You make **possible**

Cisco DNA Center API Use Case: Automating SDA Deployment Tasks

Richard Cunningham, CX Architect
Lawrence Zhu, CX Architect

DEVNET-2275

Barcelona | January 27-31, 2020

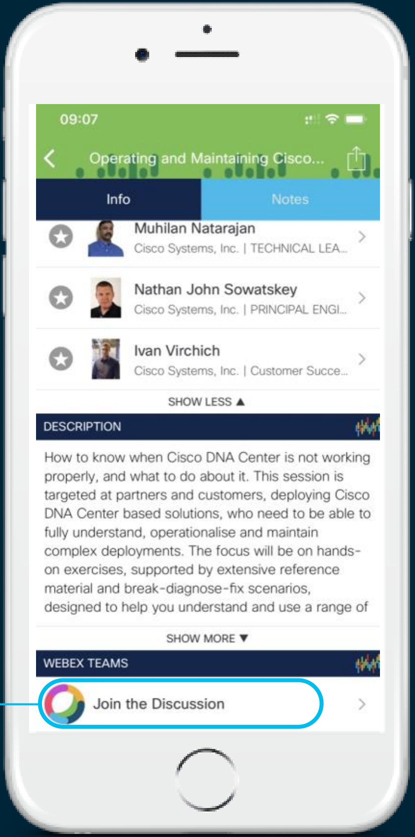
Cisco Webex Teams

Questions?

Use Cisco Webex Teams to chat with the speaker after the session

How

1. Find this session in the Cisco Events Mobile App
2. Click "Join the Discussion"
3. Install Webex Teams or go directly to the team space
4. Enter messages/questions in the team space


09:07
Operating and Maintaining Cisco...
Info
Notes
Muhilan Natarajan
Cisco Systems, Inc. | TECHNICAL LEA...
Nathan John Sowatskey
Cisco Systems, Inc. | PRINCIPAL ENGI...
Ivan Virchich
Cisco Systems, Inc. | Customer Succe...
SHOW LESS ▲
DESCRIPTION
SHOW MORE ▼
WEBEX TEAMS
Join the Discussion

How to know when Cisco DNA Center is not working properly, and what to do about it. This session is targeted at partners and customers, deploying Cisco DNA Center based solutions, who need to be able to fully understand, operationalise and maintain complex deployments. The focus will be on hands-on exercises, supported by extensive reference material and break-diagnose-fix scenarios, designed to help you understand and use a range of

Agenda

- Getting Started with DNA Center Workflow Automation
- Example Framework
- Sample Workflows / Demo

Code examples can be found at:

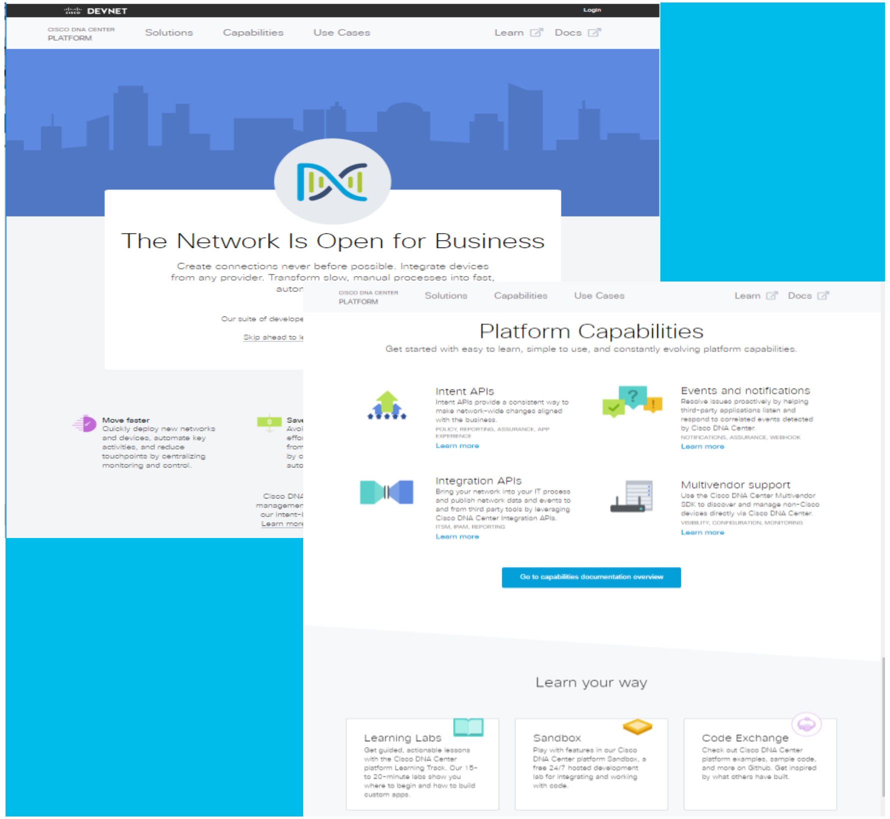
https://github.com/cunningr/dna_workflows/tree/master

Getting Started with DNA Center Automation

Cisco DNA Center Platform on DevNet

- API and SDK Library
- Learning Labs
- Sandbox
- Partner use cases
- Support and Community
- Other resources (documentation)

https://developer.cisco.com/dnacenter/



DEVNET Login

CISCO DNA CENTER PLATFORM Solutions Capabilities Use Cases Learn Docs

The Network Is Open for Business

Create connections never before possible. Integrate devices from any provider. Transform slow, manual processes into fast, auton

Our suite of develope

Skip ahead to le

Move faster
Quickly deploy new networks and devices, automate key activities, and reduce touchpoints by centralizing monitoring and control.

Sav
Avoi
effor
from
by c
auto

Cisco DNA
managemen
our intent-
Learn more

CISCO DNA CENTER PLATFORM Solutions Capabilities Use Cases Learn Docs

Platform Capabilities

Get started with easy to learn, simple to use, and constantly evolving platform capabilities.

Intent APIs
Intent APIs provide a consistent way to make network-wide changes aligned with the business.
POLICY, REPORTING, ASSURANCE, APP EXPERIENCE
Learn more

Events and notifications
Resolve issues proactively by helping third-party applications listen and respond to correlated events detected by Cisco DNA Center.
NOTIFICATIONS, ASSURANCE, WEBHOOK
Learn more

Integration APIs
Bring your network into your IT process and publish network data and events to and from third party tools by leveraging Cisco DNA Center Integration APIs.
ITSM, IPAM, REPORTING
Learn more

Multivendor support
Use the Cisco DNA Center Multivendor SDK to discover and manage non-Cisco devices directly via Cisco DNA Center.
VISIBILITY, CONFIGURATION, MONITORING
Learn more

Go to capabilities documentation overview

Learn your way

Learning Labs
Get guided, actionable lessons with the Cisco DNA Center platform Learning Track. Our 15- to 20-minute labs show you where to begin and how to build custom apps.

Sandbox
Play with features in our Cisco DNA Center platform Sandbox, a free 24/7 hosted development lab for integrating and working with code.

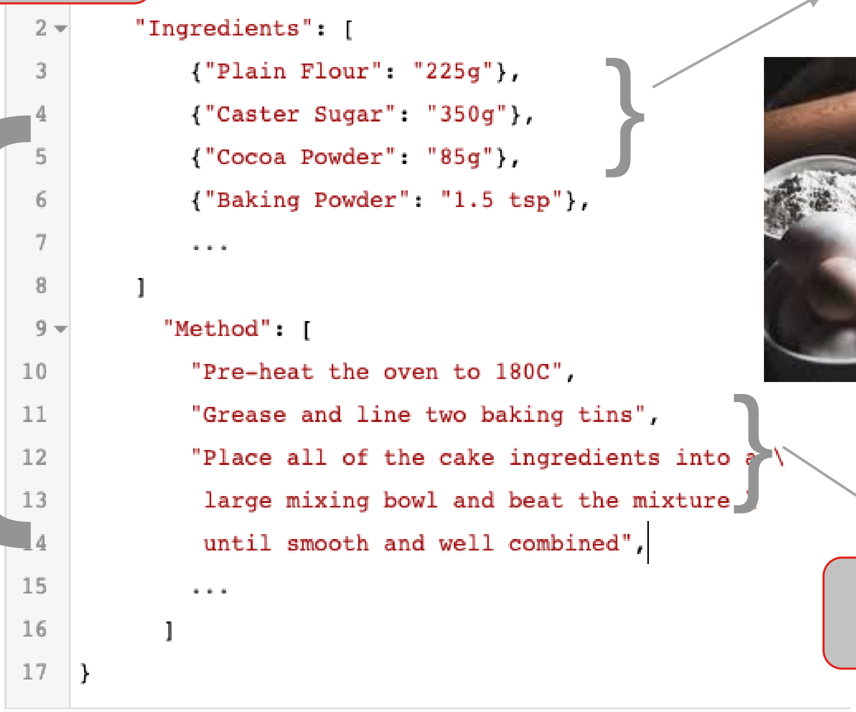
Code Exchange
Check out Cisco DNA Center platform examples, sample code, and more on Github. Get inspired by what others have built.

