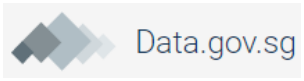
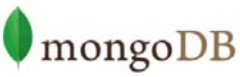
Master of Technology in Knowledge Engineering

KE5106
Semester II 2018

Lecture 8a: Webscraping 2

Charles Pang
Institute of Systems Science
National University of Singapore
Email: charlespang@nus.edu.sg



© 2018 NUS. The contents contained in this document may not be reproduced in any form or by any means, without the written permission of ISS, NUS, other than for the purpose for which it has been supplied.

Contents

- What is an API?
- What are API good for?
- API Overview
- API Examples
- Working with API in Python
- API Workshop

What is an API?

- Application Program Interfaces (APIs) offers a simple way for programmers to integrate webservices into their application software. For example:
  - Facebook, Google Maps, Dropbox, etc.
- API offers an interface to resources maintained by others- allows users to consume data resources.
- Access to an API is quite simple:
  - Make a request to a remote webserver
  - Remote webserver returns your requested data resources
- Good introduction to API can be found here:
  - https://www.programmableweb.com/
  - Links to 20K APIs around the world

© 2018 NUS. All Rights Reserved

Types of API

- Simple Object Access Protocol (SOAP)
  - Complex (envelop, body, header, message) ☹
  - Webservices Standard (Security, Contract Management) ☺
  - Verbose syntax ☹
  - XML ☹
- Representative State Transfer (REST)
  - Simple (message exchange) ☺
  - Webservices Standard (?)
  - Concise syntax ☺
  - XML, JSON ☺

© 2018 NUS. All Rights Reserved

What are APIs good for?

- Data is changing quickly (streaming)
  - An example of this is monitoring traffic conditions, displaying real-time weather data, etc.
- No need to write scrapers
  - Make a simple "**GET**" request
- No worry about website changes
  - When the site owner changes the layout or implements a new feature, your scraping will not break.
- Plug & Play functionalities
  - Make a simple "**POST**" request (e.g. Sentiment Analysis)
  - Use Facebook authentication API for website login
  - Use Google Search API for website search

© 2018 NUS. All Rights Reserved

API Overview

The HTTP protocol centers around a concept called the Request-Response Cycle. The client sends the server a request to do something. The server, in turn, sends the client a response saying whether or not the server could do what the client asked.

© 2018 NUS. All Rights Reserved

Making an API Request

To make a valid request, the client needs to include four things:

- **URL** (Uniform Resource Locator)
  - Endpoint (the location of the resource itself)
- **Method** (specify actions for the server)
- **Headers** (Operating parameters of the transaction)
- **Body** (data that is transferred)

These four items become a request string that is sent over HTTP from the client to the resource server.

```
"http://api.nytimes.com/svc/search/v2/articlesearch.
json?q=world+cup&api-key=c1b5b7678b07e7b69e37b6"
```

© 2018 NUS. All Rights Reserved

API Requests Methods

GET - Asks the server to retrieve a resource

```
requests.get('https://api.github.com/events')
```

POST - Asks the server to create a new resource

```
requests.post('http://httpbin.org/post',data={'key':'value'})
```

PUT - Asks the server to edit/update an existing resource

```
requests.put('http://httpbin.org/post',data={'key':'value'})
```

DELETE - Asks the server to delete a resource

```
requests.delete('http://httpbin.org/post',data={'key':'value'})
```

© 2018 NUS. All Rights Reserved

API Response


```
{
  "status":"OK",
  "copyright":"Copyright (c) 2018 The New York Times Company. All Rights Reserved.",
  "response":{
    "docs":[ ],
    "meta":{
      "hits":40237,
      "offset":0,
      "time":56
    }
  }
}
```

```
"web_url":"https://www.nytimes.com/2018/01/30/opinion/sotu-viewers-guide-trump.html",
"snippet":"I know what to look for, because I play the president on TV.",
"blog":{ },
"source":"The New York Times",
"multimedia":[ ],
"headline":{ },
"keywords":[ ],
"pub_date":"2018-01-30T20:29:18+0000",
"document_type":"article",
"news_desk":"OpEd",
"byline":{ },
"type_of_material":"Op-Ed",
"_id":"5a71f0bad4211f00015be55b",
"word_count":955,
"score":1.5090607,
"uri":"nyt://article/2469a51c-8fd6-5f6b-b125-ceb6f481643d"
```


