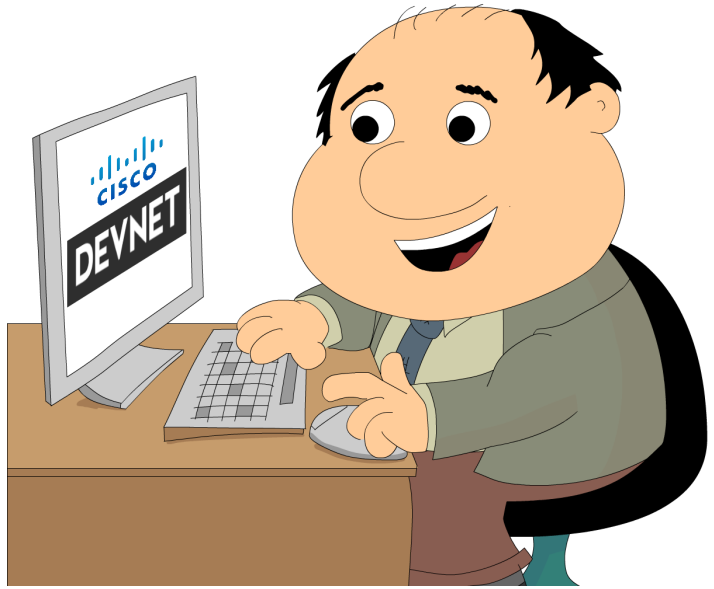
REST APIs Part 1: HTTP is for more than Web Browsing

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



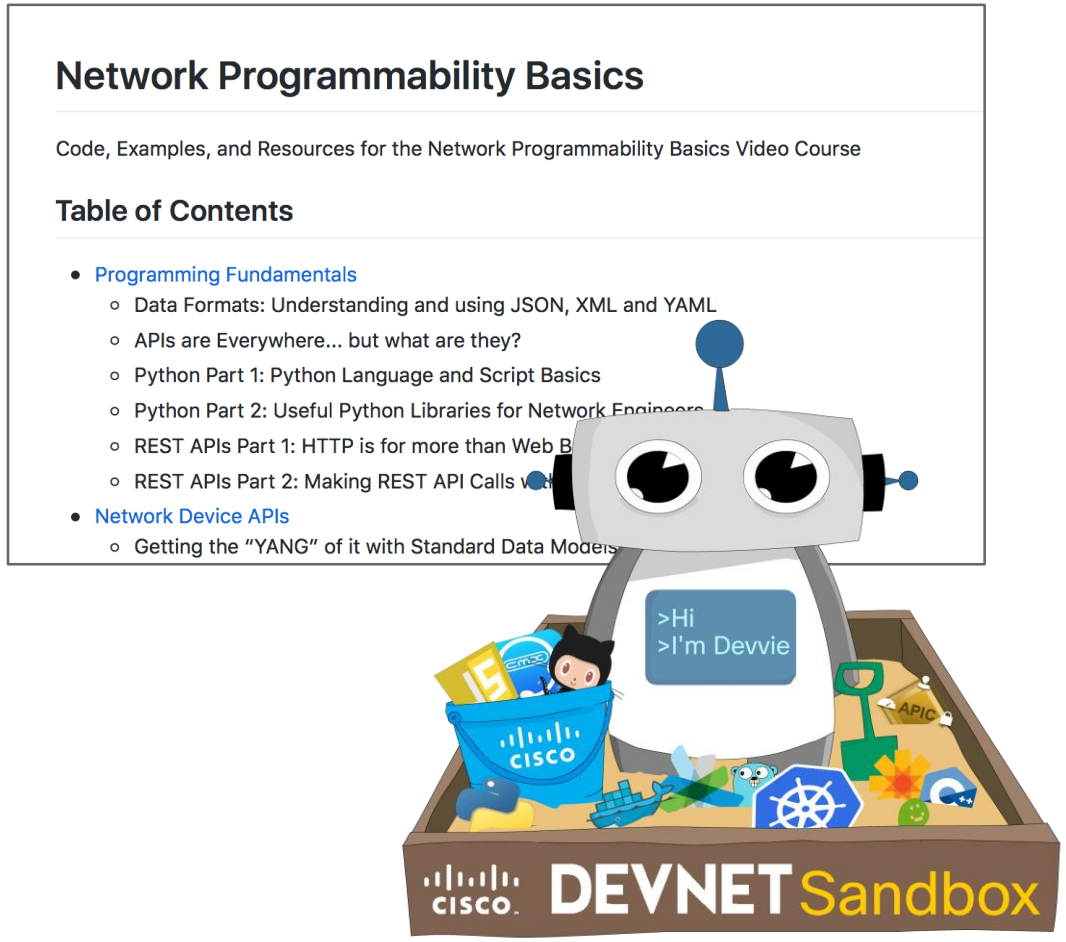
Network Programmability Basics: The Lessons

