

The background is a vibrant, abstract graphic. It features a central bright white light source from which numerous colorful rays emanate, creating a sunburst or starburst effect. The rays transition through a spectrum of colors including yellow, orange, red, and various shades of blue and green. Overlaid on this are several large, semi-transparent, wavy shapes in similar color tones, giving the overall image a sense of motion and energy.

cisco *Live!*

Let's go

#CiscoLive



The bridge to possible

# Jenkins Automations for Cisco DNA Center

Gabi Zapodeanu,  
Technical Marketing Engineer  
@zapodeanu  
DEVNET-2151

**GitHub**

[github.com/cisco-en-programmability](https://github.com/cisco-en-programmability)



<https://YouTube.com/c/CiscoENProgrammability>

**CISCO** *Live!*

#CiscoLive

# Cisco Webex App

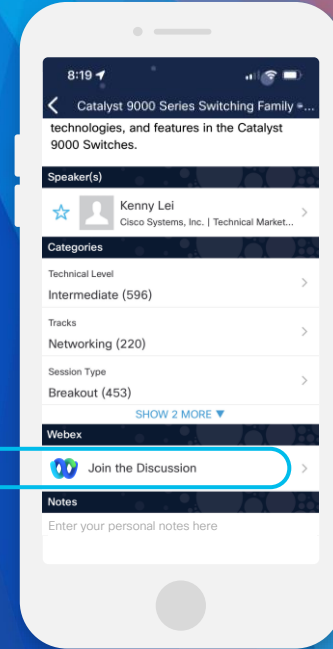
## Questions?

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

## How

- 1 Find this session in the Cisco Live Mobile App
- 2 Click “Join the Discussion”
- 3 Install the Webex App or go directly to the Webex space
- 4 Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until June 9, 2023.



<https://ciscolive.ciscoevents.com/ciscolivebot/#DEVNET-2151>

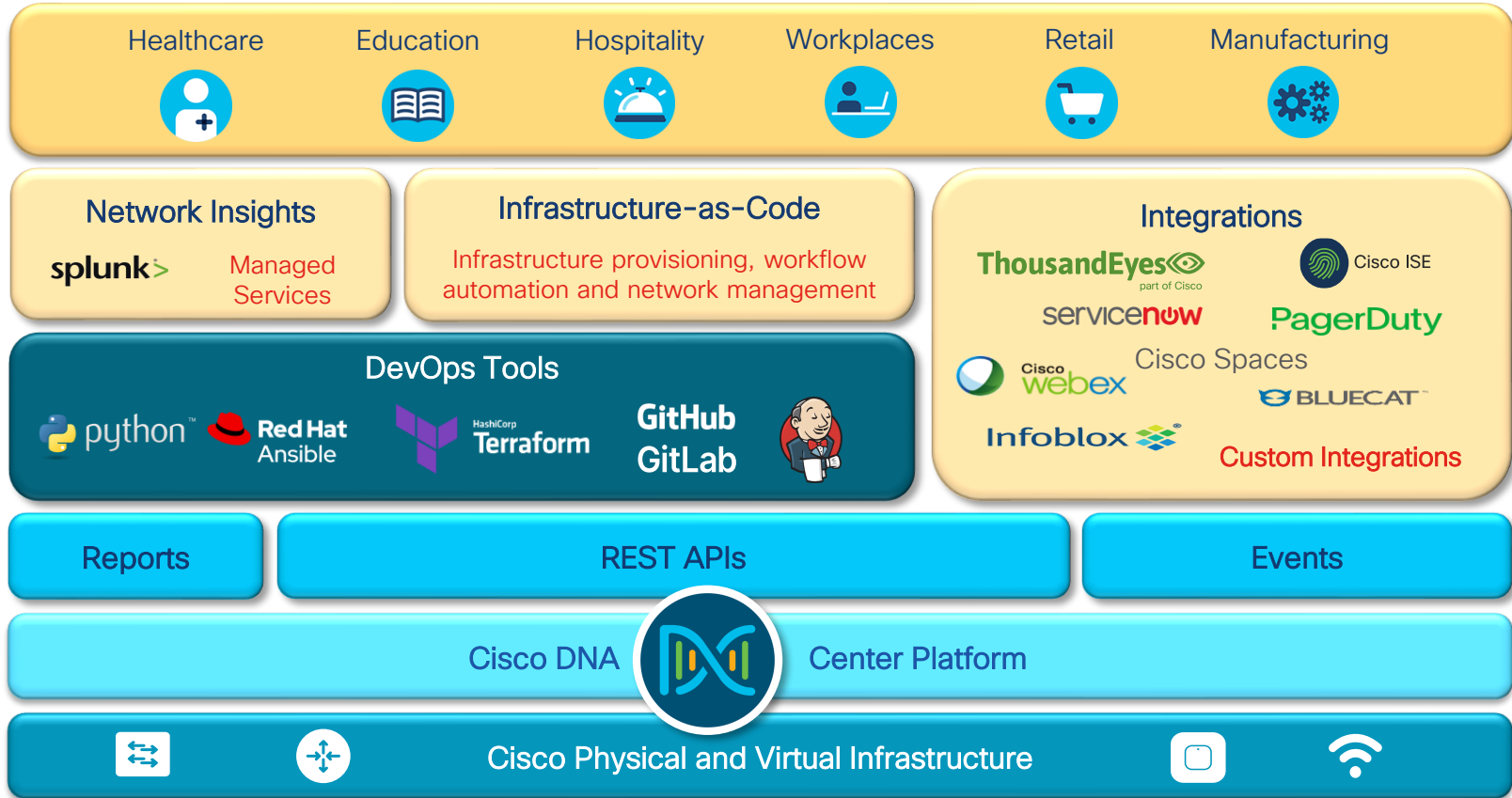
# Agenda

- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
- Developer Resources
- Summary

