar-26	HQ BLK C-SR.C.4.16
12	Lab	7pm to 9pm, Tue, 31-Mar-26	ICT233_JAN26_T01 -> Virtual Class -> Zoom

3 pre-class quizzes (PCQ), 1 quiz, 1 Tutor-Marked Assignment (TMA) and 1 ECA:
(Please always refer to CANVAS for the latest update!)

Assessment	Description	Weight Allocation	Deadline
Quiz	PCQ 1	6%	06 Jan 2026 - 13 Jan 2026, 2355hrs
	PCQ 2		27 Jan 2026 - 3 Feb 2026, 2355hrs
	PCQ 3		24 Feb 2026 - 3 Mar 2026, 2355hrs
	Quiz		28 March 2026 (0900hrs) to 30 March 2026 (2355 hrs)
Assignment	TMA	24%	Thursday, 12 March 2026, 2355 hours
Examination	ECA	70%	Start of ECA: Thursday, 02 April 2026, 12 noon End of ECA: Sunday, 05 April 2026, 12 noon End of ECA Grace Period: Monday, 06 April 2026, 12 am
TOTAL		**100%**	

SUSS
SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

Learning Outcomes

By the end of this course, you should be able to:

Knowledge & Understanding (Theory Component)

- Develop analytic mindset to understand and interpret datasets
- Analyze HTTP and design parsing methods for information retrieval
- Apply Object-Relational Mapping (ORM) to manage information between objects and databases
- Compose and utilize query languages for information retrieval from databases

Key Skills (Practical Component)

- Perform ETL (Extraction, Transformation, Loading) and calculations using Python and Pandas
- Develop programs to perform CRUD operations on database information
- Formulate communication methods for exchanging information over the WWW
- Conduct effective data visualization

Learning Resource

- **Official Slides**

Canvas ICT233 L01 Group under Modules

- **Additional Materials**

Canvas ICT233 T01 Group under Modules

- **Study Guide**

ICT233 Study Guide

- **Recommended Textbook**

Dale, K. (2022). Data Visualization with Python and JavaScript: Scrape, Clean, Explore, and Transform Your Data. " O'Reilly Media, Inc.".

Severance, C. (2016). Python for everybody: Exploring Data using python 3. Charles Severance.

- **Tips!**
  1. Study with the slides first!
  2. If you wish to deepen your understanding, go for the study guide and attempt formative assessments
  3. If you really want to achieve the best, go for the recommended textbook which includes all the details

Plagiarism and Collusion

- SUSS views academic plagiarism/collusion as a serious matter and actions will be taken on both parties in plagiarism/collusion