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

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



Topics to Cover

- What is REST?
- A Look Under the Hood at REST?
- Some REST Examples
- REST API Tools

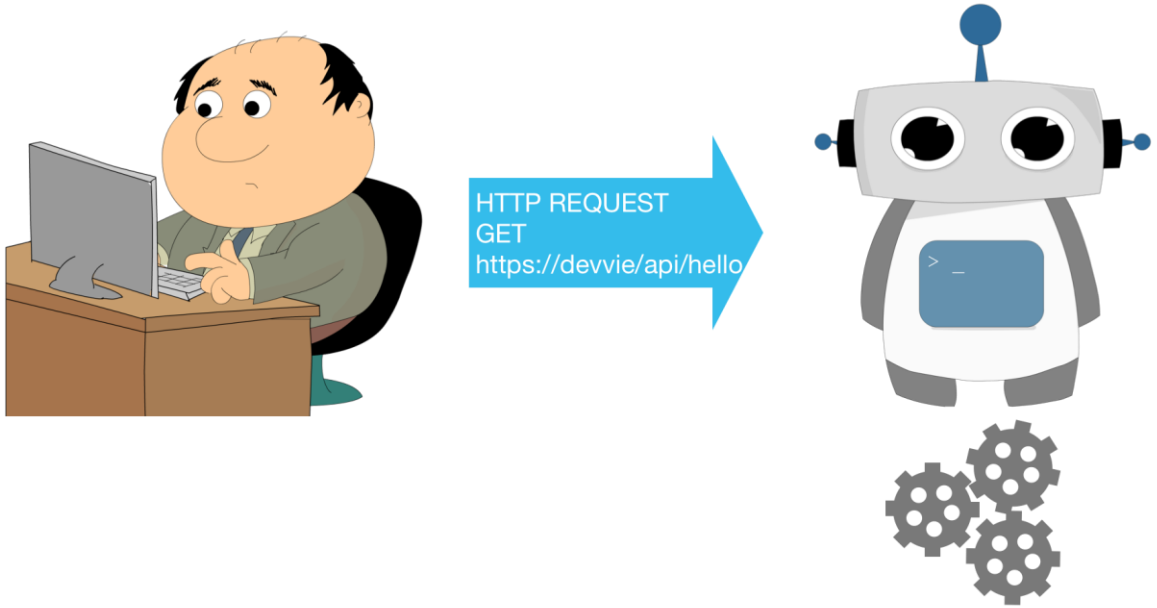
What is REST?

