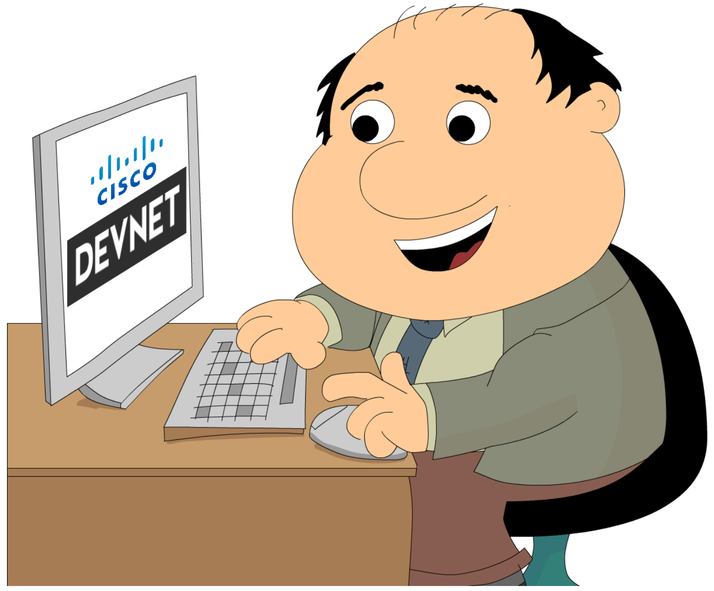
Python Part 3: Useful Python Libraries for Network Engineers

A Network Programmability Basics Presentation

Hank Preston, ccie 38336
Developer Evangelist
@hfpreston

Network Programmability Basics Modules

- Introduction: How to be a Network Engineer in a Programmable Age
- **Programming Fundamentals**
- Network Device APIs
- Network Controllers
- Application Hosting and the Network
- NetDevOps



© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Network Programmability Basics: The Lessons

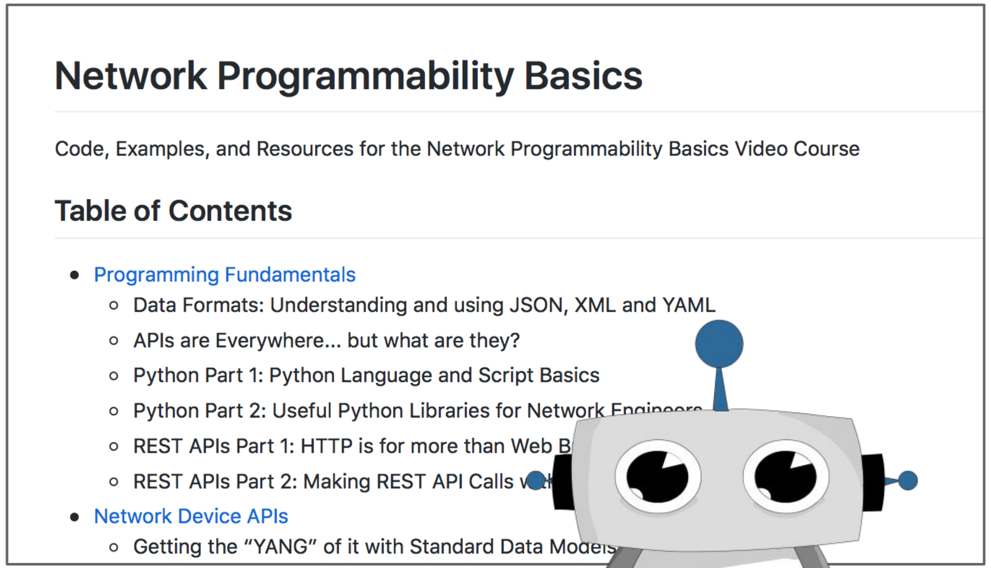
Module: Programming Fundamentals

- Data Formats: Understanding and using JSON, XML and YAML
- APIs are Everywhere... but what are they?
- REST APIs Part 1: HTTP is for more than Web Browsing
- REST APIs Part 2: Making REST API Calls with Postman
- Python Part 1: Python Language and Script Basics
- Python Part 2: Working with Libraries and Virtual Environments
- Python Part 3: Useful Python Libraries for Network Engineers