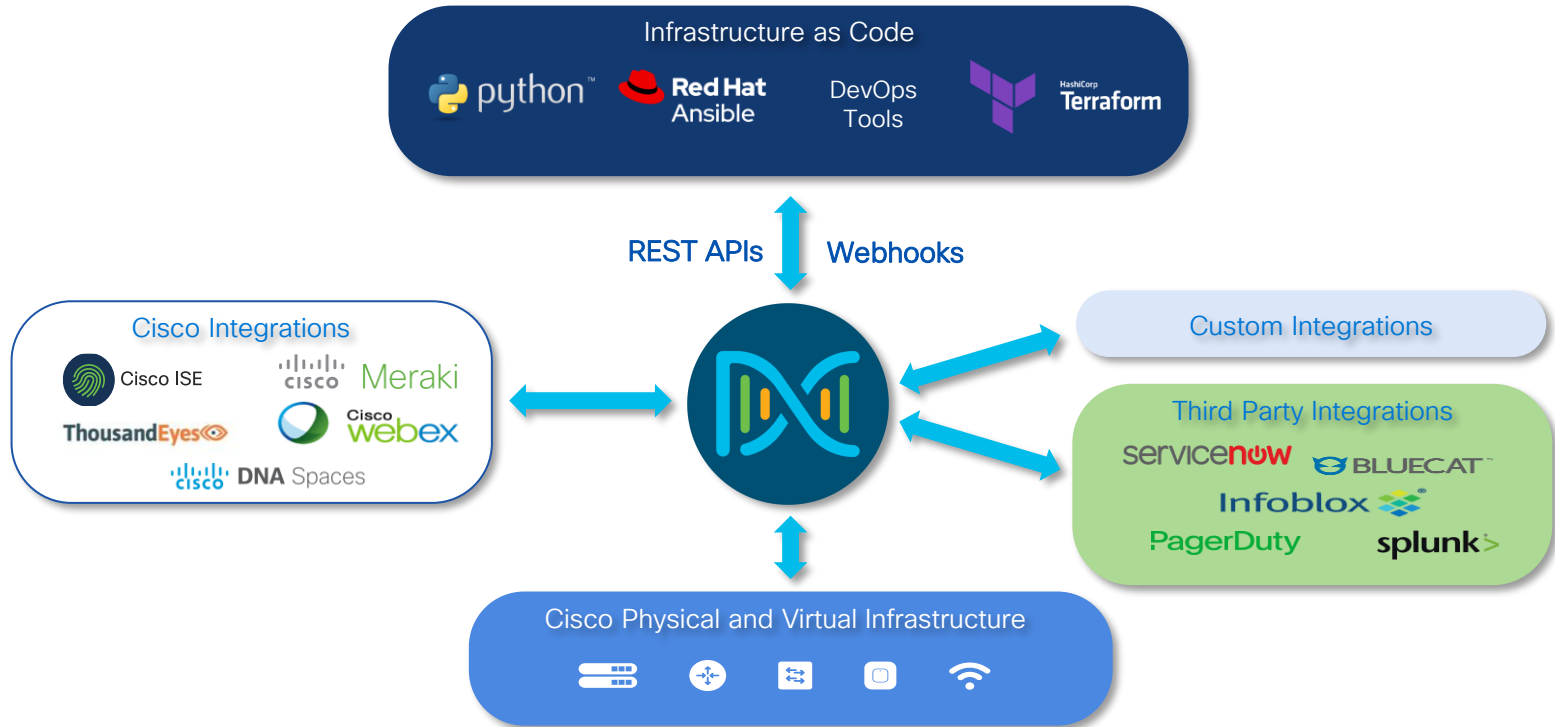
# Agenda

- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
- Developer Resources
- Summary

# Cisco DNA Center Platform



# Cisco DNA Center Platform



# Cisco DNA Center Platform Overview

## Event Notifications

- Assurance Issues
- AI/ML Insights
- System Health
- Integration Connectivity
- License Management
- Webhooks
- PagerDuty
- Webex
- Syslog
- SNMP

## Northbound REST APIs

- Network Inventory
- Network Topology
- Network Design
- Provisioning
- SWIM, PnP
- Path Trace
- Assurance
- SDA
- Templates
- RMA
- Config Archive
- Sensors

## IT Ecosystem Integrations

- IT Service Management
- IP Address Management
- Reporting
- Wireless Planning
- Incident Response
- SIEM, Splunk

## Developer Resources

- Sample Code, Videos
- Python SDK, Ansible, Terraform
- Cisco DevNet
  - Sandboxes, Learning Labs
  - Developer Guides
  - Sample Code



# DNA Center Platform – Overview

Platform / Overview

Welcome to the Cisco DNA Center Platform. Programmatically access your network through Intent APIs, integrate with your preferred IT systems to create end-to-end solutions and add support for multi-vendor devices.

### Bundles

Bundles are easy to use feature sets for consuming Intent APIs, integrations, events and notifications. View all the available bundles, enable relevant bundles and customize the configuration preferences to consume events as per your application(s) or IT system(s) needs.

### Developer Toolkit

Discover APIs to manage your network, configure integration flows and access network data to analyze, export and visualize complex reports.

### Runtime Dashboard

Get insights into API usage, view events published to IT systems such as number of API calls, response time(s), events published, bundles activated etc.

### Configurations

View and set global or bundle specific settings to manage your integration configurations and modify event specific settings.