DNA Center Workflow Automation – Key Ingredients

Data Exchange

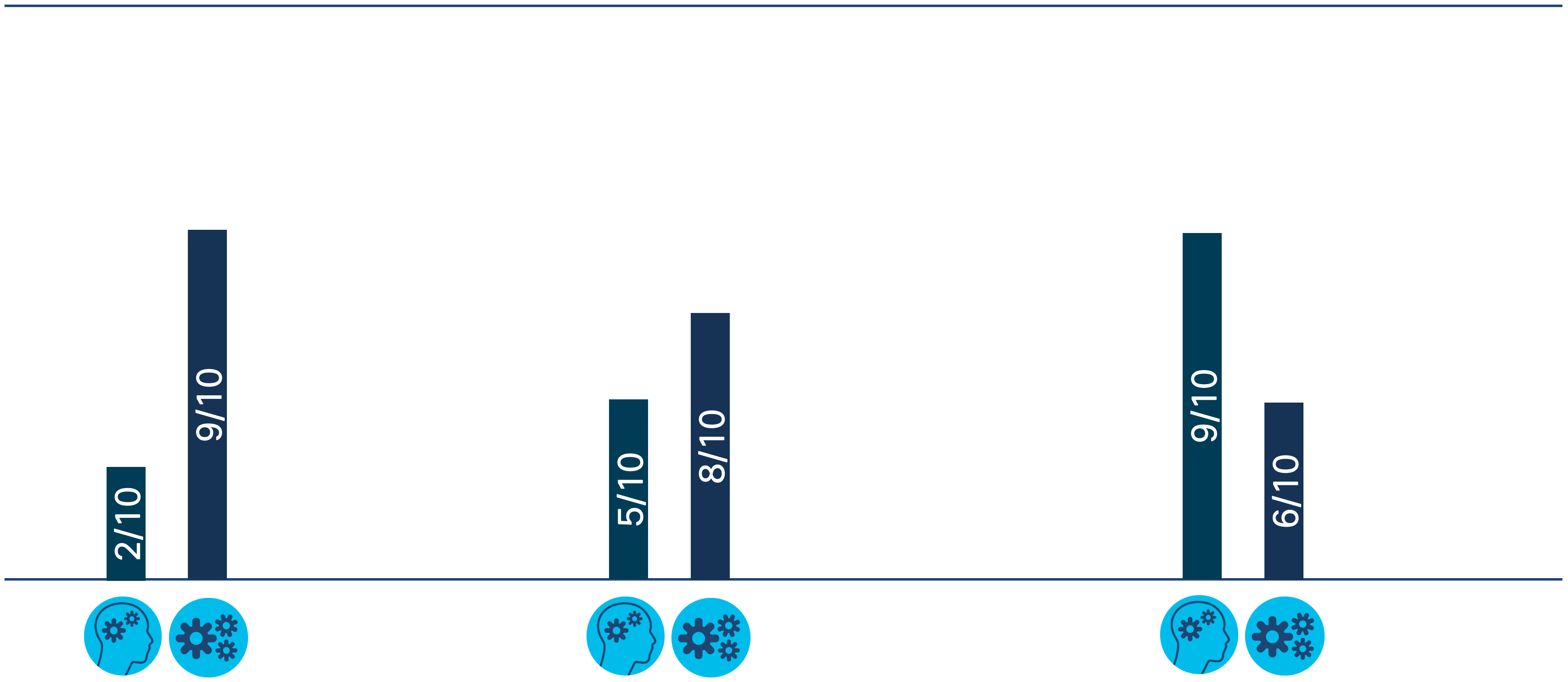
Structured Data

Task Sequencing

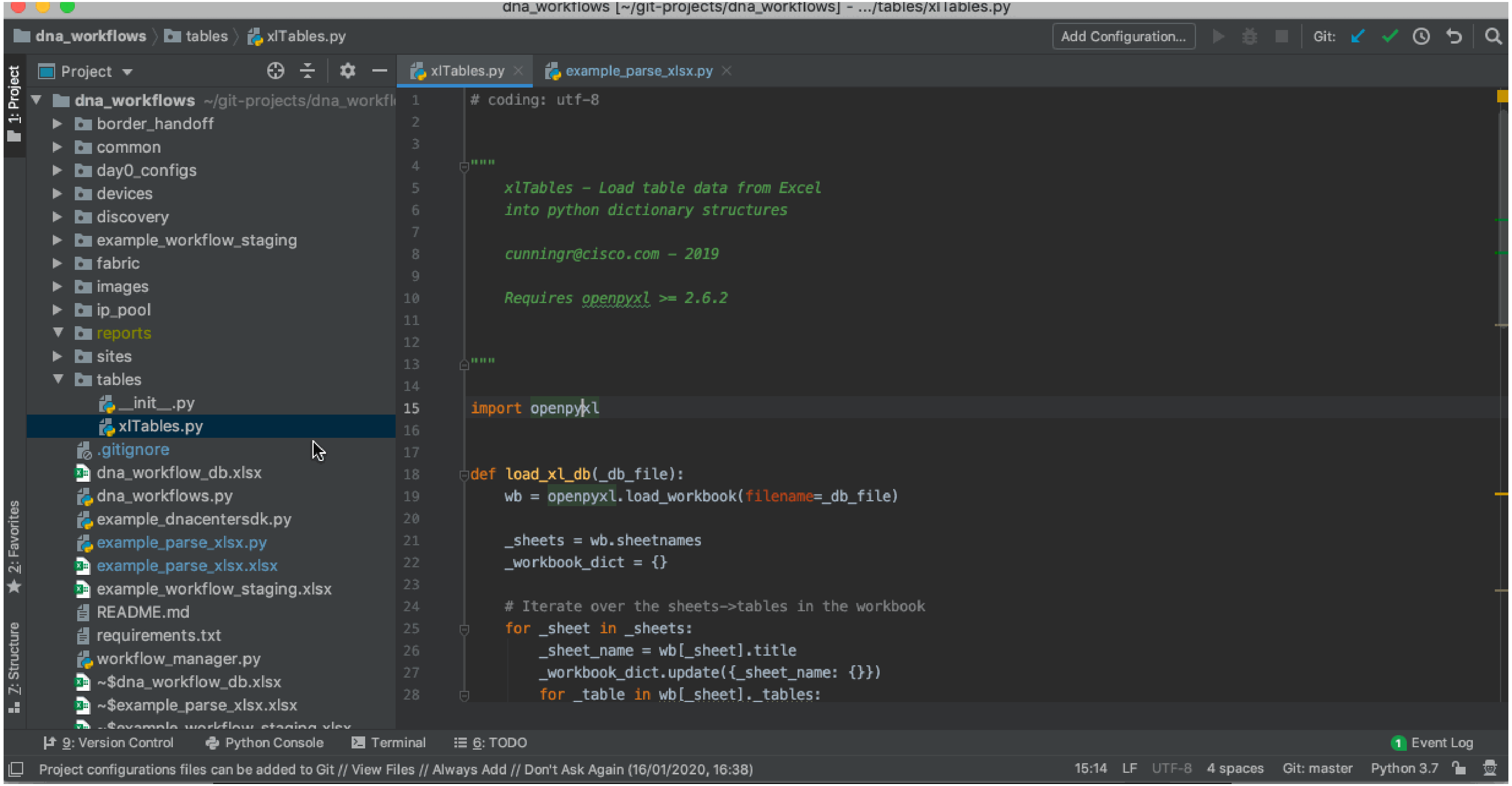


```
    "Ingredients": [
        {"Plain Flour": "225g"},
        {"Caster Sugar": "350g"},
        {"Cocoa Powder": "85g"},
        {"Baking Powder": "1.5 tsp"},
        ...
    ]
      "Method": [
        "Pre-heat the oven to 180C",
        "Grease and line two baking tins",
        "Place all of the cake ingredients into a \
         large mixing bowl and beat the mixture \
         until smooth and well combined",
        ...
      ]
}
```