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Code and Develop Along

- Get the Code!
  - github.com/CiscoDevNet/netprog_basics
- Setup Lab Prerequisites
  - Each lab includes a README with details
- Access to Infrastructure
  - DevNet Sandbox
  - Specifics in lab README

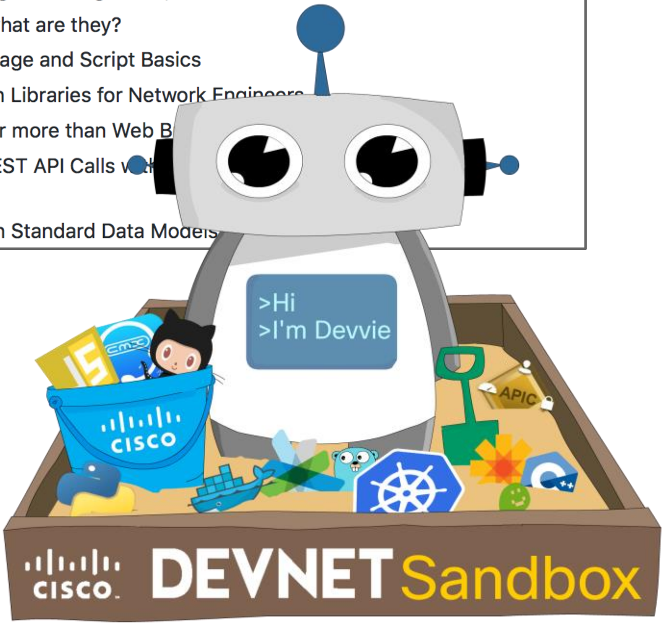


Network Programmability Basics

Code, Examples, and Resources for the Network Programmability Basics Video Course

Table of Contents

- Programming Fundamentals
  - Data Formats: Understanding and using JSON, XML and YAML
  - APIs are Everywhere... but what are they?
  - Python Part 1: Python Language and Script Basics
  - Python Part 2: Useful Python Libraries for Network Engineers
  - REST APIs Part 1: HTTP is for more than Web B
  - REST APIs Part 2: Making REST API Calls w
- Network Device APIs
  - Getting the "YANG" of it with Standard Data Models



Topics to Cover

- Libraries to Work with Data
  - XML
  - JSON
  - YAML
  - CSV
- API Libraries
  - REST APIs
  - NETCONF/YANG
  - CLI

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Libraries to Work with Data

Manipulating Data of All Formats

- XML - xmltodict
  - `pip install xmltodict`
    `import xmltodict`
- JSON
  - import json
- YAML - PyYAML
  - `pip install PyYAML`
    `import yaml`
- CSV
  - `import csv`

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Treat XML like Python Dictionaries with xmltodict

- Easily work with XML data
- Convert from XML -> Dict* and back
  - xmltodict.parse(**xml_data**)
  - xmltodict.unparse(**dict**)
- Python includes a native [Markup](#) (html/xml) interfaces as well
  - More powerful, but also more complex

** Technically to an OrderedDict*

```
>>> import xmltodict
>>> from pprint import pprint
>>>
>>> xml_example = open("xml_example.xml").read()
>>> pprint(xml_example)
('<?xml version="1.0" encoding="UTF-8" ?>\n'
 '<interface xmlns="ietf-interfaces">\n'
 '  <name>GigabitEthernet2</name>\n'
 '  <description>Wide Area Network</description>\n'
 '  <enabled>true</enabled>\n'
 '  <ipv4>\n'
 '    <address>\n'
 '      <ip>172.16.0.2</ip>\n'
 '      <netmask>255.255.255.0</netmask>\n'
 '    </address>\n'
 '  </ipv4>\n'
 '</interface>\n')
>>>
>>> xml_dict = xmltodict.parse(xml_example)
>>> int_name = xml_dict["interface"]["name"]
>>> int_name
'GigabitEthernet2'
>>>
>>> xmltodict.unparse(xml_dict)
'<?xml version="1.0" encoding="utf-8"?>\n<interface xmlns="ietf-interf
gabitEthernet2</name><description>Wide Area Network</description><enab
led><ipv4><address><ip>172.16.0.2</ip><netmask>255.255.255.0</netmask
v4></interface>'
```