- Submitted assignments shall go through SUSS plagiarism detection software

- We go extra miles to catch the cases, so DO NOT try ☺

Introduction to Data Programming

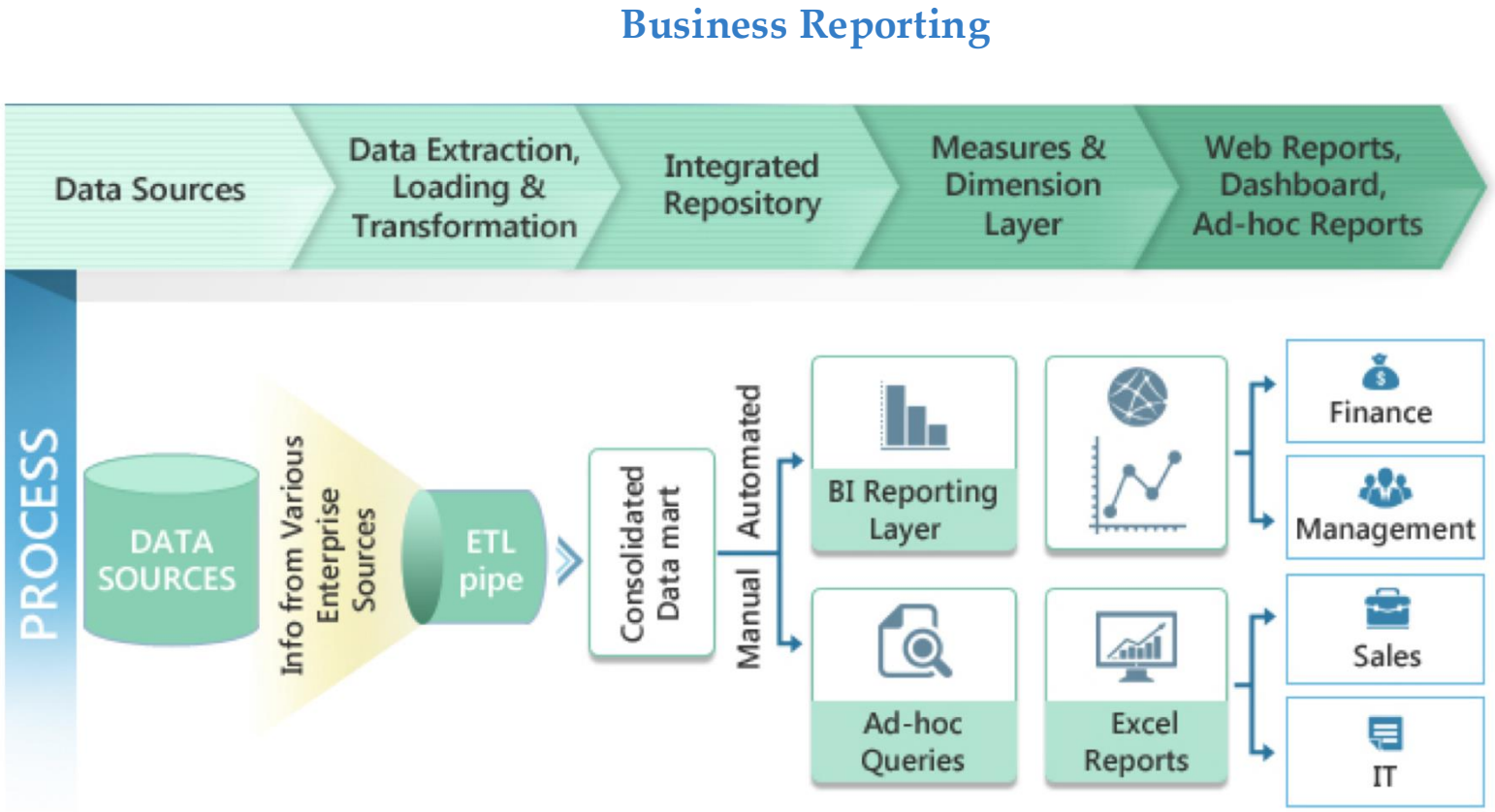
SUSS
SINGAPORE UNIVERSITY
OF SOCIAL SCIENCES

Learning Outcomes

At the end of this unit, you are expected to be able to:

- Recap using Python programming language to retrieve, create and update data via a variety of data sources and file types
- Reflect on the various techniques of processing and parsing information, e.g. using regular expression libraries to recognise and process text patterns
- Understand how information is presented, delivered and consumed on the internet via http Hypertext Transport Protocol (HTTP)
- Understand the concept of web services as a means to provide and exchange information between consumers and providers of information
- Appreciate the architectural philosophies behind web services
- Know the popular data formats (XML, JSON) and their relevant usage patterns in information retrieval and exchanges

ETL Process


Business Reporting
Data Sources
Data Extraction, Loading & Transformation
Integrated Repository
Measures & Dimension Layer
Web Reports, Dashboard, Ad-hoc Reports
PROCESS
DATA SOURCES
Info from Various Enterprise Sources
ETL pipe
Consolidated Data mart
Automated
Manual
BI Reporting Layer
Ad-hoc Queries
Excel Reports
Finance
Management
Sales
IT

ETL Process

Extract

- Extract raw data from various source systems e.g. databases, APIs, flat files, or other data repositories.

Transform

1) Data Cleaning: Removing errors, duplicates, and inconsistencies.
   - misspellings, incorrect values, missing data, or non-standard formats
   - e.g. inconsistent date formats "2024-08-14" vs. "14/08/2024".
2) Data Integration: Combining data from different sources into a unified format.
3) Data Transformation: Sorting, filtering, aggregating, or enriching the data.
4) Data Normalization: Standardizing data to ensure consistency.
   - organizing data into tables to reduce redundancy
   - ensure that similar data is stored in a uniform manner

Load

- Load transformed data into the target system e.g. data warehouse or database
- Facilitate easy querying and analysis.