Structured Data



Demo
Structured Data with Excel

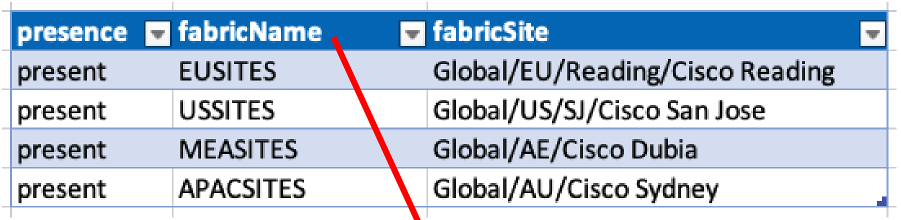


```
dna_workflows ~/git-projects/dna_workfl
    border_handoff
    common
    day0_configs
    devices
    discovery
    example_workflow_staging
    fabric
    images
    ip_pool
    reports
    sites
    tables
        __init__.py
        xlTables.py
    .gitignore
    dna_workflow_db.xlsx
    dna_workflows.py
    example_dnacentersdk.py
    example_parse_xlsx.py
    example_parse_xlsx.xlsx
    example_workflow_staging.xlsx
    README.md
    requirements.txt
    workflow_manager.py
    ~$dna_workflow_db.xlsx
    ~$example_parse_xlsx.xlsx
```

```python
# coding: utf-8


"""
    xlTables - Load table data from Excel
    into python dictionary structures

    cunningr@cisco.com - 2019

    Requires openpyxl >= 2.6.2


"""

import openpyxl


def load_xl_db(_db_file):
    wb = openpyxl.load_workbook(filename=_db_file)

    _sheets = wb.sheetnames
    _workbook_dict = {}

    # Iterate over the sheets->tables in the workbook
    for _sheet in _sheets:
        _sheet_name = wb[_sheet].title
        _workbook_dict.update({_sheet_name: {}})
        for _table in wb[_sheet]._tables:
```

```
Project configurations files can be added to Git // View Files // Always Add // Don't Ask Again (16/01/2020, 16:38)    15:14  LF  UTF-8  4 spaces  Git: master  Python 3.7
```

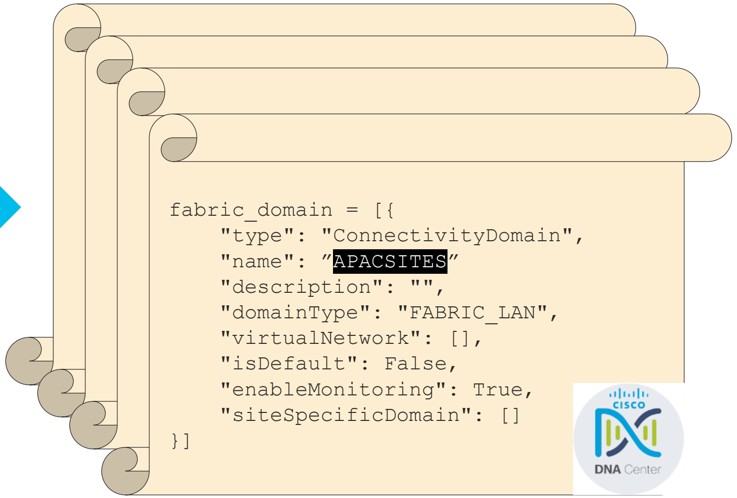
Payload Templates - Filling in the gaps



presence	fabricName	fabricSite
present	EUSITES	Global/EU/Reading/Cisco Reading
present	USSITES	Global/US/SJ/Cisco San Jose
present	MEASITES	Global/AE/Cisco Dubia
present	APACSITES	Global/AU/Cisco Sydney

```
fabric_domain = [{
        "type": "ConnectivityDomain",
        "name": "{{ fabricName }}",
        "description": "",
        "domainType": "FABRIC_LAN",
        "virtualNetwork": [],
        "isDefault": False,
        "enableMonitoring": True,
        "siteSpecificDomain": []
}]
```

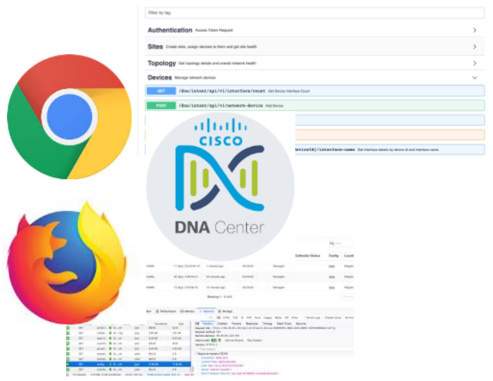
- Using Jinja2 to parse payload templates.
- Many fields may not need to be exposed to the user.



```
fabric_domain = [{
    "type": "ConnectivityDomain",
    "name": "APACSITES"
    "description": "",
    "domainType": "FABRIC_LAN",
    "virtualNetwork": [],
    "isDefault": False,
    "enableMonitoring": True,
    "siteSpecificDomain": []
}]
```

Exchanging Data with DNA Center

Data Exchange - Interacting with DNA Center


Filter by tag

Authentication Access Token Request

Sites Create sites, assign devices to them and get site health

Topology Get topology details and overall network health

Devices Manage network devices

GET /dna/intent/api/v1/interface/count Get Device Interface Count

POST /dna/intent/api/v1/network-device Add Device

...deviceId}/interface-name Get interface details by device id and interface name

POSTMAN

curl://

JSONLint - The JSON Validator

HTTPie

YAML

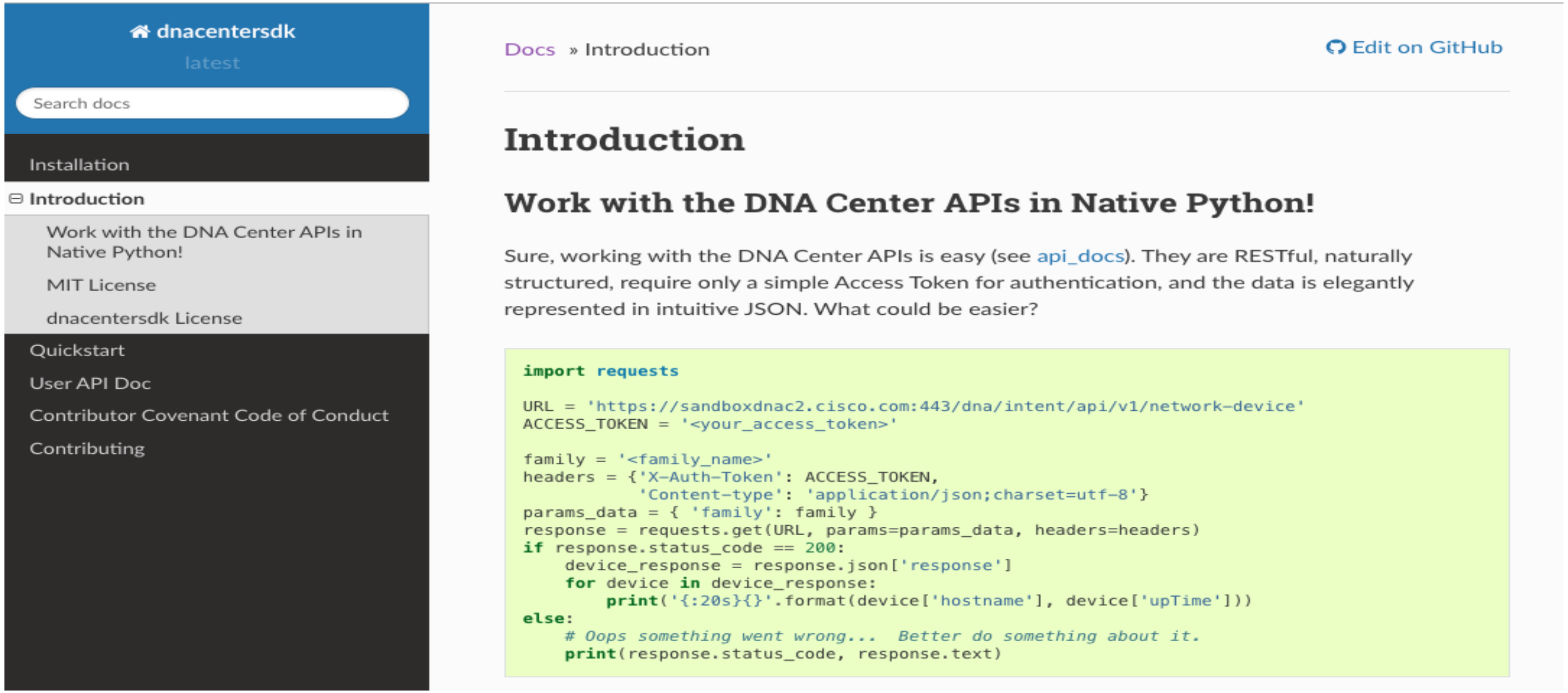
JSON

Requests

urllib

DNA Center API SDK

- https://dnacentersdk.readthedocs.io/en/latest/api/api.html



Introduction

Work with the DNA Center APIs in Native Python!