https://pypi.python.org/pypi/xmltodict

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

netprog_basics/programming_fundamentals/python_part_3/xml_example.xml

To JSON and back again with json

- JSON and Python go together like peanut butter and jelly
  - `json.loads(`**`json_data`**`)`
  - `json.dumps(`**`object`**`)`
- JSON Objects convert to Dictionaries
- JSON Arrays convert to Lists

```
>>> import json
>>> from pprint import pprint
>>>
>>> json_example = open("json_example.json").read()
>>> pprint(json_example)
('{\n'
 '    "ietf-interfaces:interface": {\n'
 '        "name": "GigabitEthernet2",\n'
 '        "description": "Wide Area Network",\n'
 '        "enabled": true,\n'
 '        "ietf-ip:ipv4": {\n'
 '            "address": [\n'
 '                {\n'
 '                    "ip": "172.16.0.2",\n'
 '                    "netmask": "255.255.255.0"\n'
 '                }\n'
 '            ]\n'
 '        }\n'
 '    }\n'
 '}\n')
>>>
>>> json_python = json.loads(json_example)
>>> int_name = json_python["ietf-interfaces:interface"]["name"]
>>> int_name
'GigabitEthernet2'
>>>
>>> json.dumps(json_python)
'{"ietf-interfaces:interface": {"name": "GigabitEthernet2", "description":
a Network", "enabled": true, "ietf-ip:ipv4": {"address": [{"ip": "172.16.0
ask": "255.255.255.0"}]}}}'
```

https://docs.python.org/3/library/json.html

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

netprog_basics/programming_fundamentals/python_part_3/json_example.json

YAML? Yep, Python Can Do That Too!

- Easily convert a YAML file to a Python Object
  - `yaml.load(`**`yaml_data`**`)`
  - `yaml.dump(`**`object`**`)`
- YAML Objects become Dictionaries
- YAML Lists become Lists

```
>>> import yaml
>>> from pprint import pprint
>>>
>>> yml_example = open("yaml_example.yaml").read()
>>> pprint(yml_example)
('---\n'
 'ietf-interfaces:interface:\n'
 '  name: GigabitEthernet2\n'
 '  description: Wide Area Network\n'
 '  enabled: true\n'
 '  ietf-ip:ipv4:\n'
 '    address:\n'
 '    - ip: 172.16.0.2\n'
 '      netmask: 255.255.255.0\n')
>>>
>>> yaml_python = yaml.load(yml_example)
>>> int_name = yaml_python["ietf-interfaces:interface"]["name"]
>>> int_name
'GigabitEthernet2'
>>>
>>> yaml.dump(yaml_python)
'ietf-interfaces:interface:\n  description: Wide Area Network\n  en
tf-ip:ipv4:\n    address:\n    - {ip: 172.16.0.2, netmask: 255.255
igabitEthernet2\n'
```

https://pypi.python.org/pypi/PyYAML/3.12

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

netprog_basics/programming_fundamentals/python_part_3/yaml_example.yaml

Import Spreadsheets and Data with csv

- Treat CSV data as lists
  - `csv.reader(`**`file_object`**`)`
- Efficiently processes large files without memory issues
- Options for header rows and different formats

```
>>> import csv
>>> import pprint
>>>
>>> csv_example = open("csv_example.csv").read()
>>> csv_example
'"router1","10.1.0.1","New York"\n"router2","10.2.0.1","Denver"\n"router3",
","Austin" \n'
>>>
>>> csv_example = open("csv_example.csv")
>>> csv_python = csv.reader(csv_example)
>>> for row in csv_python:
        print("{} is in {} and has IP {}".format(row[0], row[2], row[1]))


router1 is in New York and has IP 10.1.0.1
router2 is in Denver and has IP 10.2.0.1
router3 is in Austin  and has IP 10.3.0.1
```

https://docs.python.org/3/library/csv.html

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

netprog_basics/programming_fundamentals/python_part_3/csv_example.csv

API Libraries

Access Different APIs Easily

- REST APIs – [requests]
  - `pip install requests`
    `import requests`
- NETCONF – [ncclient]
  - `pip install ncclient`
    `import ncclient`
- Network CLI – [netmiko]
  - `pip install netmiko`
    `import netmiko`

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Make HTTP Calls with Ease using requests

- Full HTTP Client
- Simplifies authentication, headers, and response tracking
- Great for REST API calls, or any HTTP request

```
>>> import requests
>>> from pprint import pprint
>>> router = {"ip": "10.10.20.21",
          "port": "443",
          "user": "root",
          "pass": "cisco123"}
>>>
```

http://docs.python-requests.org

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

netprog_basics/programming_fundamentals/python_part_3/api_requests_example.py

Demo Time!