## Notifications

Find

**BUNDLE UPDATE**  
Basic ITSM (ServiceNow) CMDB synchronization successfully configured.  
Apr 24 2023, 05:26 pm  
[View Details](#) | [Dismiss](#)

**BUNDLE UPDATE**  
Network Issue Monitor and Enrichment for ITSM (ServiceNow) successfully configured.  
Feb 01 2023, 04:17 pm  
[View Details](#) | [Dismiss](#)

**BUNDLE UPDATE**  
Network Issue Monitor and Enrichment for ITSM (ServiceNow) successfully configured.  
Feb 01 2023, 04:04 pm

# DNA Center Platform – Developer Toolkit

APIs

Integration Flows

Event Notifications

ITSM Integration

Know Your Network

Applications

Clients

Compliance

Devices

Issues

Security Advisories

Sensors

Sites

Topology

Users

Operational Tasks

Command Runner

Discovery

File

Platform / Developer Toolkit

Review API docs, configure integration workflows and subscribe to events

Search

Search API

|        |                                                      |                                                                                                                                                                   |                                                                           |     |
|--------|------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------|-----|
| DELETE | Delete Device by Id                                  | Deletes the network device for the given Id                                                                                                                       | /network-device/\${id}                                                    | ... |
| GET    | Get Module Info by Id                                | Returns Module info by 'module id'                                                                                                                                | /network-device/module/\${id}                                             | ... |
| GET    | Return PowerSupply/ Fan details for the Given device | Return PowerSupply/ Fan details for the Given device                                                                                                              | /network-device/\${deviceUuid}/equipment                                  | ... |
| GET    | Returns POE interface details for the device.        | Returns POE interface details for the device, where deviceuuid is mandatory & accepts comma separated interface names which is optional and returns...            | /network-device/\${deviceUuid}/interface/poe-detail                       | ... |
| POST   | Clear Mac-Address table                              | Clear mac-address on an individual port. In request body, operation needs to be specified as 'ClearMacAddress'. In the future more possible operations will be... | /interface/\${interfaceUuid}/operation                                    | ... |
| GET    | Get Device list                                      | Returns list of network devices based on filter criteria such as management IP address, mac address, hostname, etc. You can use the .* in any value to conduct... | /network-device                                                           | ... |
| GET    | Get Device Interfaces by specified range             | Returns the list of interfaces for the device for the specified range                                                                                             | /interface/network-device/\${deviceid}/\${startIndex}/\${recordsToReturn} | ... |

Try

Call an API without writing any code

# Try an API Call

**Cisco DNA Center** Get Device list

## Try "Get Device list"

Method: **GET** Public URL : <https://10.93.141.45/dna/intent/api/v1/network-device>

**PARAMETERS**

QUERY PARAMETERS ☐ Select All

- ☒ hostname
- ☐ managementIpAddress
- ☐ macAddress
- ☐ locationName

**Response** Headers Status Code: 200

```
1 {
2   "response": [
3     {
4       "type": "Cisco Catalyst 9300 Switch",
5       "description": "Cisco IOS Software [Cupertino], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 17.9.1, RELEASE SOFTWARE (fc8) T
6       "family": "Switches and Hubs",
7       "memorySize": "NA",
8       "lastUpdateTime": "1684454228783",
9       "macAddress": "30:8b:b2:ba:c2:80",
10      "softwareType": "IOS-XE",
11      "softwareVersion": "17.9.1",
12      "deviceSupportLevel": "Supported",
13      "serialNumber": "FJC2325T0PH",
14      "collectionInterval": "Global Default",
15      "inventoryStatusDetail": "<status><general code=\\\"SUCCESS\\\"/></status>",
16      "managementState": "Managed",
17      "upTime": "77 days, 4:46:16.98",
18      "roleSource": "MANUAL",
19      "associatedWlcIp": "",
20      "apEthernetMacAddress": null,
21      "lastUpdated": "2023-05-18 23:57:08",
22      "bootDateTime": "2023-03-02 19:11:08",
23      "series": "Cisco Catalyst 9300 Series Switches",
24      "snmpContact": "",
25      "snmpLocation": "",
26      "errorCode": null,
```

# Code Preview

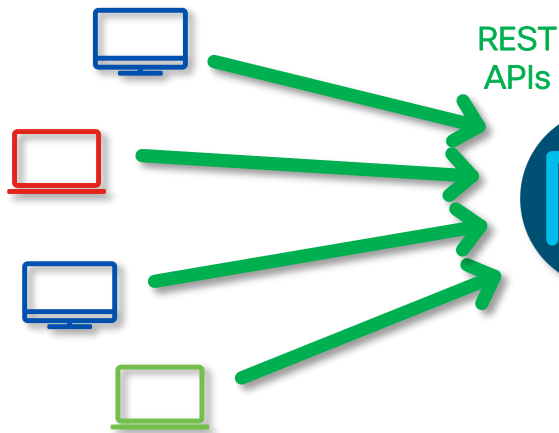
The screenshot displays the Cisco DNA Center API interface. On the left is a sidebar with the 'Cisco DNA Center' logo and a list of navigation items: APIs, Integration Flows, Events, Search, Know Your Network, Applications, Clients, Compliance, Devices, Issues, Security Advisories, Sensors, Sites, Topology, Users, Operational Tasks, Command Runner, Discovery, File, and Path Trace. The main panel is titled 'Get Device Count' and shows a GET request to the URL `https://10.93.141.45/dna/intent/api/v1/network-device/count`. Below the URL, it states: 'Returns the count of network devices based on the filter criteria by management IP address, mac address, hostname and location name'. A link to the 'Cisco DevNet API Guide' is provided. There are three tabs: 'Features', 'Responses', and 'Code Preview', with the latter being selected and highlighted by a green box. A language selection dropdown is also highlighted with a green box, showing a search bar and a list of languages: Node - Request, Python (selected), Ruby, Javascript, JQuery, and PHP. A dark blue callout box on the right contains the text: 'Code Preview' in orange, followed by 'Generate code in few programming languages' in white. At the bottom right of the main panel are 'Close' and 'Try' buttons.