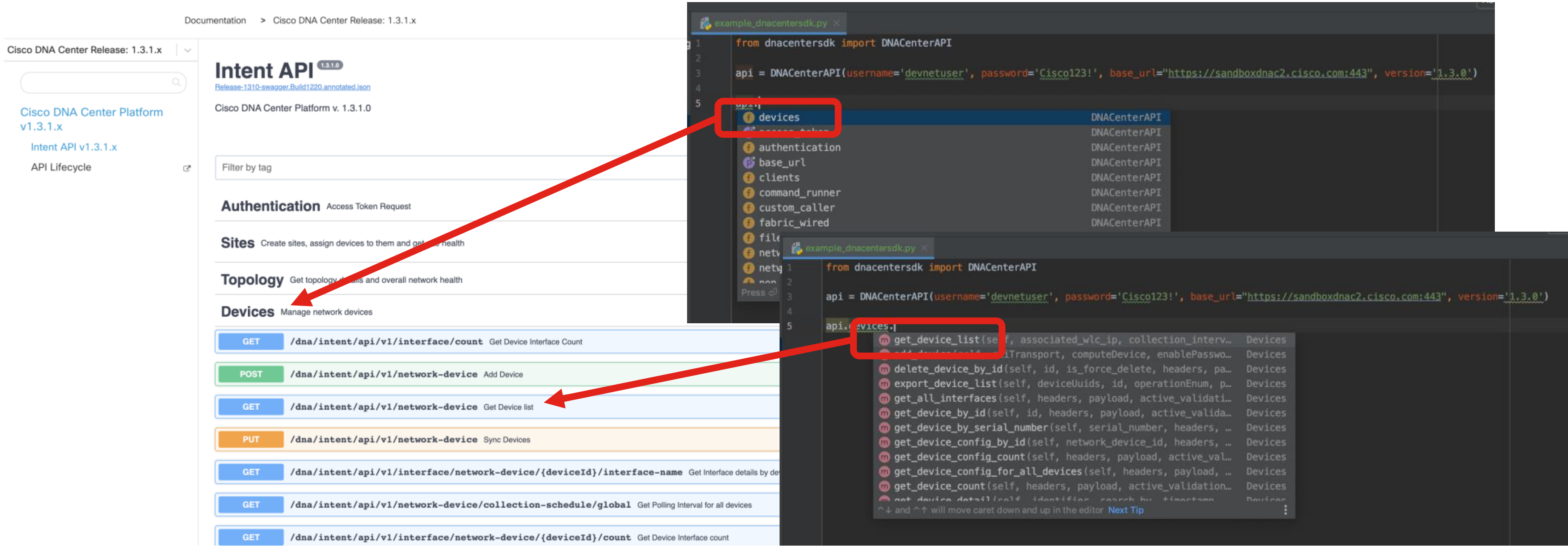
Sure, working with the DNA Center APIs is easy (see api_docs). They are RESTful, naturally structured, require only a simple Access Token for authentication, and the data is elegantly represented in intuitive JSON. What could be easier?

```python
import requests

URL = 'https://sandboxdnac2.cisco.com:443/dna/intent/api/v1/network-device'
ACCESS_TOKEN = '<your_access_token>'

family = '<family_name>'
headers = {'X-Auth-Token': ACCESS_TOKEN,
           'Content-type': 'application/json;charset=utf-8'}
params_data = { 'family': family }
response = requests.get(URL, params=params_data, headers=headers)
if response.status_code == 200:
    device_response = response.json['response']
    for device in device_response:
        print('{:20s}{}'.format(device['hostname'], device['upTime']))
else:
    # Oops something went wrong...  Better do something about it.
    print(response.status_code, response.text)
```

DNA Center SDK

- Based on the DNA Center Intent API

https://developer.cisco.com/docs/dna-center/api/1-3-1-x/


Documentation > Cisco DNA Center Release: 1.3.1.x

Cisco DNA Center Release: 1.3.1.x

Cisco DNA Center Platform v1.3.1.x
Intent API v1.3.1.x
API Lifecycle

Intent API 1.3.1.0
Release-1310-swagger.Build1220.annotated.json

Cisco DNA Center Platform v. 1.3.1.0

Filter by tag

Authentication Access Token Request

Sites Create sites, assign devices to them and get … health

Topology Get topology … and overall network health

Devices Manage network devices

GET	`/dna/intent/api/v1/interface/count`	Get Device Interface Count
POST	`/dna/intent/api/v1/network-device`	Add Device
GET	`/dna/intent/api/v1/network-device`	Get Device list
PUT	`/dna/intent/api/v1/network-device`	Sync Devices
GET	`/dna/intent/api/v1/interface/network-device/{deviceId}/interface-name`	Get Interface details by de
GET	`/dna/intent/api/v1/network-device/collection-schedule/global`	Get Polling Interval for all devices
GET	`/dna/intent/api/v1/interface/network-device/{deviceId}/count`	Get Device Interface count

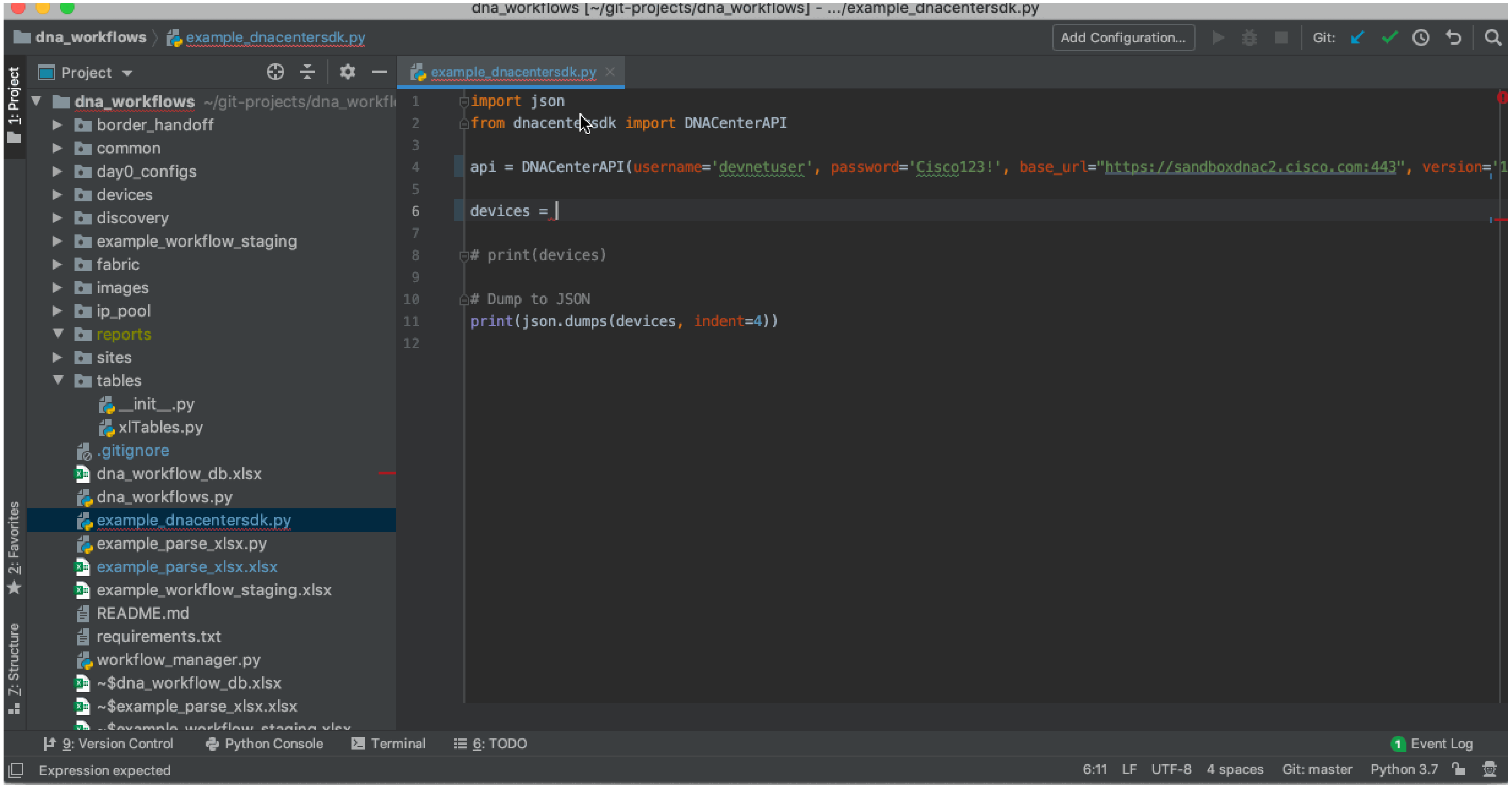
example_dnacentersdk.py

```
from dnacentersdk import DNACenterAPI

api = DNACenterAPI(username='devnetuser', password='Cisco123!', base_url="https://sandboxdnac2.cisco.com:443", version='1.3.0')

api.
  devices                                   DNACenterAPI
                                            DNACenterAPI
  authentication                            DNACenterAPI
  base_url                                  DNACenterAPI
  clients                                   DNACenterAPI
  command_runner                            DNACenterAPI
  custom_caller                             DNACenterAPI
  fabric_wired                              DNACenterAPI
  file
  netw
  netw
Press
```