YANG Model Data with NETCONF and ncclient

- Full NETCONF Manager (ie client) implementation in Python
  - See later presentation on NETCONF details
- Handles all details including authentication, RPC, and operations
- Deals in raw XML

```
>>> from ncclient import manager
>>> from pprint import pprint
>>> import xmltodict
>>>
>>> router = {"ip": "10.10.20.21",
          "port": 830,
          "user": "root",
          "pass": "cisco123"}
>>> netconf_filter = """
<filter>
  <interfaces xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces">
    <interface>
      <name>GigabitEthernet1</name>
    </interface>
  </interfaces>
</filter>
"""
>>> m = manager.connect(host=router["ip"],
                    port=router["port"],
                    username=router["user"],
                    password=router["pass"],
                    hostkey_verify=False)
>>>
>>> interface_netconf = m.get_config("running", netconf_filter)
>>> interface_python = xmltodict.parse(interface_netconf.xml)["rpc-reply"]["data"]
>>> pprint(interface_python["interfaces"]["interface"]["name"]["#text"])
'GigabitEthernet1'
```

https://ncclient.readthedocs.io

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

netprog_basics/programming_fundamentals/python_part_3/api_ncclient_example.py

Demo Time!

For When CLI is the Only Option – netmiko

- If no other API is available…
- Builds on paramiko library for SSH connectivity
- Support for a range of vendors network devices and operating systems
- Send and receive clear text
  - Post processing of data will be key

```
>>> from netmiko import ConnectHandler
>>> from pprint import pprint
>>>
>>> router = {"device_type": "cisco_ios",
          "host": "10.10.20.21",
          "user": "root",
          "pass": "cisco123"}
>>>
>>> net_connect = ConnectHandler(ip=router["host"],
                             username=router["user"],
                             password=router["pass"],
                             device_type=router["device_type"])
>>>
>>> interface_cli = net_connect.send_command("show run int Gig1")
>>>
>>> pprint(interface_cli)
('Building configuration...\n'
 '\n'
 'Current configuration : 136 bytes\n'
 '!\n'
 'interface GigabitEthernet1\n'
 ' ip address 10.10.20.21 255.255.255.0\n'
 ' ip nat outside\n'
 ' negotiation auto\n'
 ' no mop enabled\n'
 ' no mop sysid\n'
 'end\n')
```

https://github.com/ktbyers/netmiko

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

netprog_basics/programming_fundamentals/python_part_3/api_netmiko_example.py

Demo Time!