Just Another Use for the HTTP Protocol

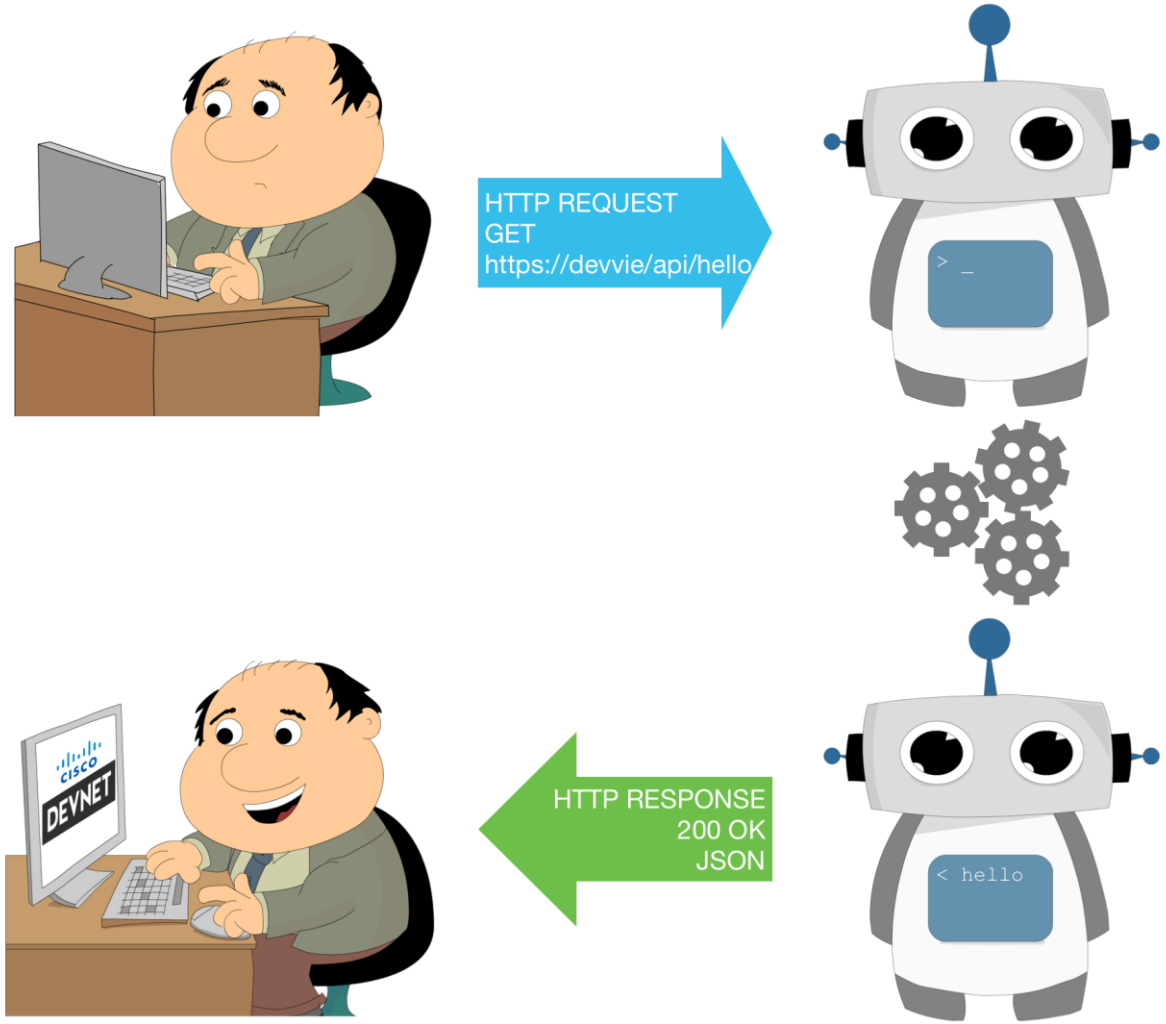
- Representational state transfer (REST)
- API framework built on HTTP
- APIs often referred to as web services
- Popular due to performance, scale, simplicity, and reliability

Requests and Response, the REST API Flow

Requests and Response, the REST API Flow



Requests and Response, the REST API Flow



A Look Under the Hood at REST?

The URI: What are you Requesting?

- http:// or https://
  - Define whether secure or open http
- Server or Host
  - Resolves to the IP and port to connect to
- Resource
  - The location of the data or object of interest on the server
- Parameters
  - Details to scope, filter, or clarify a request. Often optional.

HTTP Methods: What to do?

HTTP Verb	Typical Purpose (CRUD)	Description
POST	Create	Used to create a new object, or resource. *Example: Add new book to library*
GET	Read	Retrieve resource details from the system. *Example: Get list of books from the library*
PUT	Update	Typically used to replace or update a resource. Can be used to modify or create. *Example: Update the borrower details for a book*
PATCH	Update	Used to modify some details about a resource. *Example: Change the author of a book*
DELETE	Delete	Remove a resource from the system. *Example: Delete a book from the library.*

Response Status Codes: Did it work?

Status Code	Status Message	Meaning
200	OK	All looks good
404	Not Found	Resource not found

Response Status Codes: Did it work?