**Code Preview**  
Generate code in few programming languages

# Agenda

- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
- Developer Resources
- Summary

# Cisco DNA Center DevOps Tools Journey

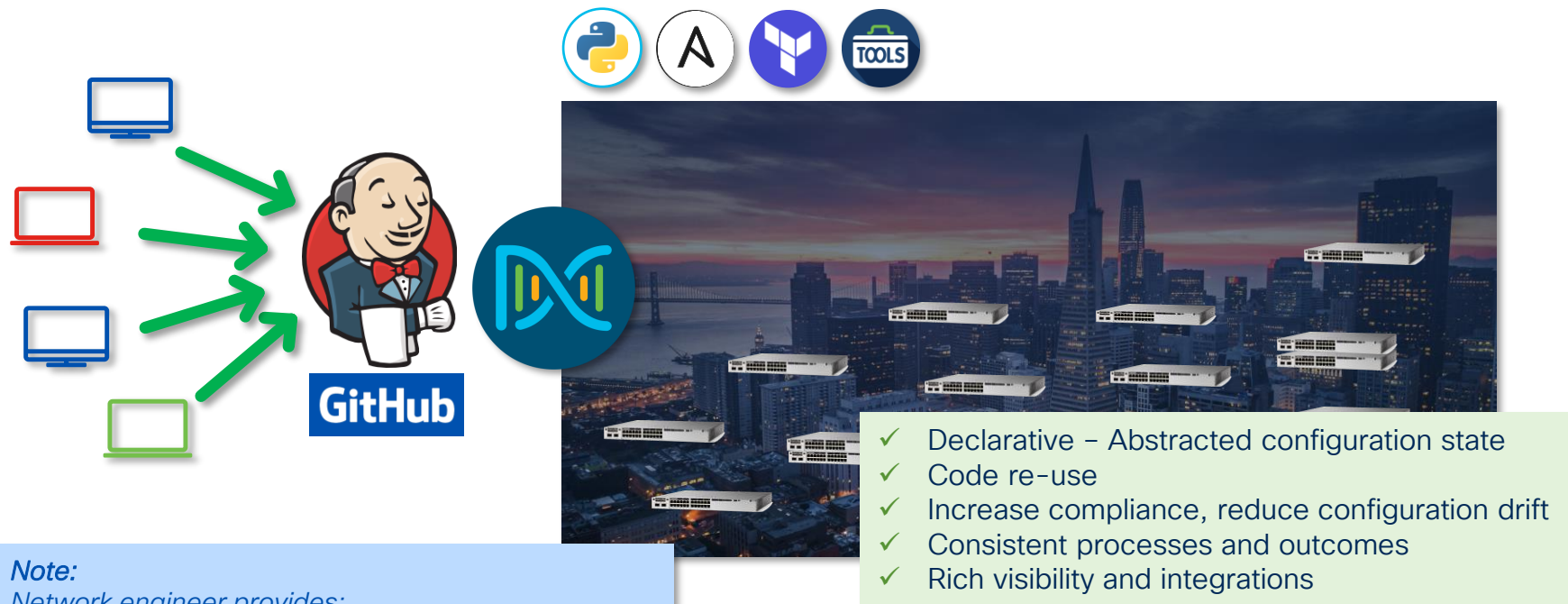


**Note:**  
Network engineer provides:  
Configuration details and application files



- X Various home-grown scripts/apps to manage
- X Complexity and lack of consistency
- X Limited visibility and reporting
- X Security concerns