example_dnacentersdk.py

```
from dnacentersdk import DNACenterAPI

api = DNACenterAPI(username='devnetuser', password='Cisco123!', base_url="https://sandboxdnac2.cisco.com:443", version='1.3.0')

api.devices.
  get_device_list(self, associated_wlc_ip, collection_interv…            Devices
  …iTransport, computeDevice, enablePasswo…                              Devices
  delete_device_by_id(self, id, is_force_delete, headers, pa…            Devices
  export_device_list(self, deviceUuids, id, operationEnum, p…            Devices
  get_all_interfaces(self, headers, payload, active_validati…            Devices
  get_device_by_id(self, id, headers, payload, active_valida…            Devices
  get_device_by_serial_number(self, serial_number, headers, …            Devices
  get_device_config_by_id(self, network_device_id, headers, …            Devices
  get_device_config_count(self, headers, payload, active_val…            Devices
  get_device_config_for_all_devices(self, headers, payload, …            Devices
  get_device_count(self, headers, payload, active_validation…            Devices
  get_device_detail(self, identifier, search_by, timestamp…              Devices
^↓ and ^↑ will move caret down and up in the editor Next Tip
```

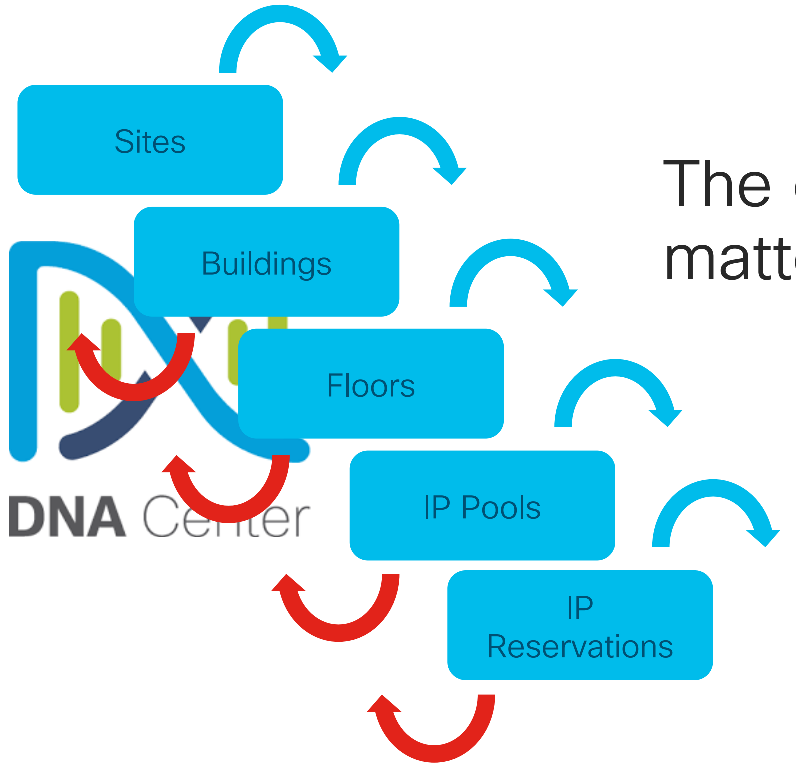
Demo
Using the DNA Center SDK



```python
import json
from dnacentersdk import DNACenterAPI

api = DNACenterAPI(username='devnetuser', password='Cisco123!', base_url="https://sandboxdnac2.cisco.com:443", version='1

devices = 

# print(devices)

# Dump to JSON
print(json.dumps(devices, indent=4))
```

Building Workflows

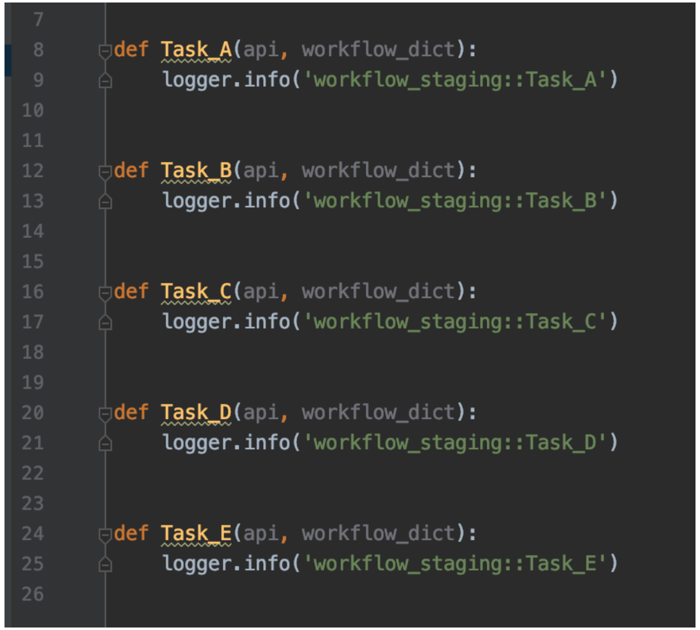
Workflow – Task Sequencing



The order in which we do things matters!

Workflow – Organising Code

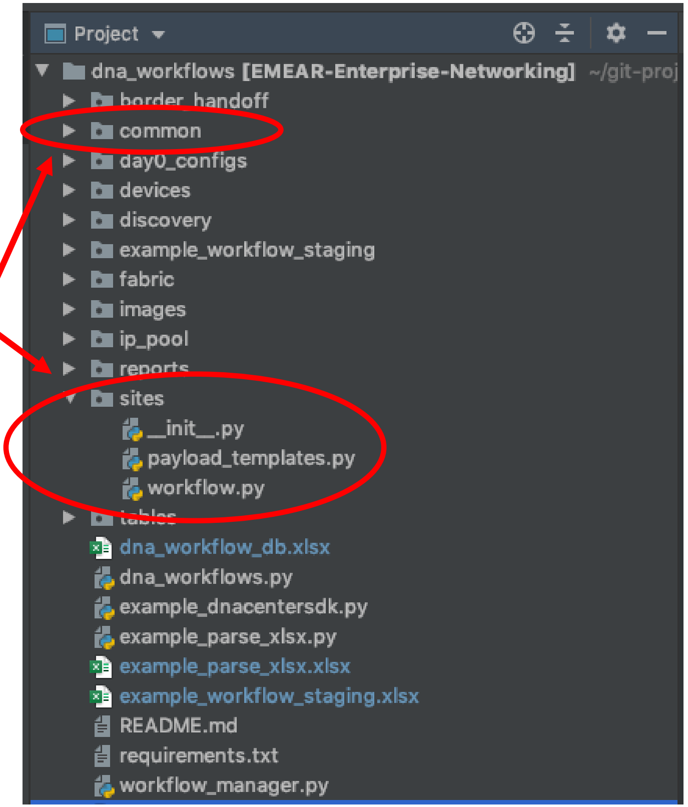
- Arranging code into functions makes it easy to read and maintain.
- Create ‘task’ based functions that do one thing and do it well.
- Allow easy reordering of tasks or restructuring of workflow.



```python

def Task_A(api, workflow_dict):
    logger.info('workflow_staging::Task_A')


def Task_B(api, workflow_dict):
    logger.info('workflow_staging::Task_B')


def Task_C(api, workflow_dict):
    logger.info('workflow_staging::Task_C')


def Task_D(api, workflow_dict):
    logger.info('workflow_staging::Task_D')


def Task_E(api, workflow_dict):
    logger.info('workflow_staging::Task_E')