ETL Process

Extract

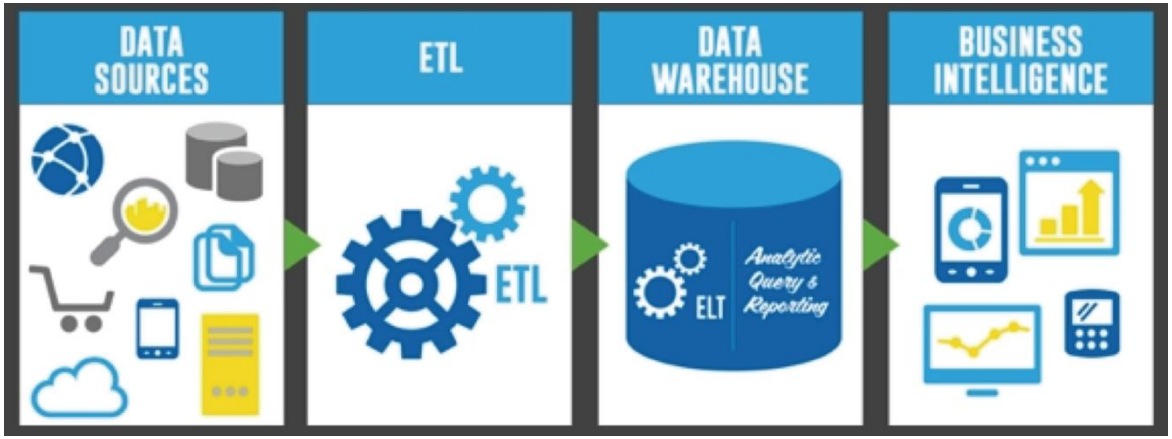
- One or more source systems containing customer, financial, or product data (CRM, Accounting system, Warehouse, MES)
- Files types - Flat files, XML, Oracle, IBM DB2, SQL Server,, IBM Websphere MQ, ODBC, JDBC, Hadoop Distributed File System (HDFS), Hive/HCatalog, JSON, Mainframe (IBM z/OS), Salesforce.com, SAP/R3

Transform

- Applying business rules, cleansing, and validating the data.
- Aggregation, Copy, Join, Sort, Merge, Partition, Filter, Reformat, Lookup
- Mathematical: +, -, x, /, Abs, IsValidNumber, Mod, Pow, Rand, Round, Sqrt, ToNumber, Truncate, Average, Min, Max
- Logical: And, Or, Not, IfThenElse, RegEx, Variables
- Text: Concatenate, CharacterLengthOf, LengthOf, Pad, Replace, ToLower, ToText, ToUpper, Translate, Trim, Hash
- Date: DateAdd, DateDiff, DateLastDay, DatePart, IsValidDate
- Format: ASCII, EBCDIC, Unicode

Load

- Load the results into one or more target systems such as a data warehouse, datamart, or business intelligence reporting system. Output: Flat files, XML, Oracle, IBM DB2, SQL Server, Teradata, Sybase, Vertica, Netezza, Greenplum, ODBC, JDBC, Hadoop Distributed File System (HDFS), Hive/HCatalog, Mainframe (IBM z/OS), Salesforce.com, Tableau, QlikView



SUSS
SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

Using Python for Data Processing

- **Setting Up Environment**

1) Python (https://www.python.org/)

2) Anaconda (https://www.anaconda.com/download)

3) Jupyter Notebook

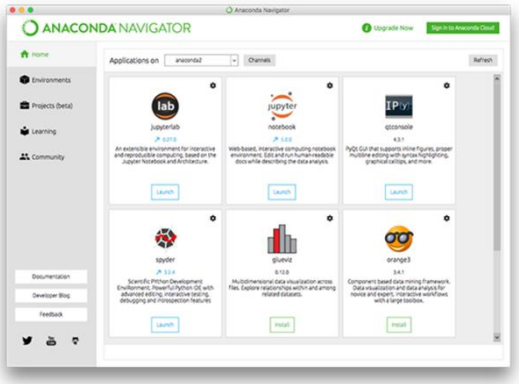


Figure 1.1 Anaconda Navigator
(Source: Created by developer)