# Cisco DNA Center Jenkins Integration

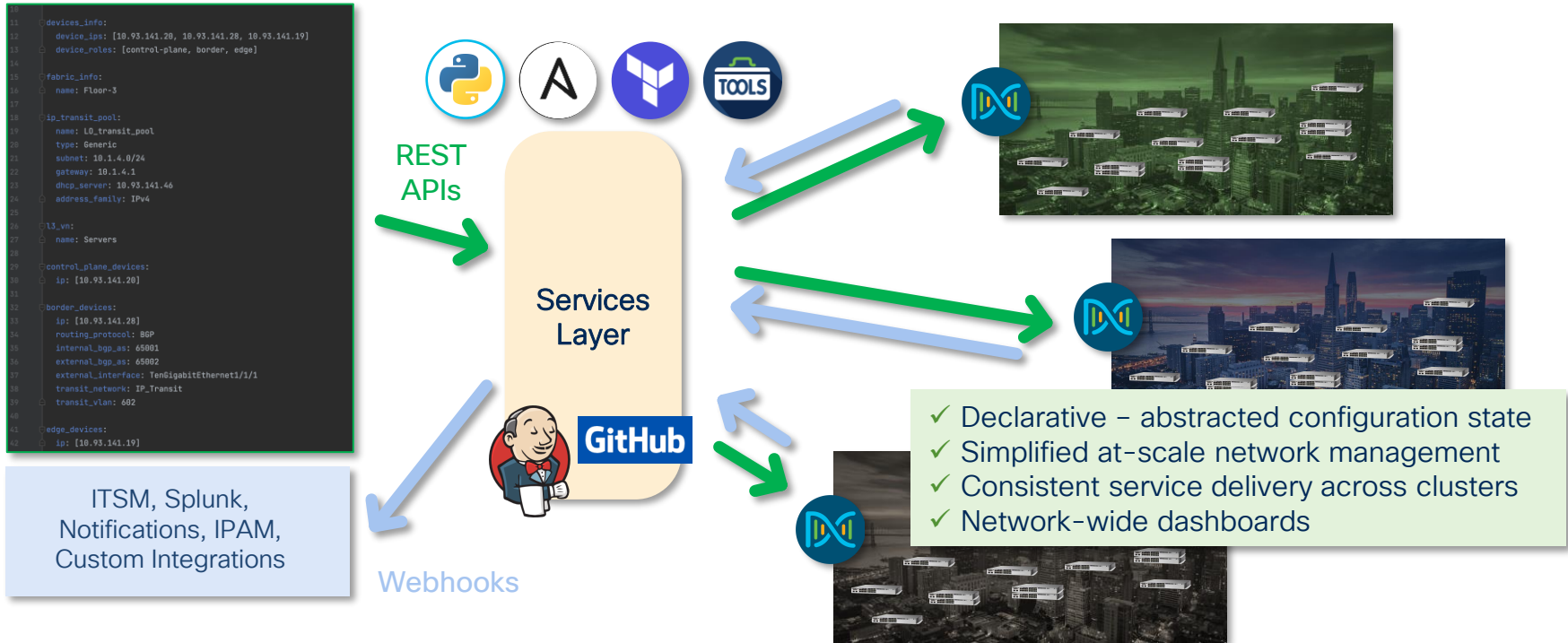


## Note:

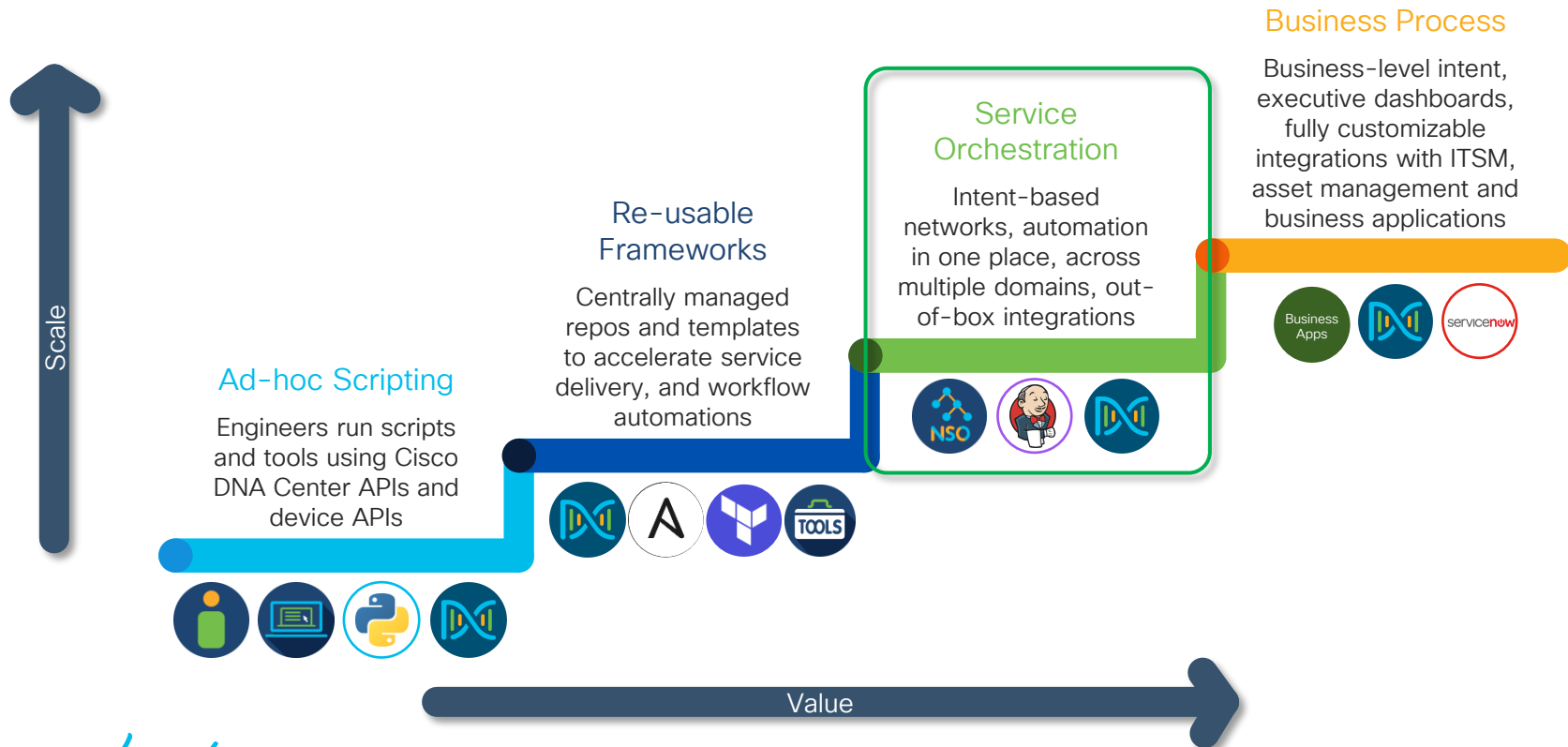
Network engineer provides:  
Intent to be used by predefined automation workflows