© 2018 NUS. All Rights Reserved

API Example (public access)

The prefix for web resource endpoints is `https://data.gov.sg/api/3/action` and they return responses in **JSON** format. You can retrieve dataset and resource metadata through these functions:

Resource Functions	Description
`/package_metadata_show?id={package id}`	Gets the metadata of a dataset and all of its data resources. All of the data resources' download link, last updated date, etc. can be retrieved here.
`/resource_metadata_show?id={resource id}`	Gets the metadata of a specific data resource. This function is a subset of `package_metadata_show` as it would have listed all of its data resources. The data resource download link, last updated date, etc. can be retrieved here.

```
requests.get('https://data.gov.sg/api/action/package_metadata_show?id=park-connector-loop')
```

© 2018 NUS. All Rights Reserved

API Example (authentication)

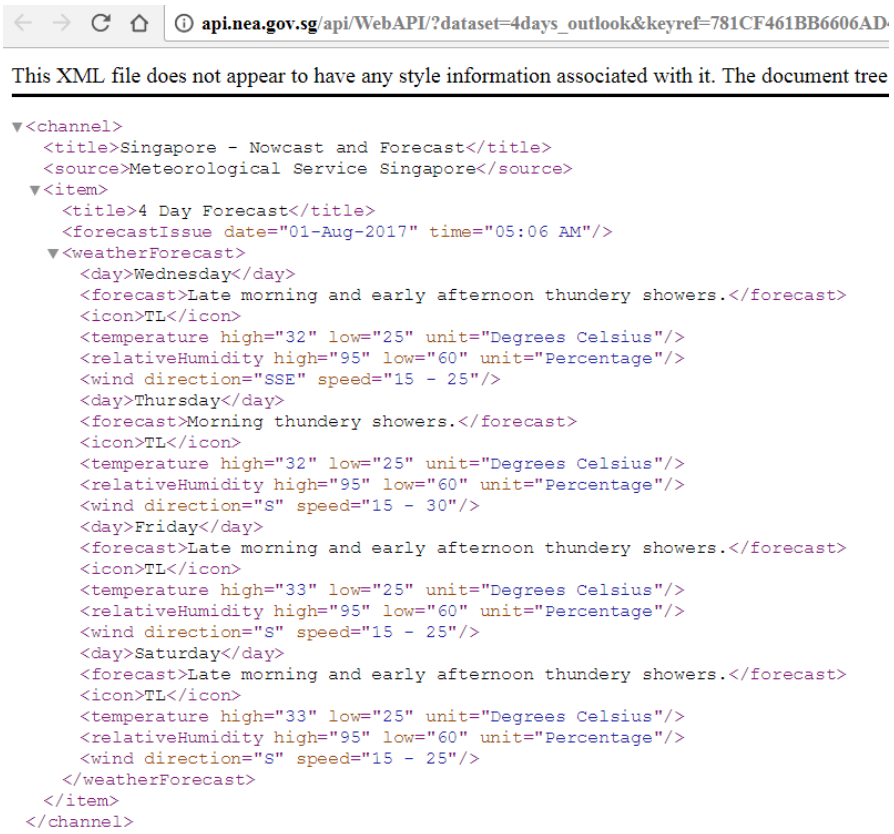
```
#Authentication parameters
headers = { 'AccountKey' : '6HsAmP1e0R/EkEYWOcjKg==',
            'accept' : 'application/json'} #this is by default

#API parameters
uri = 'http://datamall2.mytransport.sg/' #Resource URL
path = '/ltaodataservice/BusRoutes?'

#Build query string & specify type of API call
target = urlparse(uri + path)
print target.geturl()
method = 'GET'
body = ''

#Get handle to http
h = http.Http()

#Obtain results
response, content = h.request(
    target.geturl(),
    method,
    body,
    headers)
```

Uses the **urllib** instead of **requests**

LTA_DataMall_API_User_Guide v.4.7 (17 Jul 2018)