Python Primer Exercises – Pandas for Everyone Appendix

1) Installation

2) Command Line

3) Project Template

4) Using Python

5) Working Directory

6) Environment

7) Install Package

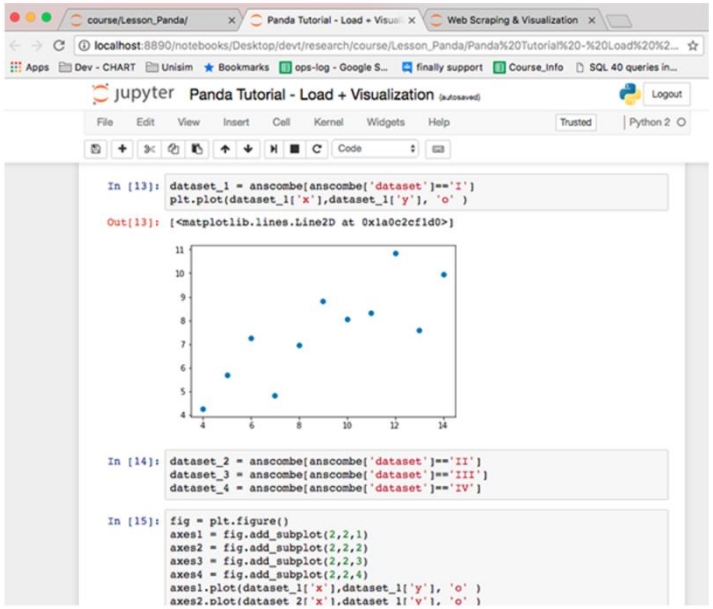


Figure 1.2 Jupyter Notebook
(Source: Created by developer)

SUSS
SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

Using Python for Data Processing

➢ Setting Up Environment

Create New Jupyter Notebook Document

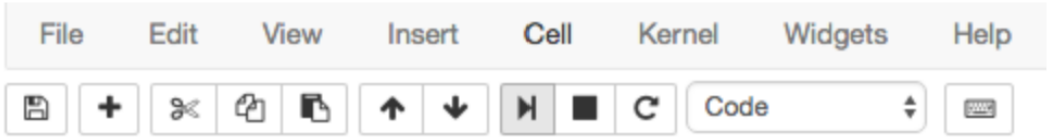


Figure 1.3 Jupyter Notebook Menu, Running Codes
(Source: Created by developer)

```
In [2]: print ('Hello World')
        Hello World
```

Figure 1.4 Jupyter Notebook Output
(Source: Created by developer)

Using Python for Data Processing

- **Using Python to Read and Write Data**

1) Getting Data

2) Dictionary and JSON format

3) CSV, TSV, Other formats

Python Primer Exercises – Pandas for Everyone Appendix

1) Import libraries

2) List

3) Tuples

4) Dictionaries

5) Slicing Values

6) Loops

7) Comprehensions

8) Functions

SUSS
SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

Using Python for Data Processing

- **Using Python to Read and Write Data**
- Getting Data
- Dictionary and JSON (JavaScript Object Notation) format

```
data = ['John', 'Tom', 'Mary', 'Jane']
class_list = data

with open('classdata.txt') as file:
    class_list = file.readlines()

for name in class_list:
     print(name)

class_list = [{'name': 'John', 'email': 'john@gmail.com', 'id': 1},
    {'name': 'Mary', 'email': 'mary@gmail.com', 'id': 2},
    {'name': 'Peter', 'email': 'peter@gmail.com', 'id': 3}
    ]
```

Using Python for Data Processing

➢ Using Python to Read and Write Data

- CSV, TSV, Other formats - Input

car_park_no	address	x_coord	y_coord	car_park_type	type_of_parking_syster	short_term_parkin	free_parking	night_parking
ACB	BLK 270/271 ALBEI	30314.7936	31490.4942	BASEMENT CAR PARK	ELECTRONIC PARKING	WHOLE DAY	NO	YES
ACM	BLK 98A ALJUNIED	33758.4143	33695.5198	MULTI-STOREY CAR PAI	ELECTRONIC PARKING	WHOLE DAY	SUN & PH FR 7AM-	YES
AH1	BLK 101 JALAN DU	29257.7203	34500.3599	SURFACE CAR PARK	ELECTRONIC PARKING	WHOLE DAY	SUN & PH FR 7AM-	YES
AK19	BLOCK 253 ANG M	28185.4359	39012.6664	SURFACE CAR PARK	COUPON PARKING	7AM-7PM	NO	NO
AK31	BLK 302/348 ANG I	29482.029	38684.1754	SURFACE CAR PARK	COUPON PARKING	NO	NO	NO
AK52	BLOCK 513 ANG M	29889.3457	39382.8134	SURFACE CAR PARK	COUPON PARKING	WHOLE DAY	NO	YES

Figure 1.5 CSV Format
(Source: Created by developer)

```
import csv

with open('data.csv') as f:
    reader = csv.reader(f)
    for row in reader:
        print(row)
```