Summing up

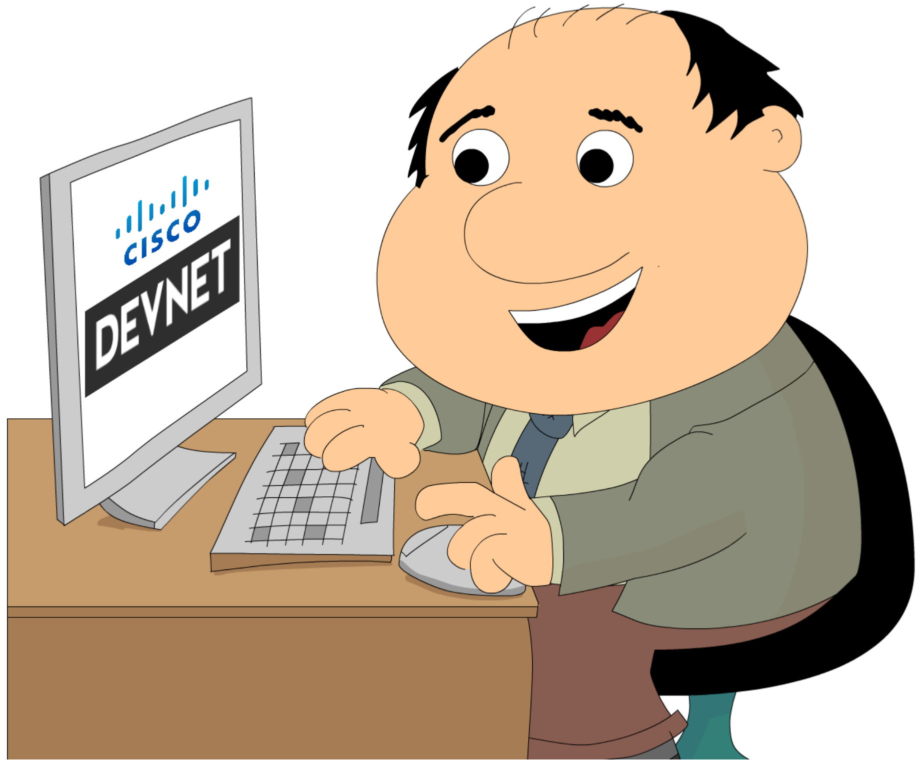
Review

- Looked at how to use Python libraries to work with XML, JSON, YAML and CSV data
- Learned about libraries for leveraging REST APIs, NETCONF, and CLI interfaces

© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Call to Action!

- Complete the full **Network Programmability Basics** Course
- Run the examples and exercises yourself!
  - Bonus Examples!
- Join DevNet for so much more!
  - Learning Labs
  - Development Sandboxes
  - Code Samples and API Guides

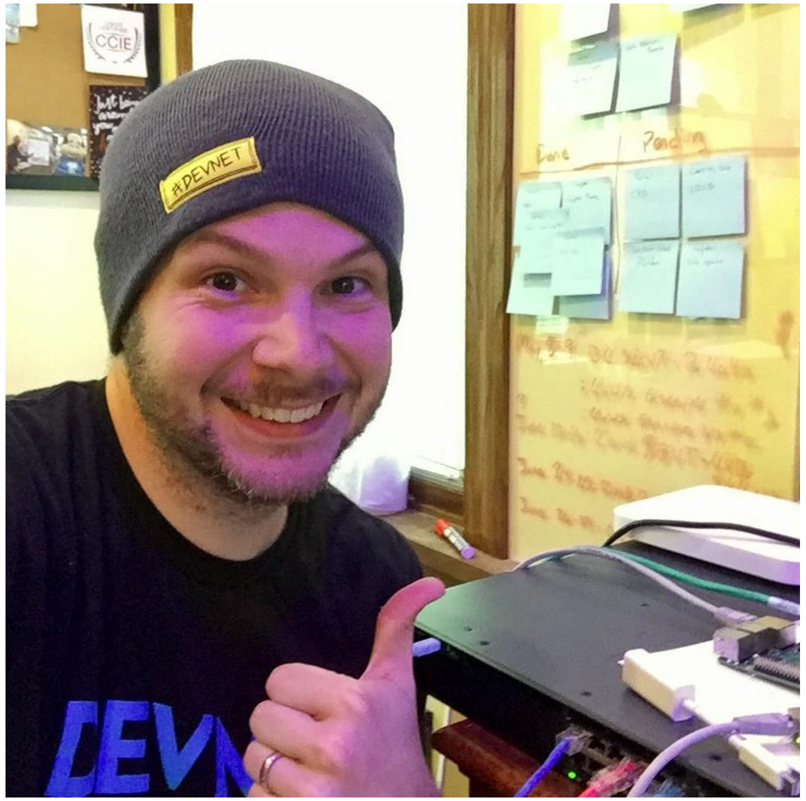


© 2017 Cisco and/or its affiliates. All rights reserved. Cisco Confidential

Got more questions? Come find me!

- hapresto@cisco.com
- @hfpreston
- http://github.com/hpreston

- @CiscoDevNet
- facebook.com/ciscodevnet/
- http://github.com/CiscoDevNet



© 2017 Cisco and/or its affiliates. All rights reserved.   Cisco Confidential