Jenkins docs: <https://jenkins.io>

# Multi Cisco DNA Center Managed Networks



# Cisco DNA Center and DevOps Journey



# GitOps for Cisco DNA Center?

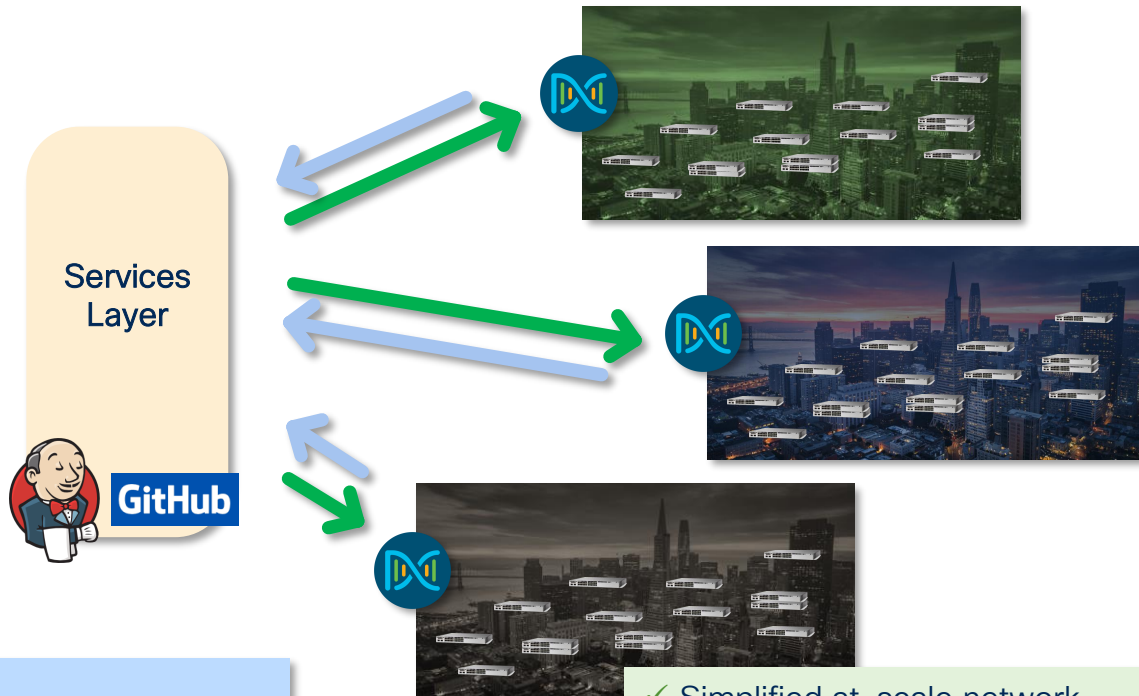
```
project_name: Ansible_Project
template_name: csr_ntp_snmp
filter: device_family
filter_value: Cisco Cloud Services Router 1000V
cli_template: |
!
no snmp-server host 10.93.135.30 version 2c R0
no snmp-server host 10.93.130.50 version 2c RW
!
ntp server 171.68.48.78
no ntp server 171.68.48.80
```

```
project_name: Ansible_Project
template_name: access_switches_logging
filter: role
filter_value: ACCESS
cli_template: |
!
logging host 10.93.141.37
logging host 10.93.141.37 transport udp port 8514
!
service timestamps debug datetime localtime
service timestamps log datetime localtime
```

- GitOps uses Git repositories as a single source of truth to deliver Infrastructure-as-Code:
  - Automations and integrations apps
  - Intent-based configurations documented as code
  - Inventories: clients, devices, sites, fabrics
- Code changes, intent configurations, and network state are tracked, and version controlled
- Consistency across Cisco DNA Center clusters and IT teams
- Enhanced security and visibility, robust role-based access

# Git Configurations + Inventories

- CLI templates and profiles
- Network settings
- SDA-as-Code configurations
- Custom events notifications and subscriptions
- Reports and inventory
- Compliance

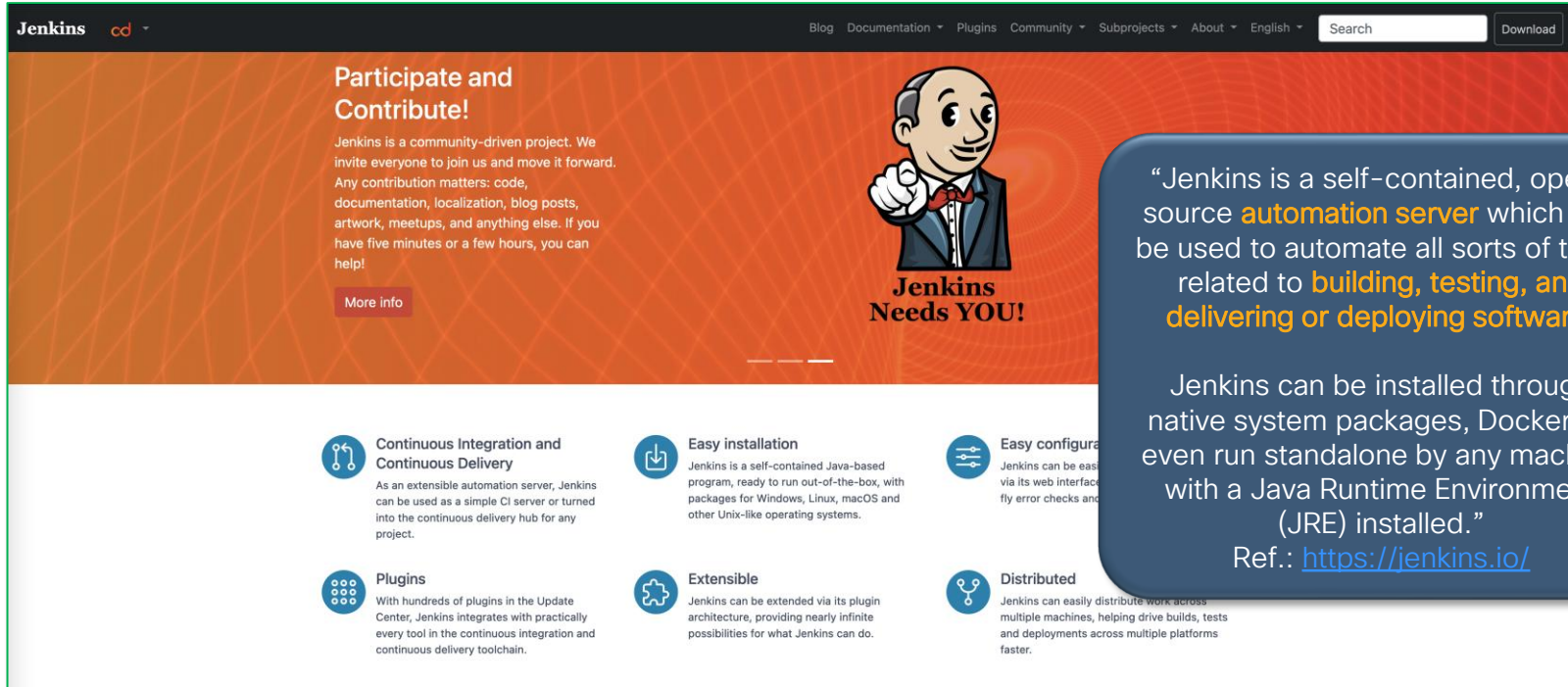


## Notes:

*Configurations could be CLI templates, profiles, network settings or YAML/JSON intent configurations - SDA  
Enables time travel, changes, reporting*