Output:

```
['car_park_no', 'address', 'x_coord', 'y_coord',
'car_park_type', 'type_of_parking_system',
'short_term_parking', 'free_parking', 'night_parking']
['ACB', 'BLK 270/271 ALBERT CENTRE BASEMENT CAR PARK',
'30314.7936', '31490.4942', 'BASEMENT CAR PARK', 'ELECTRONIC
PARKING', 'WHOLE DAY', 'NO', 'YES']
['ACM', 'BLK 98A ALJUNIED CRESCENT', '33758.4143',
'33695.5198', 'MULTI-STOREY CAR PARK', 'ELECTRONIC PARKING',
'WHOLE DAY', 'SUN & PH FR 7AM-10.30PM', 'YES']
```

Using Python for Data Processing

➢ Using Python to Read and Write Data

- CSV, TSV, Other formats - Input

```
carpark_list = []
for row in reader:
    car_list.append(row)
```

or use the shorthand method (Python calls this List Comprehension), as follows

```
carpark_list = [row for row in reader]
```

```
import csv

with open('data.csv') as f:
    reader = csv.reader(f)
    header = next(reader) # Getting rid of the header line
    carpark_list = [row for row in reader]
```

```
reader = csv.reader(f, delimiter='\t')
```

Using Python for Data Processing

➢ Using Python to Read and Write Data

- Output CSV file as Dictionary

```
with open('data.csv') as f:
    reader = csv.DictReader(f)
    carpark_list = list(reader)
```

- Output Python Dictionary as CSV file

```
carparklist = [
    {'carpark_no': 'ACB', 'address': 'BLK 270/271 ALBERT
CENTRE BASEMENT CAR PARK', 'x_coord': '30314.7936', 'y_coord':
'31490.4942'},
    {'carpark_no': 'ACM', 'address': 'BLK 98A ALJUNIED
CRESCENT', 'x_coord': '33758.4143', 'y_coord': '33695.5198'},
    {'carpark_no': 'AH1', 'address': 'BLK 101 JALAN DUSUN',
'x_coord': '29257.7203', 'y_coord': '34500.3599'}
]
```

and you want to save the data as a CSV file, you can use the example script here:

```
# extract the names of each data column using the dict keys

cols = carparklist[0].keys()

# and write them in the first line of the CSV file

with open('newdata.csv', 'w') as f:
    f.write(','.join(cols)+'\n')

# then write the rest of the CSV file

    for o in carparklist:
        row = [str(o[col]) for col in cols]
        f.write(','.join(row)+'\n')
```

Using Python for Data Processing

- **Regular Expression**
- String Operations
- String & Number Formatting
- Regular Expressions – Special Characters
- Escape Characters

```
mystring = 'But soft what light through yonder window breaks'
substring = 'soft'
mystring.find(substring)
```

```
txt = '0123456789'
print(txt[:5])
print(txt[3:])
```

```
01234
3456789
```

```
mystr = 'Extract the number after colon :0.8475'
number = mystr[mystr.find(':')+1:]
print(float(number))
```

```
0.8475
```

```
a = 'Value of variable A'
b = 123
print ('VAR a is {} and VAR B is {}'.format(a,b))
```

```
VAR a is Value of variable A and VAR B is 123
```

```
s = 'VAR a is {a} and VAR B is {b}'
print(s.format(a = 'Value of variable A', b = 123))
```

```
VAR a is Value of variable A and VAR B is 123
```

```
print ('This is a large number {}'.format(2719249123))
print ('This is a large number {:,}'.format(2719249123))
```

```
This is a large number 2719249123
This is a large number 2,719,249,123
```

SUSS SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

Using Python for Data Processing

➢ Regular Expression

- https://docs.python.org/2/library/re.html

Download the data file from https://www.py4e.com/code3/mbox-short.txt and save the file in the same directory where you run the Python program. The mbox-short.txt file looks like this:

```
From stephen.marquard@uct.ac.za Sat Jan  5 09:14:16 2008
Return-Path: <postmaster@collab.sakaiproject.org>
Received: from murder (mail.umich.edu [141.211.14.90])
         by frankenstein.mail.umich.edu (Cyrus v2.3.8) with LMTPA;
         Sat, 05 Jan 2008 09:14:16 -0500
X-Sieve: CMU Sieve 2.3
Received: from murder ([unix socket])
         by mail.umich.edu (Cyrus v2.2.12) with LMTPA;
         Sat, 05 Jan 2008 09:14:16 -0500
Received: from holes.mr.itd.umich.edu (holes.mr.itd.umich.edu [141.211.14.79])
        by flawless.mail.umich.edu () with ESMTP id m05EEFR1013674;
        Sat, 5 Jan 2008 09:14:15 -0500
Received: FROM paploo.uhi.ac.uk (app1.prod.collab.uhi.ac.uk [194.35.219.184])
        BY holes.mr.itd.umich.edu ID 477F90B0.2DB2F.12494 ;
         5 Jan 2008 09:14:10 -0500
Received: from paploo.uhi.ac.uk (localhost [127.0.0.1])
        by paploo.uhi.ac.uk (Postfix) with ESMTP id 5F919BC2F2;
        Sat,  5 Jan 2008 14:10:05 +0000 (GMT)
Message-ID: <200801051412.m05ECIaH010327@nakamura.uits.iupui.edu>
Mime-Version:
```