```

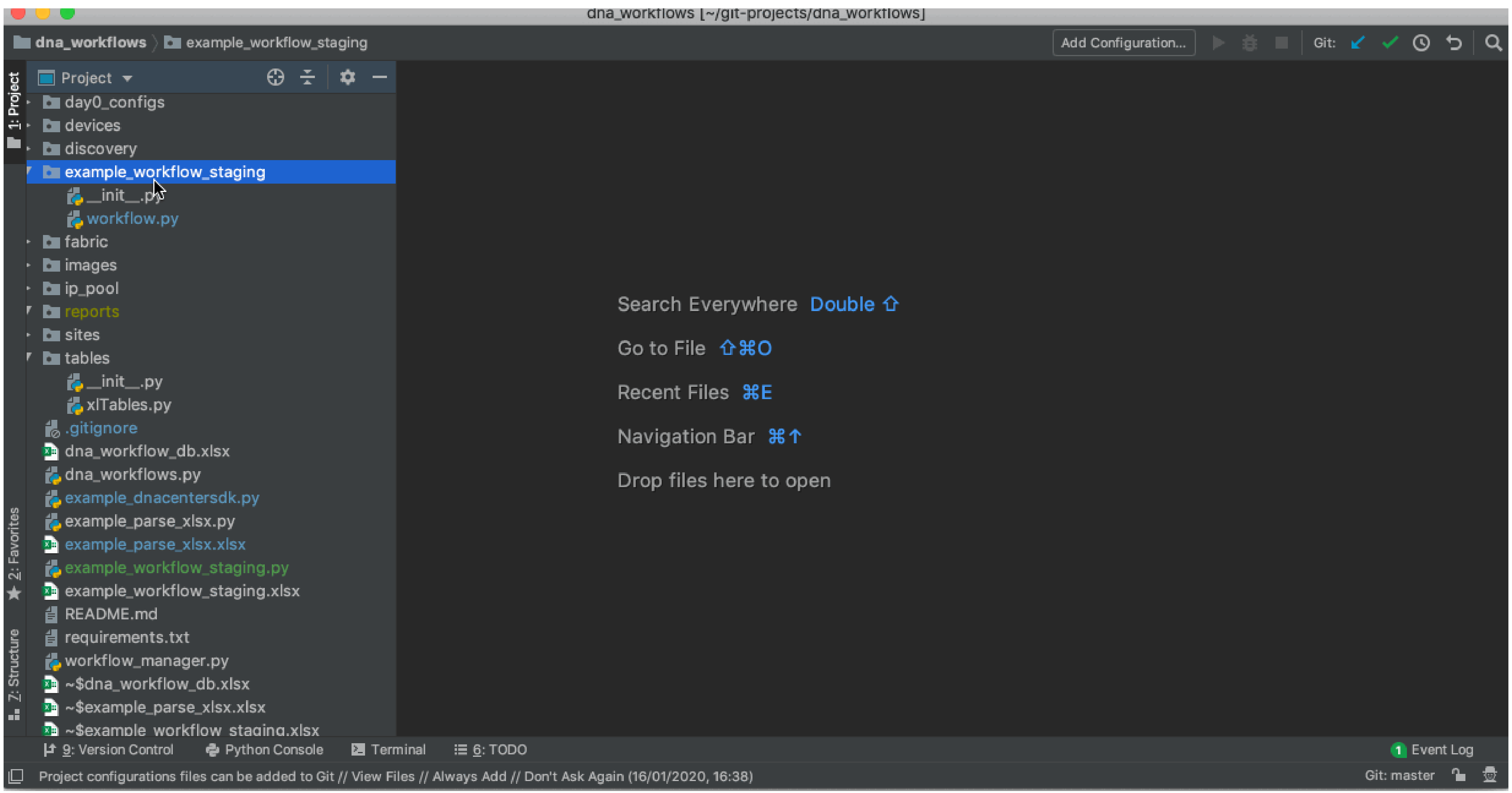
Workflow – Organising Code

- Use python modules to organize your code.
- Python 'import' statements will look in the local 'run dir' for folders with the module name.
- Maintain 'common' code in a separate file or module.



```
dna_workflows [EMEAR-Enterprise-Networking] ~/git-proj
  border_handoff
  common
  day0_configs
  devices
  discovery
  example_workflow_staging
  fabric
  images
  ip_pool
  reports
  sites
    __init__.py
    payload_templates.py
    workflow.py
  tables
  dna_workflow_db.xlsx
  dna_workflows.py
  example_dnacentersdk.py
  example_parse_xlsx.py
  example_parse_xlsx.xlsx
  example_workflow_staging.xlsx
  README.md
  requirements.txt
  workflow_manager.py
```

Demo
Modules, Tasks and Workflow Control


dna_workflows [~/git-projects/dna_workflows]

dna_workflows 〉 example_workflow_staging

Add Configuration... Git:

Project

- day0_configs
- devices
- discovery
- example_workflow_staging
  - __init__.py
  - workflow.py
- fabric
- images
- ip_pool
- reports
- sites
- tables
  - __init__.py
  - xlTables.py
- .gitignore
- dna_workflow_db.xlsx
- dna_workflows.py
- example_dnacentersdk.py
- example_parse_xlsx.py
- example_parse_xlsx.xlsx
- example_workflow_staging.py
- example_workflow_staging.xlsx
- README.md
- requirements.txt
- workflow_manager.py
- ~$dna_workflow_db.xlsx
- ~$example_parse_xlsx.xlsx
- ~$example_workflow_staging.xlsx

Search Everywhere Double ⇧

Go to File ⇧⌘O

Recent Files ⌘E

Navigation Bar ⌘↑

Drop files here to open

```
Project configurations files can be added to Git // View Files // Always Add // Don't Ask Again (16/01/2020, 16:38)
```

Git: master

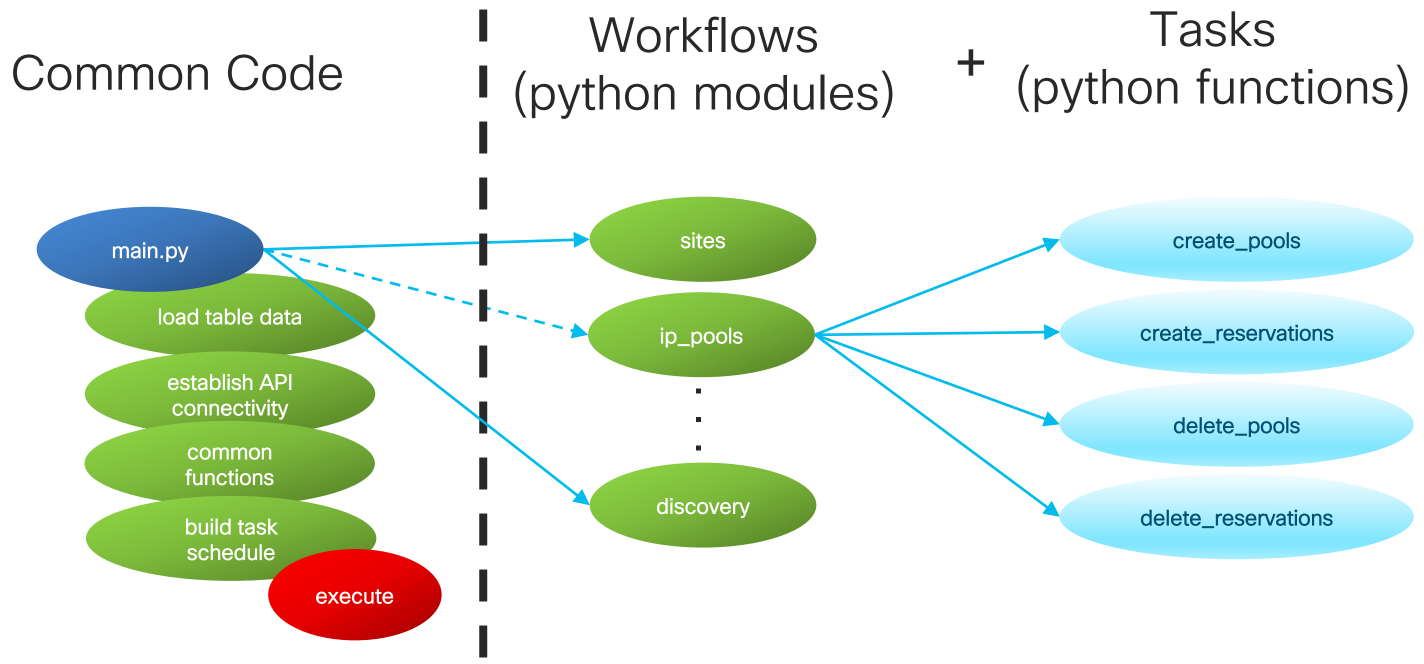
An Example Framework

Example Framework

- Ingestion of structured configuration data -> Excel
- Common API handling -> dnacentersdk
- Task scheduling -> Task execution from Excel
- Logging framework -> python logging
- JSON payload templates -> Jinja2 templates
- Status reporting -> SUCCESS|IN PROGRESS|FAIL
- Example workflows (sites, ip_pools, discovery)
- Workflow module management (create | export)