Status Code	Status Message	Meaning
200	OK	All looks good
201	Created	New resource created
400	Bad Request	Request was invalid
401	Unauthorized	Authentication missing or incorrect
403	Forbidden	Request was understood, but not allowed
404	Not Found	Resource not found
500	Internal Server Error	Something wrong with the server
503	Service Unavailable	Server is unable to complete request

Headers: Details and meta-data

Header	Example Value	Purpose
Content-Type	application/json	Specify the format of the data in the body
Accept	application/json	Specify the requested format for returned data
Authorization	Basic dmFncmFudDp2YWdyYW50	Provide credentials to authorize a request
Date	Tue, 25 Jul 2017 19:26:00 GMT	Date and time of the message

- Used to pass information between client and server
- Included in both REQUEST and RESPONSE
- Some APIs will use custom headers for authentication or other purpose

Data: Sending and Receiving

- Contained in the body
- POST, PUT, PATCH requests typically include data
- GET responses will include data
- Format typically JSON or XML
  - Check “Content-Type” header

```
{
    'title': 'Hamlet',
    'author': 'Shakespeare'
}
```

HTTP Authentication and Security

- None: the Web API resource is public, anybody can place call.
- Basic HTTP: a username and password are passed to the server in an encoded string.
  - Authorization: Basic ENCODEDSTRING
- Token: a secret generally retrieved from the Web API developer portal. Keyword (ie token) is API dependent
  - Authorization: Token aikasf8adf9asd9akasdf0asd
- OAuth: Standard framework for a flow to retrieve an access token from an Identity Provider.
  - Authorization: Bearer 8a9af9adadf0asdf0adfa0af
- Authorization can be short-lived and require refreshing of tokens

Some REST Examples

The Internet Chuck Norris Database

```
DevNet$ curl https://api.icndb.com/jokes/random

{
    "type": "success",
    "value": {
        "id": 201,
        "joke": "Chuck Norris was what Willis was talkin' about.",
        "categories": []
    }
}

DevNet$ curl https://api.icndb.com/jokes/random?limitTo=nerdy

{
    "type": "success",
    "value": {
        "id": 537,
        "joke": "Each hair in Chuck Norris's beard contributes to make the world's largest DDOS.",
        "categories": [
            "nerdy"
        ]
    }
}
```

- http://www.icndb.com/api/
- No authentication needed
- Well constructed API with many options

Network Programmability with RESTCONF

The Request

```
DevNet$ curl -vk \
  -u root:D_Vay\!_10\& \
  -H 'accept: application/yang-data+json' \
https://ios-xe-mgmt.cisco.com:9443/restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2

> GET /restconf/data/ietf-interfaces:interfaces/interface=GigabitEthernet2 HTTP/1.1
> Host: 10.10.20.21
> User-Agent: curl/7.51.0
> accept: application/yang-data+json
> authorization: Basic dmFncmFudDp2YWdyYW50
>
```

- -u provides `user:password` for Basic Authentication
- -H to set headers
- Lines beginning with “>” indicate Request elements
- Lines beginning with “<” indicate Response elements (next slide)

Network Programmability with RESTCONF

The Response - Headers

```
< HTTP/1.1 200 OK
< Server: nginx
< Date: Thu, 27 Jul 2017 00:01:52 GMT
< Content-Type: application/yang-data+json
< Transfer-Encoding: chunked
< Connection: close
< Last-Modified: Tue, 25 Jul 2017 19:15:57 GMT
< Cache-Control: private, no-cache, must-
revalidate, proxy-revalidate
< Etag: 1501-10157-179272
< Pragma: no-cache
<
```