Using Python for Data Processing

➢ Regular Expression

- http://www.rexegg.com/regex-quickstart.html
- Regular Expressions – Special Characters
- Escape Characters

The Regex Cheat Sheet at http://www.rexegg.com/regex-quickstart.html provides a good reference of most regular expressions. Here is a partial listing:

```
Character       Legend  Example Sample Match
\d      one digit from 0 to 9   \d\d    25
\w      ASCII letter, digit or underscore       \w\w\w  B_1
\s      space, tab, newline, carriage   a\sb\sc a b c
                return, vertical tab
\S              one character that is   \S\S\S\S        a3d_
not whitespace
.       any character   x.y     xry or x1y
\       Escapes a special character     \.\{\}  .{}
+       One or more     \w--\w+ A--b1_1
{3}     Exactly three times     \w{3}   Aa_
*       Zero or more times      A*B*C*  AAACCC
?       Once or none    plurals?        plural
```

Using Python for Data Processing

➢ Regular Expression

- http://www.rexegg.com/regex-quickstart.html
- Regular Expressions – Special Characters
- Escape Characters

The method findall() returns a list. "\S+@\S+" here means one or more non-white space character, followed by "@", followed by one or more non-white space character.

```
import re
hand = open('mbox-short.txt')
for line in hand:
    line = line.rstrip()
    x = re.findall('\S+@\S+', line)
    if len(x) > 0:
        print(x)
```

We need to modify the search pattern to state that the first and the last character of the email string must be either an upper or lower case letter, or a number. In regular expression, the format "[a-zA-Z0-9]" fulfills the purpose. The revised regular expression is:`[a-zA-Z0-9]\S*@\S*[a-zA-Z0-9]`

The code with the revised regular expression should produce output as

```
['stephen.marquard@uct.ac.za']
['postmaster@collab.sakaiproject.org']
['200801051412.m05ECIaH010327@nakamura.uits.iupui.edu']
```

If we want to extract just the domain name of the email addresses, we can simply add parentheses to the regular expression, like this: '[a-zA-Z0-9]\S*@(\S*[a-zA-Z0-9])'

```
import re
hand = open('mbox-short.txt')
for line in hand:
    line = line.rstrip()
    x = re.findall('[a-zA-Z0-9]\S*@(\S*[a-zA-Z0-9])', line)
    if len(x) > 0:
        print(x)
```

SUSS SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

Using Python for Data Processing

➢ Regular Expression

- Regular Expressions Special Characters

S/N	Character	Legend	Example	Sample Match
1	\d	one digit from 0 to 9	\d\d	25
2	\w	ASCII letter, digit or underscore	\w\w\w	B_1
3	\s	space, tab, newline, carriage	a\sb\sc	a b c
4	\S	one character that is not whitespace	\S\S\S\S	a3d_
5	.	any character	x.y	xry or x1y
6	\	Escapes a special character	\.\{\}	.{}
7	+	One or more	\w--\w+	A--b1_1
8	{3}	Exactly three times	\w{3}	Aa_
9	*	Zero or more times	A*B*C*	AAACCC
10	?	Once or none	plurals?	plural

Data Processing via Internet

➢ Reading Web Page Using HTTP

- HyperText Transport Protocol – HTTP

HTTP is basically a set of rules to facilitate the transfer and exchange of files on Web pages on the internet.

- Getting web data with Sockets

```
import socket

mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect(('data.pr4e.org', 80))
cmd = 'GET http://data.pr4e.org/romeo.txt HTTP/1.0\r\n\r\n'.encode()
mysock.send(cmd)

while True:
    data = mysock.recv(512)
    if len(data) < 1:
        break
    print(data.decode())

mysock.close()
```

The program should produce the following:

```
HTTP/1.1 200 OK
Date: Mon, 11 Jun 2018 03:41:45 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Sat, 13 May 2017 11:22:22 GMT
ETag: "a7-54f6609245537"
Accept-Ranges: bytes
Content-Length: 167
Connection: close
Content-Type: text/plain

But soft what light through yonder window breaks
It is the east and Juliet is the sun
Arise fair sun and kill the envious moon
Who is already s
ick and pale with grief
```

Data Processing via Internet

➢ Reading Web Page Using HTTP

- HyperText Transport Protocol – HTTP

HTTP is basically a set of rules to facilitate the transfer and exchange of files on Web pages on the internet.