- ✓ Simplified at-scale network management
- ✓ Consistent and efficient service delivery across clusters

# What is Jenkins?



The screenshot shows the Jenkins homepage. At the top is a navigation bar with links for Blog, Documentation, Plugins, Community, Subprojects, About, and English, along with a search bar and a Download button. The main header features the text "Participate and Contribute!" and a cartoon character of a man in a suit pointing, with the text "Jenkins Needs YOU!" below it. A callout box on the right contains a quote about Jenkins being a self-contained, open-source automation server and a reference link. Below the header, there are six feature cards: Continuous Integration and Continuous Delivery, Easy installation, Easy configuration, Plugins, Extensible, and Distributed.

**Participate and Contribute!**

Jenkins is a community-driven project. We invite everyone to join us and move it forward. Any contribution matters: code, documentation, localization, blog posts, artwork, meetups, and anything else. If you have five minutes or a few hours, you can help!

[More info](#)

**Jenkins Needs YOU!**

**Continuous Integration and Continuous Delivery**

As an extensible automation server, Jenkins can be used as a simple CI server or turned into the continuous delivery hub for any project.

**Easy installation**

Jenkins is a self-contained Java-based program, ready to run out-of-the-box, with packages for Windows, Linux, macOS and other Unix-like operating systems.

**Easy configuration**

Jenkins can be easily configured via its web interface, with built-in error checks and feedback.

**Plugins**

With hundreds of plugins in the Update Center, Jenkins integrates with practically every tool in the continuous integration and continuous delivery toolchain.

**Extensible**

Jenkins can be extended via its plugin architecture, providing nearly infinite possibilities for what Jenkins can do.

**Distributed**

Jenkins can easily distribute work across multiple machines, helping drive builds, tests and deployments across multiple platforms faster.

“Jenkins is a self-contained, open-source **automation server** which can be used to automate all sorts of tasks related to **building, testing, and delivering or deploying software**.  
Jenkins can be installed through native system packages, Docker, or even run standalone by any machine with a Java Runtime Environment (JRE) installed.”  
Ref.: <https://jenkins.io/>

# Why Cisco DNA Center and Jenkins?



+



Jenkins is a self-contained, open-source **automation server** which may be used to:

- Support network infrastructure management as Infrastructure-as-Code
- Integrate with GitHub for validated and approved apps
- Integrate with GitHub for intent configurations, profiles and policies
- Be the platform used by all network engineers to run apps and scripts

The screenshot shows the Jenkins User Documentation page. The left sidebar contains a navigation menu with sections: > User Documentation Home, User Handbook (with links to Overview, Installing, Using, Pipeline, Blue Ocean, Managing, Securing, System Administration, Scaling, Troubleshooting, and Glossary), Tutorials (with links to Guided Tour, Pipeline, and Build Tools), and Resources (with links to Pipeline Syntax reference, Pipeline Steps reference, and LTS Upgrade guides). The main content area has a header 'Jenkins User Documentation' and a welcome message. It includes sections for 'What is Jenkins?' and 'About this documentation'. A blue callout box on the right lists key topics covered in the documentation.

**Jenkins**  [cd](#) [Blog](#) [Success Stories](#) [Documentation](#) [Plugins](#) [Community](#) [Subprojects](#) [Security](#) [About](#) [Download](#)

[> User Documentation Home](#)

**User Handbook**

- [User Handbook Overview](#)
- [Installing Jenkins](#)
- [Using Jenkins](#)
- [Pipeline](#)
- [Blue Ocean](#)
- [Managing Jenkins](#)
- [Securing Jenkins](#)
- [System Administration](#)
- [Scaling Jenkins](#)
- [Troubleshooting Jenkins](#)
- [Glossary](#)

**Tutorials**

- [Guided Tour](#)
- [Jenkins Pipeline](#)
- [Using Build Tools](#)

**Resources**

- [Pipeline Syntax reference](#)
- [Pipeline Steps reference](#)
- [LTS Upgrade guides](#)

## Jenkins User Documentation

Welcome to the Jenkins user documentation - for people wanting to use Jenkins's existing functionality and plugin features.

If you want to extend the functionality of Jenkins by developing plugins, see the [Jenkins](#) (developer documentation).

### What is Jenkins?

Jenkins is a self-contained, open source automation server that automates the process of building, testing, and delivering or deploying software.

Jenkins can be installed through native system packages or by manually installing the Jenkins Runtime Environment (JRE) installed.

### About this documentation

This documentation begins with a [Guided Tour](#) to help you get up and running with Jenkins and introduce you to Jenkins's main feature, Pipeline.

There are also [tutorials](#) geared to developers who want to orchestrate and automate building their project in Jenkins using Pipeline and Blue Ocean.

If you've never used Jenkins before or have limited Jenkins experience, then the Guided Tour and introductory tutorials are good places to start.

If you are looking for more detailed information about using Jenkins, please refer to the [User Handbook](#).

Includes everything you need to know:

- How to install, upgrade, configure
- How to get started
- Jenkins management, credentials and plugins
- Pipeline syntax and tutorials
- Security features

# Jenkins Dashboard

- Jenkins preferences
- Group pipelines in views
- Build queue and execution status
- Pipeline stats – health, number of runs, success and failure, ...

