Learn REST API with Python in 90 minutes

Larry cai <larry.caiyu@gmail.com>

Agenda

- What is REST API ?
- Exercise 1: use curl to play with REST API
- Exercise 2: use requests module to GET headers
- Exercise 3: write in python script
- Exercise 4: HTTPS & Basic Authentication towards Github
- Exercise 5: POST your script into gist
- Summary & Homework & Reference

REST API

- REST = REpresentation State Transfer
- REST vs. SOAP
- XML vs. JSON

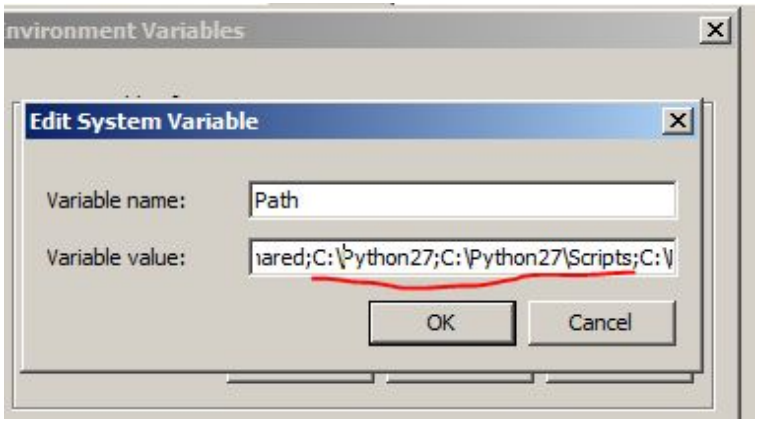
Resource	POST create	GET read	PUT update	DELETE delete
/dogs	create a new dog	list dogs	bulk update dogs	delete all dogs
/dogs/1234	error	show Bo	if exists update Bo if not error	delete Bo

- REST/JSON is first choice to use ! (twitter, jenkins...)

See slides http://www.slideshare.net/rmaclean/json-and-rest for detail
Or
"Beautiful REST + JSON APIs" by Les Hazlewood
http://www.youtube.com/watch?v=hdSrT4yjS1g

Environment

- Python 2.7.x In Windows with Git Bash
  - http://www.python.org/ftp/python/2.7.3/python-2.7.3.msi
  - http://git-scm.com/downloads
  - Add into Path



Exercise 1: using curl

- Try to get the data in browser and command using cURL http://httpbin.org/get

```
$ export http_proxy=
$ export https_proxy=
  $ curl -v http://httpbin.org/get
```

Exercise: use urllib2 module

- SKIP this, it wastes time

Requests Module

- Python's standard **urllib2** module provides most of the HTTP capabilities you need, but the API is thoroughly **broken**.
- Python HTTP: When in doubt, or when not in doubt, use Requests. Beautiful, simple, Pythonic.

http://www.python-requests.org

Exercise 2: use requests module

- Install requests module (could use pip)
  http://docs.python-requests.org/en/latest/user/install/#install

- Use it interactive mode

```
$ python
>>> import requests
>>> r = requests.get("http://httpbin.org/get")
```

```
>>> r.status_code
200
>>> r.headers['content-type']
'application/json; charset=utf8'
>>> r.encoding
'utf-8'
>>> r.text
u'{"type":"User"...'
>>> r.json()
```