- Using urllib
  - Text
  - Image

```
import urllib
file = urllib.urlopen('http://data.pr4e.org/romeo.txt')
for line in file:
    print(line.decode().strip())
```

```
f = open('img.jpg','wb')
f.write(urllib.urlopen('http://pbs.twimg.com/profile_images/92744634787929293O/Fi0D7FGJ_400x400.jpg').read())
f.close()
```

- Parsing Web Pages Using Regular Expressions

The program gives us a list of all the strings that match the regular expression, returning the link text between the double quotes.

```
import urllib
import re
url = 'http://www.bbc.com'
html = urllib.urlopen(url).read()
links = re.findall(b'href="(http://.+?)"', html)
for link in links:
    print(link.decode())
```

Data Processing via Internet

➢ Using Web Services

- Parsing XML

The figure below shows the various components of a typical XML document.

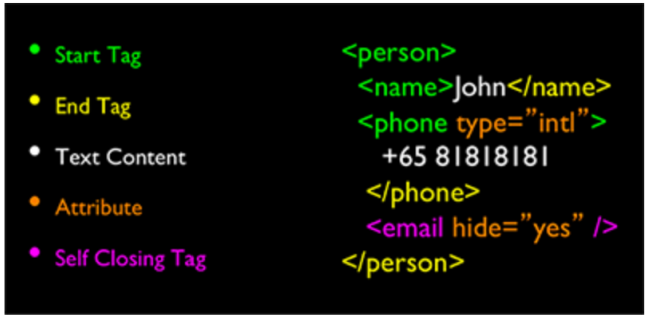


Figure 1.6 XML Components
(Source: http://www.dr-chuck.com/)

You can visualise an XML document as a tree structure, where there is top parent tag and other tags as children of their parent nodes.

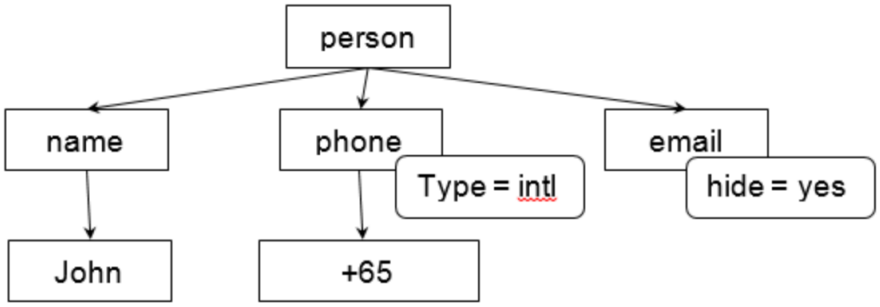


Figure 1.7 A Tree Representative of XML
(Source: Created by developer)

Data Processing via Internet

➢ Using Web Services

- Parsing JSON

```
class_list ='''
[
{"id": 1,
"name": "John",
"email": "john@gmail.com"},
{"id": 2,
"name": "Mary",
"email": "mary@gmail.com"},
{"id": 3,
"name": "Peter",
"email": "peter@gmail.com"}
]
'''
import json

info = json.loads(class_list)
print(len(info))
for item in info:
    print(item['id'], item['name'], item['email'])
```

Data Processing via Internet

➢ Using Web Services

- Parsing Data from Web Services

```
import urllib2, json

url = '''
https://data.gov.sg/api/action/datastore_search?resource_id=9326ca53-9153-4a9c-b93f-8ae032637b70
'''

req = urllib2.Request(url, headers={ 'User-Agent': 'Mozilla/5.0' })
html = urllib2.urlopen(req).read()

data = json.loads(html)
print(len(data['result']['records']))# Number of records
print(data['result']['records'][0].keys())
```

If everything goes well, you should get

```
100
[u'gross_mthly_25_percentile', u'degree', u'university',
u'gross_monthly_median', u'school', u'employment_rate_permanent',
u'basic_monthly_median', u'gross_mthly_75_percentile',
u'gross_monthly_mean', u'basic_monthly_mean', u'year', u'_id',
u'employment_rate_overall']
```