© 2018 NUS. All Rights Reserved

API Example (direct browser)



© 2018 NUS. All Rights Reserved

API Example (Tool)

LTA DataMall | API User Guide & Documentation
Version 3.7 (05 Apr 2017)

Land Transport Authority

1. MAKING API CALLS

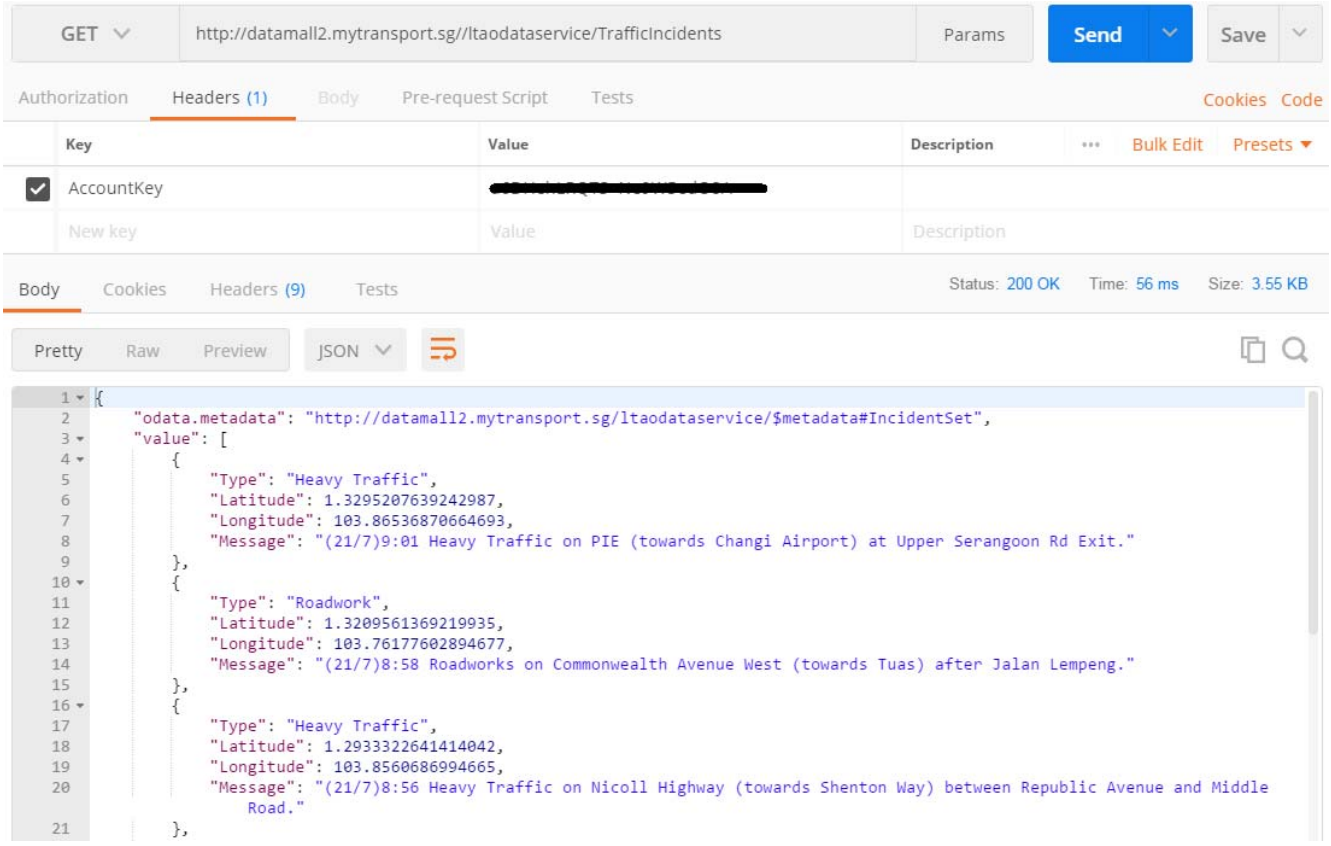
API calls need to be made programmatically in regular intervals to obtain the constant stream of data for your respective development or research needs. For illustration purposes, the API call below is being made via a third-party application – Postman.