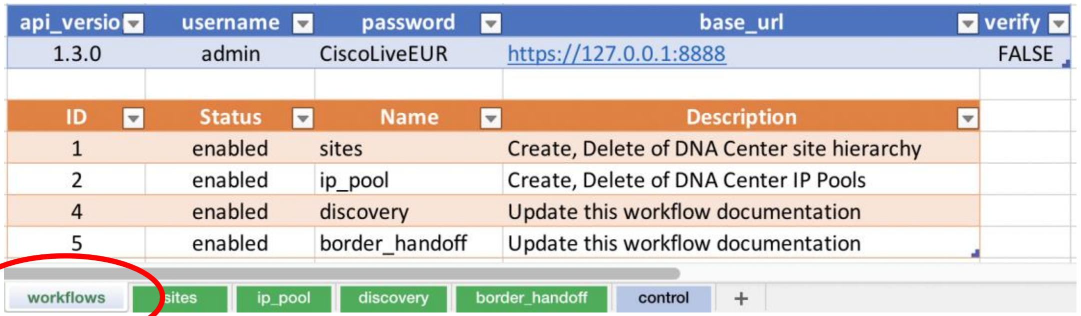
Git Repository: https://github.com/cunningr/dna_workflows/tree/master

Example Framework



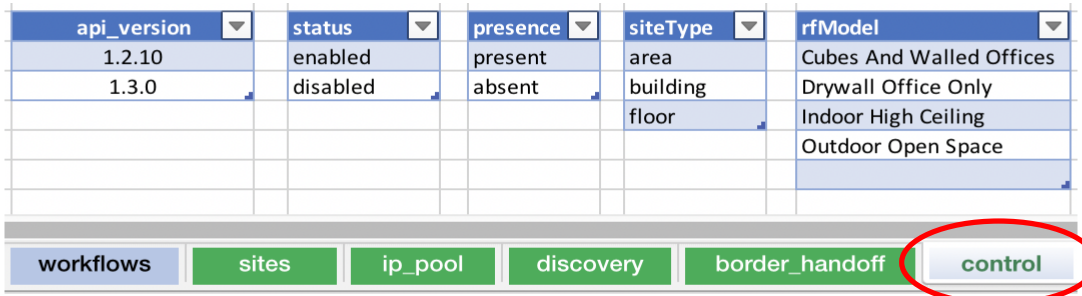
Workflow Database – Workflow Sheet

- Multi-sheet Excel workbook
- **Workflows inventory table**
  - DNAC access information
  - Workflow name
  - Status (enabled / disabled)
- **Control** sheet
  - Data validation lists



api_versio	username	password	base_url	verify
1.3.0	admin	CiscoLiveEUR	https://127.0.0.1:8888	FALSE

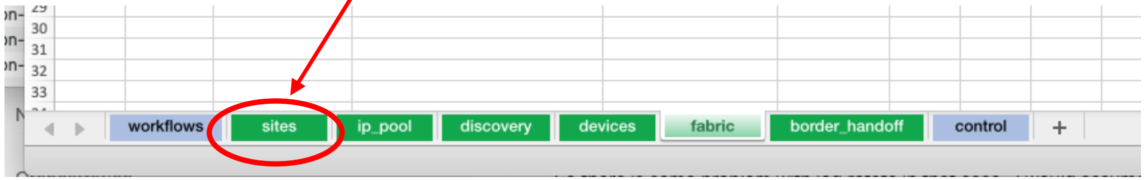
ID	Status	Name	Description
1	enabled	sites	Create, Delete of DNA Center site hierarchy
2	enabled	ip_pool	Create, Delete of DNA Center IP Pools
4	enabled	discovery	Update this workflow documentation
5	enabled	border_handoff	Update this workflow documentation



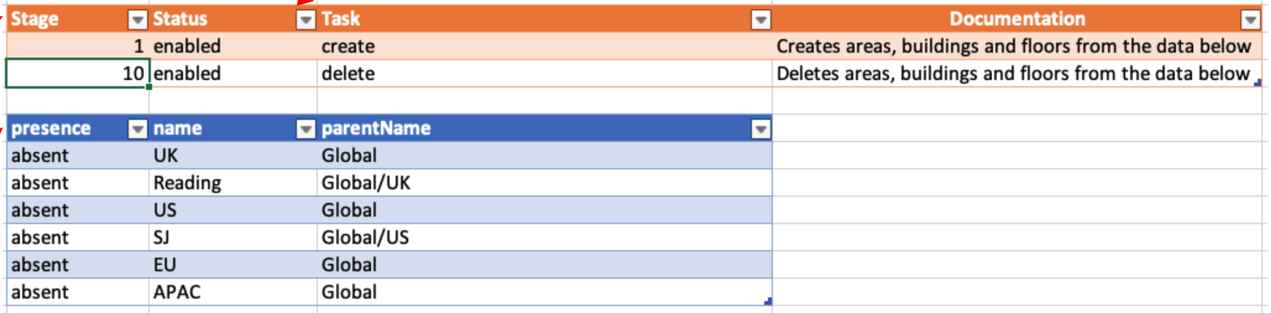
api_version	status	presence	siteType	rfModel
1.2.10	enabled	present	area	Cubes And Walled Offices
1.3.0	disabled	absent	building	Drywall Office Only
			floor	Indoor High Ceiling
				Outdoor Open Space

Workflow Database – Workflow Sheet

- Workflow 'sheet' => python module



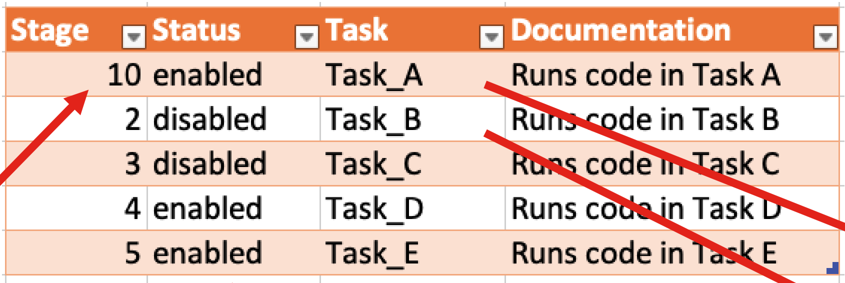
- Task => python function
- Control Table
- Input data tables



Stage	Status	Task	Documentation
1	enabled	create	Creates areas, buildings and floors from the data below
10	enabled	delete	Deletes areas, buildings and floors from the data below

presence	name	parentName
absent	UK	Global
absent	Reading	Global/UK
absent	US	Global
absent	SJ	Global/US
absent	EU	Global
absent	APAC	Global

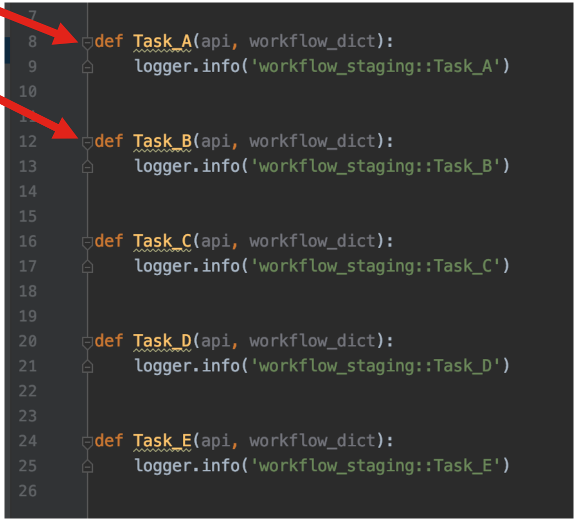
Workflow - Task Sequencing



Stage	Status	Task	Documentation
10	enabled	Task_A	Runs code in Task A
2	disabled	Task_B	Runs code in Task B
3	disabled	Task_C	Runs code in Task C
4	enabled	Task_D	Runs code in Task D
5	enabled	Task_E	Runs code in Task E