The Response - Data

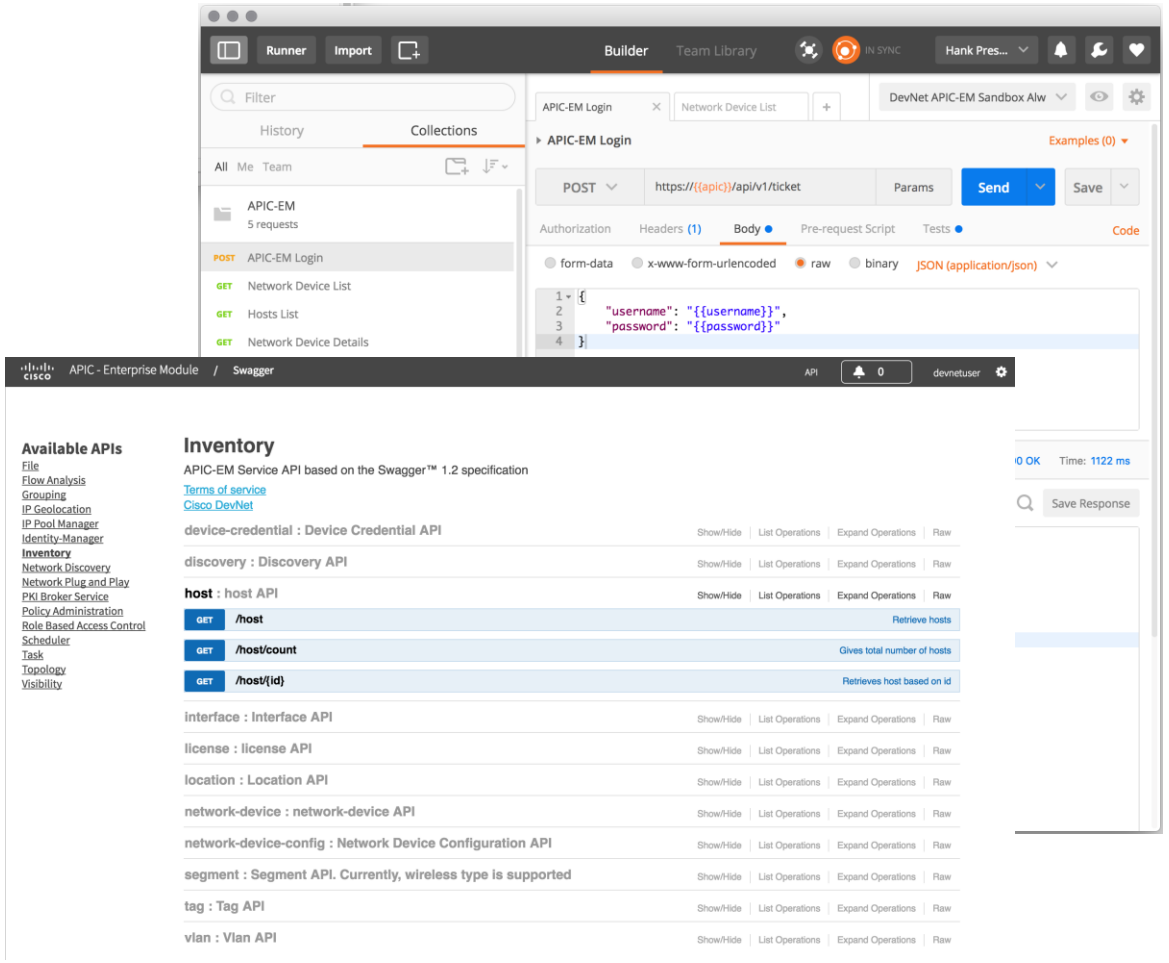
```
{
  "ietf-interfaces:interface": {
    "name": "GigabitEthernet2",
    "description": "Wide Area Network",
    "type": "iana-if-type:ethernetCsmacd",
    "enabled": true,
    "ietf-ip:ipv4": {
      "address": [
        {
          "ip": "172.16.0.2",
          "netmask": "255.255.255.0"
        }
      ]
    },
    "ietf-ip:ipv6": {
    }
  }
}
```

Demo Time!

REST API Tools

Many Options for Working with REST APIs

- curl
  - Linux command line application
- Postman
  - Chrome browser plugin and application
- Requests
  - Python library for scripting
- Swagger
  - Dynamic API Documentation
- Browser Developer Tools
  - View traffic and details within browser