Steps to making an API call:

1. Download and install the Postman from https://www.getpostman.com/. Fire it up!
2. Make sure Http method is set to GET.
3. Enter the URL (refer to subsequent pages in this document) in the field **request URL**.
4. Enter your AccountKey under **Headers**.
5. **(OPTIONAL STEP)** The "accept" header allows you to specify the response format of your API call. Default is JSON. Specify "application/atom+xml" for XML.
6. Click on the **Send** button.

© 2018 NUS. All Rights Reserved

Postman



© 2018 NUS. All Rights Reserved

Working with API in Python

- Using APIs tools makes life a lot easier.
- Embedding API calls within an application is even more useful.
- For example, you can build an application that gives users an alert if rain/traffic-jam is imminent
- The programming way to use an API is:
  1. Determine the access authentication
  2. Determine the resource URL
  3. Construct a request
  4. Determine if the API request was successful
  5. Process the API responses

© 2018 NUS. All Rights Reserved

1. Determine the access authentication

- Earlier we discussed that clients (program) make data request from servers. The important question is how does the server know that the **client is who it claims to be?**
- The client has to proved its identity to the server by providing some credentials. The most basic credentials are Username and Password.
- Authentication is proving your identity to the server by giving it "secret" information that only you know. The server verifies your credentials. Once the server knows who you are, it can trust you and divulge the private data.

© 2018 NUS. All Rights Reserved

Authentication Schemes

- Basic Authentication
  - Username and Password
  - Around since 1996 (Netscape)
  - Standards: https://tools.ietf.org/html/rfc2617)
  - `requests.get('https://api.github.com/user', auth=('user', 'pass'))`
- API Keys
  - Uses a long series of letters and numbers that is distinct from the account owner's login password
  - `requests.get(URL,headers={'AccountKey':KEY})`
  - Data.gov.sg is currently using only API Keys

© 2018 NUS. All Rights Reserved

Authentication Schemes (cont'd)

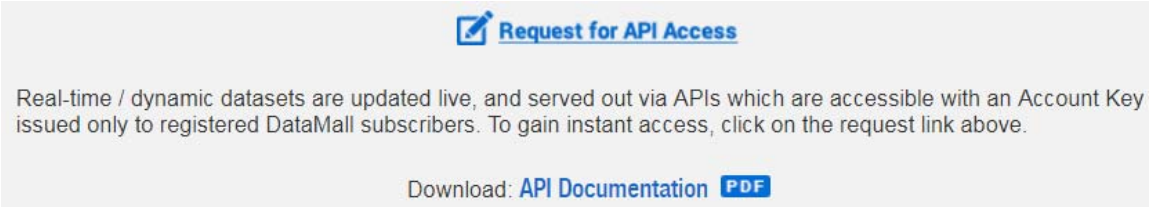
- Open Authorization (OAuth)
  - OAuth is a secure authorization protocol that deals with the authorization of third party application to access the user data without exposing their password.
  - Login with fb, gPlus, twitter in many websites - all work under this protocol.

```
1  import tweepy
2  from tweepy import OAuthHandler
3
4  consumer_key = 'YOUR-CONSUMER-KEY'
5  consumer_secret = 'YOUR-CONSUMER-SECRET'
6  access_token = 'YOUR-ACCESS-TOKEN'
7  access_secret = 'YOUR-ACCESS-SECRET'
8
9  auth = OAuthHandler(consumer_key, consumer_secret)
10 auth.set_access_token(access_token, access_secret)
11
12 api = tweepy.API(auth)
```


© 2018 NUS. All Rights Reserved

Obtaining authentication keys

- All API providers will tell you if you need authentication to access their data resources:



- Upon signing up, you will receive an Account Key:

```
6HsAmP1e0R/EkEYWOcjKg==
```

- Authentication ensures that:
  - These keys let the API operator keep track of how many requests you issue, so that you don't exceed query limits.