Order of execution

enable or disable a task



```
def Task_A(api, workflow_dict):
    logger.info('workflow_staging::Task_A')


def Task_B(api, workflow_dict):
    logger.info('workflow_staging::Task_B')


def Task_C(api, workflow_dict):
    logger.info('workflow_staging::Task_C')


def Task_D(api, workflow_dict):
    logger.info('workflow_staging::Task_D')


def Task_E(api, workflow_dict):
    logger.info('workflow_staging::Task_E')

```

Potential Use Cases

- Day 0 deployment tasks of DNA Center and SD-Access
- Digitization and central storage of operational run books
- Rollout of new SD-Access sites and fabrics
- Custom reports
- Labs setup/rebuild
- DNA Center testing

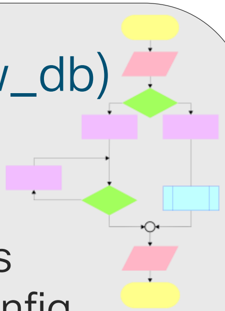
Workflow Demo

Example Workflow ‘sites’

main.py Execution Engine

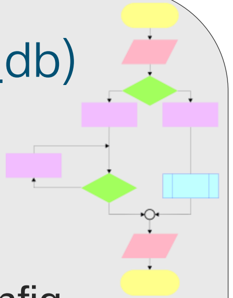
module sites

create(api, workflow_db)



```
1. GET existing sites
2. LOOP sites to add
    1. IF sites NOT exists
        1. POST site config
        2. WAIT for completion
        3. LOG job completion
3. LOOP buildings to add
    1. IF building NOT exists
    2. …
```

delete(api, workflow_db)



```
1. GET existing sites
2. LOOP buildings to delete
    1. IF buildings exists
        1. DELETE site config
        2. WAIT for completion
        3. LOG job completion
3. LOOP sites to delete
    1. IF sites exists
    2. …
```

Workflow Manager

Workflow Manager

- Run the script workflow_manager.py

```
Python3 workflow_manager.py [options]
```

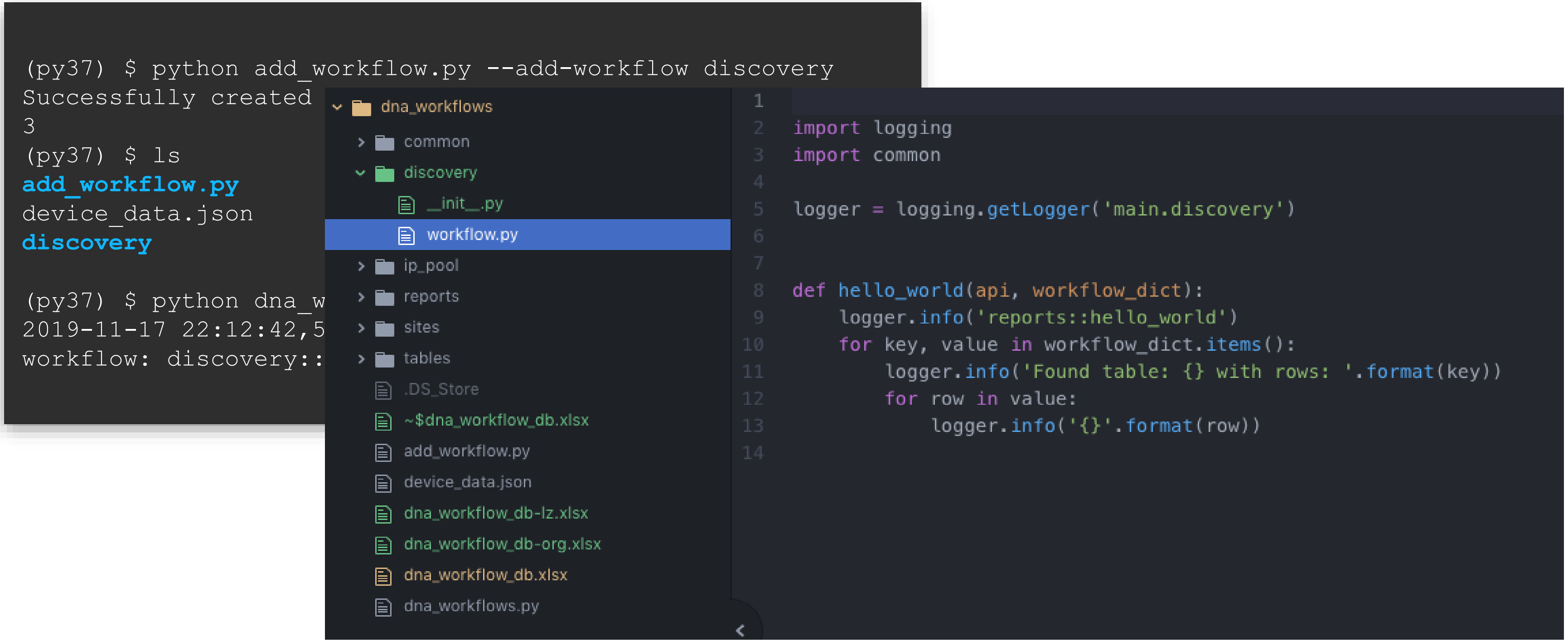
- Create a new workflow template *--add-workflow <workflow-name>*
  - A new row in the master workflows worksheet
  - A new worksheet for the added workflow with a new 'control' table
  - A new folder with basic 'Hello-World' workflow code

- Delete a workflow from the DB *--delete-workflow <workflow-name>*
- Delete a workflow from the DB and remove the python module

  --delete-workflow-and-clean <workflow-name>
- Export a workflow to a new workflow DB ready to run locally *--export-workflow*

Workflow Manager

- python add_workflow.py --add-workflow *discovery*



```
(py37) $ python add_workflow.py --add-workflow discovery
Successfully created
3
(py37) $ ls
add_workflow.py
device_data.json
discovery

(py37) $ python dna_w
2019-11-17 22:12:42,5
workflow: discovery::
```

```
dna_workflows
  common
  discovery
    __init__.py
    workflow.py
  ip_pool
  reports
  sites
  tables
  .DS_Store
  ~$dna_workflow_db.xlsx
  add_workflow.py
  device_data.json
  dna_workflow_db-lz.xlsx
  dna_workflow_db-org.xlsx
  dna_workflow_db.xlsx
  dna_workflows.py
```

```python

import logging
import common

logger = logging.getLogger('main.discovery')


def hello_world(api, workflow_dict):
    logger.info('reports::hello_world')
    for key, value in workflow_dict.items():
        logger.info('Found table: {} with rows: '.format(key))
        for row in value:
            logger.info('{}'.format(row))

```

Key Takeaways …

Key Takeaways

Create reusable code and reuse as much as possible

- Use existing modules or an SDK if possible
- Create ‘modules’ of code that can be reused
- Organise code for individual ‘tasks’ as functions

Understand the target workflow and the value in your overall use case.

- Automation doesn’t automatically make things ‘better’, it just speeds things up.

"Done is better than perfect”

- Just do it! Revise, learn and continually improve!

DEVNET Sessions on Cisco DNA Center

- DEVNET-1234: Cisco DNA Center 101: Getting started with Cisco DNA-C Platform
- DEVNET-2087: Intent APIs in DNA Center
- DEVNET-2225: Cisco DNA Center Platform – Real time notification via webhooks
- DEVNET-2275: Cisco DNA Center API Use Case – Automating SDA Deployment Tasks
- DEVNET-2425: Hands-on Scaling Common Cisco DNA Center tasks via API
- DEVNET-2877: Exploring Cisco DNA-C as a Platform
- DEVNET-3603: Explore the Programmability Options of Cisco DNA Center for Managing Network Intent
- DEVWKS-2878: Hands on with Cisco DNA Center Assurance APIs
- DEVWKS-2879: Hands on with Cisco DNA Center as a Platform

Complete your online session survey

- Please complete your session survey after each session. Your feedback is very important.
- Complete a minimum of 4 session surveys and the Overall Conference survey (starting on Thursday) to receive your Cisco Live t-shirt.
- All surveys can be taken in the Cisco Events Mobile App or by logging in to the Content Catalog on ciscolive.com/emea.

Cisco Live sessions will be available for viewing on demand after the event at ciscolive.com.

Continue your education

Demos in the Cisco Showcase

Walk-In Labs

Meet the Engineer 1:1 meetings

Related sessions

Thank you

You make **possible**