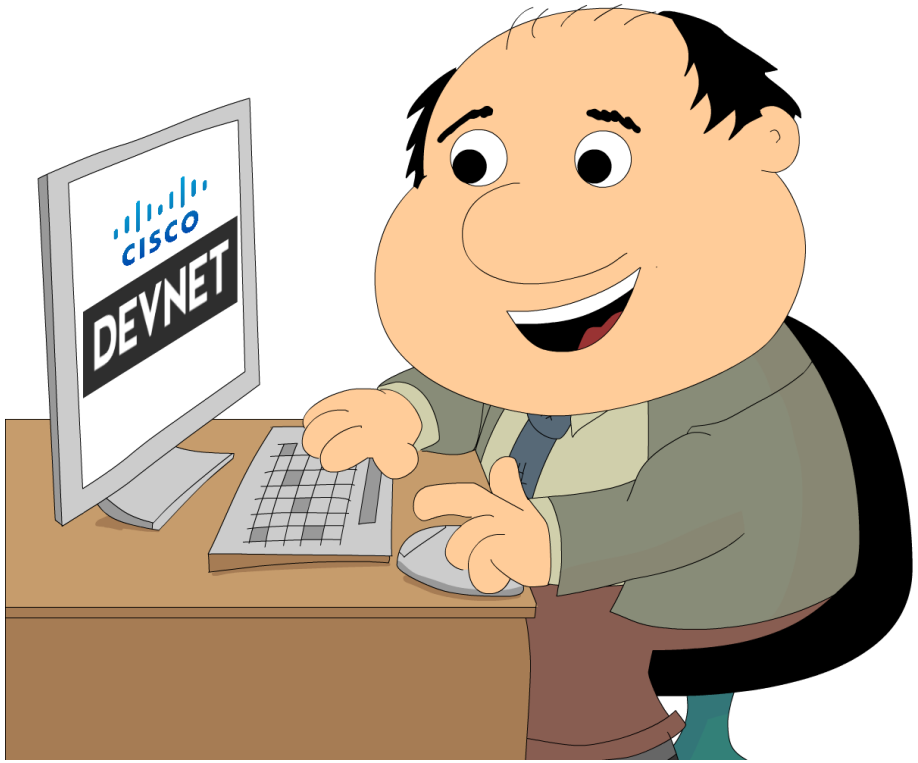
Summing up

Review

- REST APIs are built on the HTTP Protocol
- Requests and Responses
- How are URIs constructed
- Methods, Status Codes, and Headers used with REST APIs
- Authentication options for HTTP
- Looked at some example API calls

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



Got more questions? Come find me!

hapresto@cisco.com

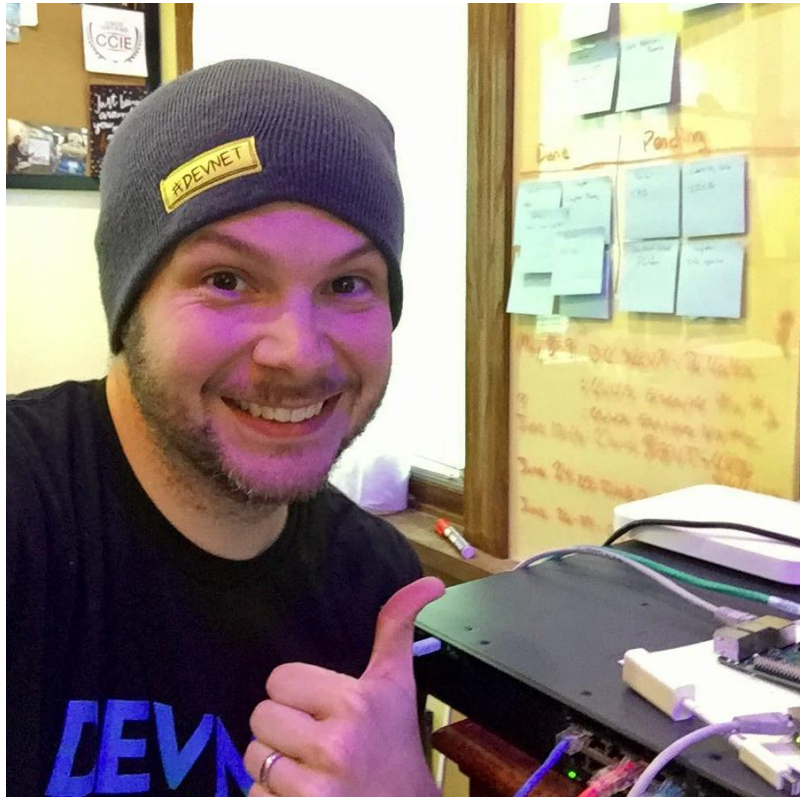
@hfpreston

http://github.com/hpreston

@CiscoDevNet

facebook.com/ciscodevnet/

http://github.com/CiscoDevNet




CISCO
DEVNET