© 2018 NUS. All Rights Reserved

2. Determine the resource URL

- All API providers will document how you can find their data resources

URL	http://datamall2.mytransport.sg/ltaodataservice/TrafficIncidents
Description	Returns incidents currently happening on the roads, such as Accidents, Vehicle Breakdowns, Road Blocks, Traffic Diversions etc.
Update Freq	2 minutes – whenever there are updates

- Baseurl = **`http://datamall2.mytransport.sg/`**
- Endpoint= **`/latodataservice/TrafficIncidents`**

© 2018 NUS. All Rights Reserved

3. Constructing your request

- By now, you would have the URL of the API and the authentication credentials.
- You can now construct your request for data using the python request library
- The various ways to use Requests is given here: http://docs.python-requests.org/en/master/user/quickstart/
- If no authentication is required, it is very easy:

```
requests.get('https://data.gov.sg/api/action/package_metadata_show?id=park-connector-loop')
```

- Usually there is a limit (e.g. 50) per request. If you want (say) 5,000 records, a program with a loop is much more efficient than cut-and-paste

© 2018 NUS. All Rights Reserved

requests.get with authentication

- If authentication is needed, find out where and how to put the authentication details.
- Set up your credentials according to the API's documentation and instructions.

```
credentials = {'AccountKey':'a6DHehLRDGKL+Me9U5edO8A==',
           'accept':'application/json'}

baseurl = 'http://datamall2.mytransport.sg/ltaodataservice/'

endpoint = 'TrafficIncidents?'

radius = {"Latitude": 1.304980,
          "Longitude": 103.831984,
          "Distance":'5000'}

requests.get(baseurl+endpoint, headers=credentials, params=radius)
```


© 2018 NUS. All Rights Reserved

Specifying a response type

- You may specify a response type in the request header using Multipurpose Internet Mail Extension (MIME):
  - `type/subtype`
- **Accept** header attribute specifies the format of response data which the client expects
  - `Accept: application/json`
  - `Accept: text/xml`
- **Content-Type** header attribute specifies the format of data being sent to the receiver.
  - `Content-Type: application/json`
  - `Content-Type: text/xml`

© 2018 NUS. All Rights Reserved

4.Determine if the API request was successful

Status codes are returned with every request made:

200: everything went okay, and the result has been returned

301: the server is redirecting you to a different endpoint. Eg. coy changes domain names or endpoint name is changed

401: you're not authenticated. This happens when you don't send the right credentials to access an API

400: you made a bad request. This happens when you don't send along the right data, among other things.

403: the resource you're trying to access is forbidden – you don't have the right permissions to see it.

404: the resource you tried to access wasn't found on the server.

```
page = requests.get(url)

if page.status_code != 200:
  return False
```

© 2018 NUS. All Rights Reserved

5.Handle the API responses

API response are commonly either in XML or JSON.

JSON (JavaScript Object Notation) is a lightweight **data-interchange format**. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of JavaScript. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including Python. These properties make JSON an ideal data-interchange language.

JSON is built on two structures:

- A collection of name/value pairs. (i.e. python dictionary)
- An ordered list of values.

XML (eXtensible Markup Language) is a markup language much like HTML. XML was designed **to store and transport data.** XML tags are not predefined and designed to be self-descriptive.

The main difference between HTML and XML are:

- XML does not use predefined tags
- XML is extensible

XML provides two advantages as a data representation language:

- It is text-based.
- It is position-independent.

© 2018 NUS. All Rights Reserved

Json vs XML

Both JSON and XML can be used to receive data from a web server.

The following JSON and XML examples both define an employees object, with an array of 3 employees:

```
{"employees":[
    { "firstName":"John", "lastName":"Doe" },
    { "firstName":"Anna", "lastName":"Smith" },
    { "firstName":"Peter", "lastName":"Jones" }
]}
```

```
<employees>
    <employee>
        <firstName>John</firstName> <lastName>Doe</lastName>
    </employee>
    <employee>
        <firstName>Anna</firstName> <lastName>Smith</lastName>
    </employee>
    <employee>
        <firstName>Peter</firstName> <lastName>Jones</lastName>
    </employee>
</employees>
```