SUSS SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

Data Processing via Internet

- **Application Programming Interfaces**
  - Service-Oriented Architecture (SOA)

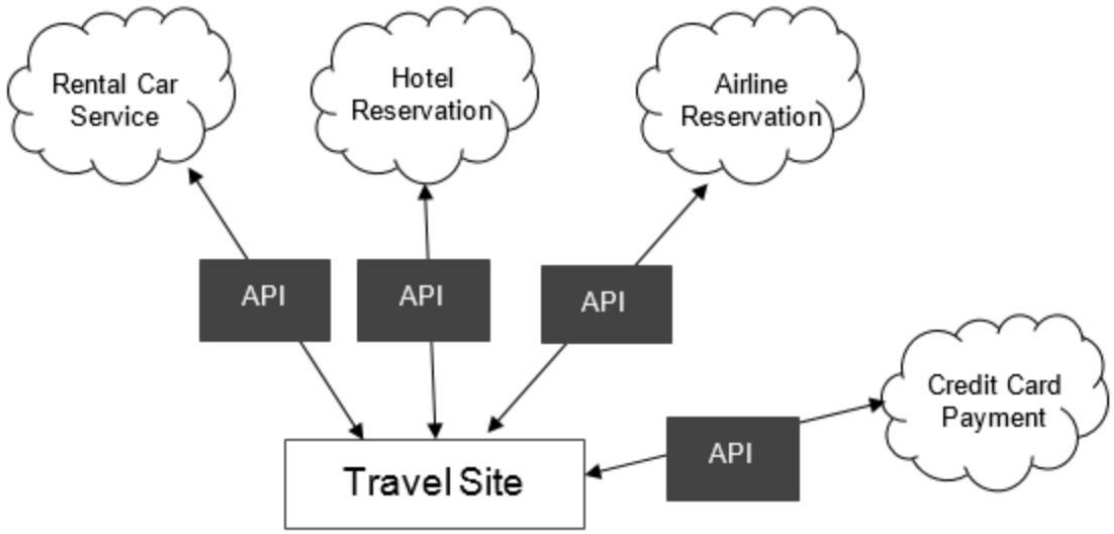


Figure 1.8 Service Oriented Architecture
(Source: Created by developer)

Data Processing via Internet

➢ Application Programming Interfaces

- Representational State Transfer Architecture (REST) - REST is an architectural style that defines a set of constraints and properties based on HTTP protocol. Web services that follow the REST architecture style are known to be RESTful.

When a request made to the web resource's URI, the user gets a response in certain data format. At the same time, it allows the user to remotely create, read, update and delete (CRUD - the four basic functions of persistent storage) them. When HTTP is used, the operations available are the common GET, POST, PUT, DELETE methods.

HTTP request	Response
GET /students	Return list of all students
GET /student/<id>	Return detail of student of <id>
POST /student	Add a new student resource
PUT /student/<id>	Update/Change details of student of <id>
DELETE /student/<id>	Delete record of student of <id>

```
import requests
URI = 'https://jsonplaceholder.typicode.com/posts/'
r = requests.get(URI)
print(r.text)
```

The output is a JSON list of all the posts in the database.


```
[
  {
    "userId": 1,
    "id": 1,
    "title": "sunt aut facere repellat provident occaecati
excepturi optio reprehenderit",
    "body": "quia et suscipit\nsuscipit recusandae
consequuntur expedita et cum\nreprehenderit molestiae ut ut
   s totam\nnostrum rerum est autem sunt rem eveniet
   hitecto"
   ,

    "userId": 1,
    "id": 2,
    "title": "qui est esse",
    "body": "est rerum tempore vitae\nsequi sint nihil
   rehenderit dolor beatae ea dolores neque\nfugiat blanditiis
   uptate porro vel nihil molestiae ut reiciendis\nqui aperiam
non debitis possimus qui neque nisi nulla"
  },
…
]
```


SUSS SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

Learning Outcomes

At the end of this unit, you are expected to be able to:

- Recap using Python programming language to retrieve, create and update data via a variety of data sources and file types
- Reflect on the various techniques of processing and parsing information, e.g. using regular expression libraries to recognise and process text patterns
- Understand how information is presented, delivered and consumed on the internet via http Hypertext Transport Protocol (HTTP)
- Understand the concept of web services as a means to provide and exchange information between consumers and providers of information
- Appreciate the architectural philosophies behind web services
- Know the popular data formats (XML, JSON) and their relevant usage patterns in information retrieval and exchanges

Thank you!

Q&A