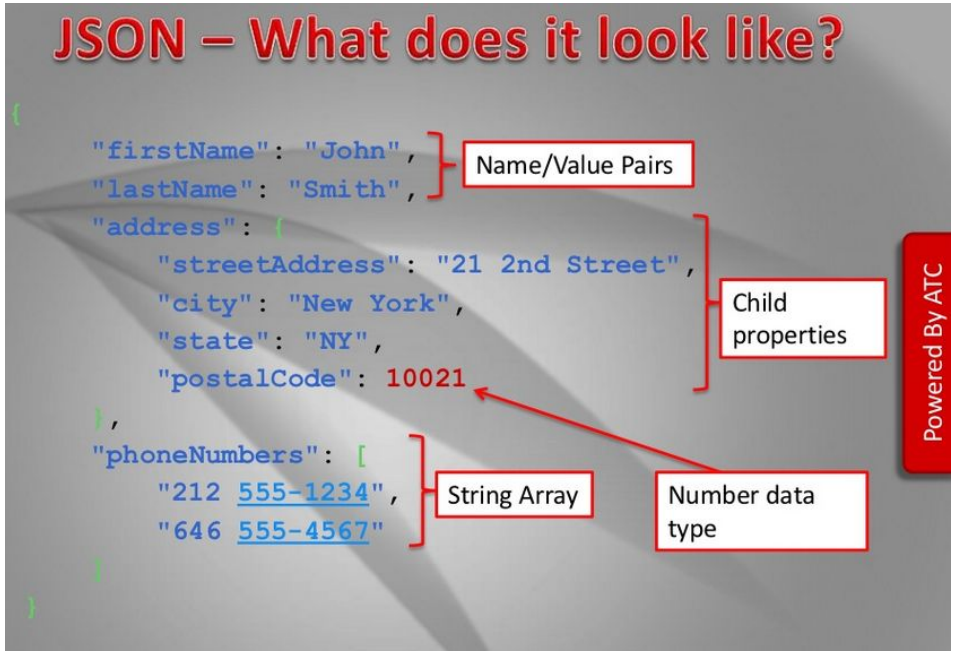
JSON format and usage in python

- **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.

- `>>> print r.json()`

- Python JSON module

```
>>>import json
>>>data["firstName"]
"John"
```



http://www.slideshare.net/rmaclean/json-and-rest

Exercise 3: write in python script

- Download exercise code `restapi.py` from github
  NOTE: this is prepared script
  - https://gist.github.com/larrycai/7823499
- Implement tasks in method exer3()
  - print result text, status_code for http://httpbin.org/get
    ```
    print r.status_code
    ```
  - Print "origin" and "User-Agent" (JSON) (use r.json())
  - Enable debug by calling enable_debug()
- Execute command to verify
  ```
  $ ./restapi.py -e 3
  ```

Exercise 4: HTTPS & Authentication

- HTTPS (SSL certification)

```
$ curl https://api.github.com/users/larrycai
=> requests.get(…,verify=False)
```

- Authentication (Basic, Digest, OAuth), use curl command

```
$ curl -v -u larrycai https://api.github.com/user
> Authorization: Basic bGFycnljYWk6TTFkZUBMMWZl

=> HTTP Basic Auth => requests.get(…, auth = HTTPBasicAuth(user, passwd))
```

- Tasks:
  - Use curl to try your own account
  - Use script get your account's "created_at"

http://developer.github.com/v3/

Exercise 5: POST data

- Upload with POST method

```
curl -X POST -d
'{"public":true,"files":{"test.txt
":{"content":"String file
contents"}}}'
https://api.github.com/gists
```

```
>>> import json
>>> json.dumps(['foo', {'bar':
      ('baz', 1.0, 2)}])
'["foo", {"bar": ["baz",1.0, 2]}]'
>>> print json.dumps("\"foo\bar")
"\"foo\bar"
```

Create a gist

```
POST /gists
```

Input

Name	Type	Description
files	hash	**Required**. Files that make up this gist.
description	string	A description of the gist.
public	boolean	Indicates whether the gist is public. Default: false

The keys in the `files` hash are the `string` filename, and the value is another `hash` with a key of `content`, and a value of the file contents. For example:

```
{
  "description": "the description for this gist",
  "public": true,
  "files": {
    "file1.txt": {
      "content": "String file contents"
    }
  }
}
```

- Task:
  - Upload your current restapi.py to YOUR gist

http://developer.github.com/v3/gists/#create-a-gist

Summary

- Requests module is preferred to use in python
- Try browser/cURL before script to access API
- REST/JSON is good marriage

- Coding and learn from others

Exercise 6 : Homework

- Implement for all gist API
- Use oauth to access github in python script
- Get all jenkins jobs in python script
- Submit your final result into gist, then give me email, you may get gift ;-)

Reference

- Understand REST API: http://www.restapitutorial.com/
- Requests python module: http://www.python-requests.org
- Github developer API v3
  - http://developer.github.com/v3/
- Jenkins API
  - https://wiki.jenkins-ci.org/display/JENKINS/Remote+access+API