© 2018 NUS. All Rights Reserved

XML format

```
<title>Singapore - Nowcast and Forecast</title>
<source>Meteorological Service Singapore</source>
<item>
    <title>4 Day Forecast</title>
    <forecastIssue date="01-Aug-2017" time="05:06 AM" />
    <weatherForecast>
        <day>Wednesday</day>
        <forecast>Late morning and early afternoon thundery showers.</forecast>
        <icon>TL</icon>
        <temperature high="32" low="25" unit="Degrees Celsius" />
        <relativeHumidity high="95" low="60" unit="Percentage" />
        <wind direction="SSE" speed="15 - 25" />
        <day>Thursday</day>
        <forecast>Morning thundery showers.</forecast>
        <icon>TL</icon>
        <temperature high="32" low="25" unit="Degrees Celsius" />
        <relativeHumidity high="95" low="60" unit="Percentage" />
        <wind direction="S" speed="15 - 30" />
        <day>Friday</day>
        <forecast>Late morning and early afternoon thundery showers.</forecast>
        <icon>TL</icon>
        <temperature high="33" low="25" unit="Degrees Celsius" />
        <relativeHumidity high="95" low="60" unit="Percentage" />
        <wind direction="S" speed="15 - 25" />
        <day>Saturday</day>
        <forecast>Late morning and early afternoon thundery showers.</forecast>
        <icon>TL</icon>
        <temperature high="33" low="25" unit="Degrees Celsius" />
        <relativeHumidity high="95" low="60" unit="Percentage" />
        <wind direction="S" speed="15 - 25" />
    </weatherForecast>
</item>
```


© 2018 NUS. All Rights Reserved

Parsing XML in Python

- Python has 3 standard library (APIs) for working with XML documents
- **Document Object Model (DOM) API** – W3C complaint API. Reads an entire file into memory and stored in a hierarchical (tree-based) form to represent all the features of an XML document.
- **Simple API for XML (SAX)** – Event-driven API. Stream-parses the file for events of interest and uses callback event handlers. Useful for large files and memory limitations since the entire file is never stored in the memory.
- **ElementTree (ET)** – Lightweight API. Represents tree as structure of lists, and attributes as dictionaries. Comes in a 'C' implementation as well. Faster than DOM and has stream processing (ET.iterparse) like SAX. Many prefer this since it is more convenient to use.
- **BeautifulSoup**

© 2018 NUS. All Rights Reserved

JSON format


```
"channel": {
  "title": "Singapore - Nowcast and Forecast",
  "source": "Meteorological Service Singapore",
  "item": {
    "title": "4 Day Forecast",
    "forecastIssue": {
      "-date": "01-Aug-2017",
      "-time": "05:06 AM"
      }
  },
  "weatherForecast":{
      "Wednesday": {
      "forecast": "Late morning and early afternoon thundery showers",
      "temperature":{"high":32, "low":25, "unit":"Degrees Celsius"},
      "relativeHumidity":{"high":95, "low":60, "unit":"Percentage"},
      "wind": {"direction":"SSE", "speed":"15 - 25"}
      },

      "Thursday":{
      "forecast":"Morning thundery showers",
      "temperature":{"high":32, "low":25, "unit":"Degrees Celsius"},
      "relativeHumidity":{"high":95, "low":60, "unit":"Percentage"},
      "wind":{"direction":"S", "speed":"15 - 30"}
      },

      "Friday":{
      "forecast":"Late morning and early afternoon thundery showers",
      "temperature":{"high":33, "low":25, "unit":"Degrees Celsius"},
      "relativeHumidity":{"high":95, "low":60, "unit":"Percentage"},
      "wind": {"direction":"S", "speed":"15 - 25"}
      },
```


© 2018 NUS. All Rights Reserved

Parsing JSON in Python

- Parsing JSON in Python is much easier than XML since it is very similar to the python dictionary data structure.
- BeautifulSoup is used to parse the raw data returned from an API call.
- All the BeautifulSoup search functionalities (e.g. findAll) are used to extract the data that we want.
- **`import json`** : is library for parsing and manipulating JSON data in Python
- **`Json.loads`**: converts JSON into Python dictionary
- **`Json.dumps`**: converts Python dictionary into JSON

© 2018 NUS. All Rights Reserved

Workshop

Last

© 2018 NUS. All Rights Reserved