The screenshot shows the Jenkins Dashboard interface. The top navigation bar includes the Jenkins logo and a search bar. The left sidebar contains navigation links: New Item, People, Build History, Project Relationship, Check File Fingerprint, Manage Jenkins (highlighted with an orange box), My Views, and Open Blue Ocean. The main content area features a tabbed interface with 'All', 'LabView', and 'OpsView' tabs. Below the tabs is a table of pipeline runs. The 'Build Queue' and 'Build Executor Status' sections are visible on the left. The 'Build Queue' section shows 'No builds in the queue.' The 'Build Executor Status' section shows a list of executors, including 'Device Inventory' which is highlighted with a blue box.

| S   | W | Name ↓                   | Last Success    | Last Failure     | Last Duration | Fav |
|-----|---|--------------------------|-----------------|------------------|---------------|-----|
| ✓   | ☁ | CLI Templates Deployment | 1 mo 2 days #14 | 1 mo 2 days #13  | 2 min 30 sec  | ▶ ☆ |
| ✗   | ☁ | Dev Ready                | 1 mo 11 days #2 | 1 mo 11 days #18 | 1 min 33 sec  | ▶ ☆ |
| ✓   | ☀ | Device Inventory         | 1 hr 18 min #6  | N/A              | 41 sec        | ▶ ☆ |
| ✓   | ☀ | GitHub Sync              | 18 hr #67       | N/A              | 27 sec        | ▶ ☆ |
| ✓   | ☀ | GitLab Sync              | 19 hr #46       | N/A              | 23 sec        | ▶ ☆ |
| ✓   | ☀ | Jenkins Sandbox          | 1 mo 11 days #8 | 1 mo 12 days #3  | 3.4 sec       | ▶ ☆ |
| ... | ☀ | Jenkinsfile Template     | N/A             | N/A              | N/A           | ▶ ☆ |
| ✓   | ☀ | PoE Sustainability       | 1 mo 4 days #56 | N/A              | 16 sec        | ▶ ☆ |
| ✓   | ☁ | SDA-as-Code              | 12 days #5      | 12 days #2       | 1 min 19 sec  | ▶ ☆ |

**Build Queue** ▼  
No builds in the queue.

**Build Executor Status** ▼  
1 Idle  
2 Idle  
3  
#2 (Install Python libraries)

# Jenkins Dashboard - Views

The screenshot shows the Jenkins Dashboard with the 'OpsView' view selected. The left sidebar contains a list of actions: New Item, People, Build History, Edit View, Delete View, Project Relationship, Check File Fingerprint, Manage Jenkins, My Views, and Open Blue Ocean. The main content area displays a table of Jenkins jobs. A green box highlights the 'All', 'LabView', and 'OpsView' tabs, with 'OpsView' being the active tab. The table lists jobs with their status, names, last success/failure times, and durations.

| S | W | Name ↓                   | Last Success   | Last Failure | Last Duration | Fav |
|---|---|--------------------------|----------------|--------------|---------------|-----|
| ✓ | ☁ | CLI Templates Deployment | 20 days #14    | 20 days #13  | 2 min 30 sec  | ▶ ☆ |
| ✓ | ☀ | Device Inventory         | 24 days #4     | N/A          |               | ▶ ☆ |
| ✓ | ☀ | GitHub Sync              | 18 hr #54      | N/A          |               | ▶ ☆ |
| ✓ | ☀ | GitLab Sync              | 18 hr #34      | N/A          |               | ▶ ☆ |
| ✓ | ☀ | PoE Sustainability       | 22 days #56    | N/A          |               | ▶ ☆ |
| ✓ | ☁ | SDA-as-Code              | 3 min 0 sec #5 | 25 days #4   |               | ▶ ☆ |

Manage your Jenkins projects/pipelines using views

The screenshot shows the Jenkins Dashboard with the 'LabView' view selected. The left sidebar is identical to the previous screenshot. The main content area displays a table of Jenkins jobs. A green box highlights the 'All', 'LabView', and 'OpsView' tabs, with 'LabView' being the active tab. The table lists jobs with their status, names, last success/failure times, and durations. At the bottom, there is a section for 'Icon legend' and 'Atom feed' links.

| S   | W | Name ↓                           | Last Success  | Last Failure | Last Duration | Fav |
|-----|---|----------------------------------|---------------|--------------|---------------|-----|
| ✓   | ☀ | Cisco DNA Center Jenkins Sandbox | 19 days #1    | N/A          | 3.8 sec       | ▶ ☆ |
| ✗   | ☁ | Dev Ready                        | 2 mo 1 day #2 | 19 days #19  | 1 min 33 sec  | ▶ ☆ |
| ✓   | ☀ | Jenkins Sandbox                  | 2 mo 1 day #8 | N/A          | 3.4 sec       | ▶ ☆ |
| ... | ☀ | Jenkinsfile Template             | N/A           | N/A          | N/A           | ▶ ☆ |

Icon: S M L      Icon legend      Atom feed for all      Atom feed for failures      Atom feed for just latest builds

# Agenda

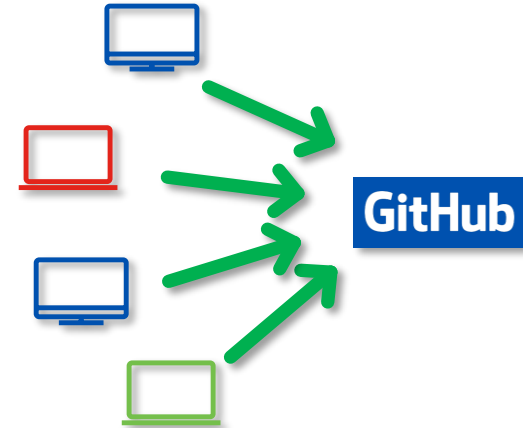
- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
  - How to Build a Pipeline
  - Device Inventory
  - GitHub Integration App
  - CLI Templates Deployment
  - SDA-as-Code
  - PoE Sustainability
- Developer Resources
- Summary

# Running an Automation Workflow

**Note:**

*We do this every time we start development or run the app*

- Create and activate a new Python virtual environment, or a Docker container
- Clone, or pull, code from Version Control Platforms (GitHub or GitLab)
- Install or upgrade Python Libraries, example: Cisco DNA Center SDK
- Provide the required params
- Run the application



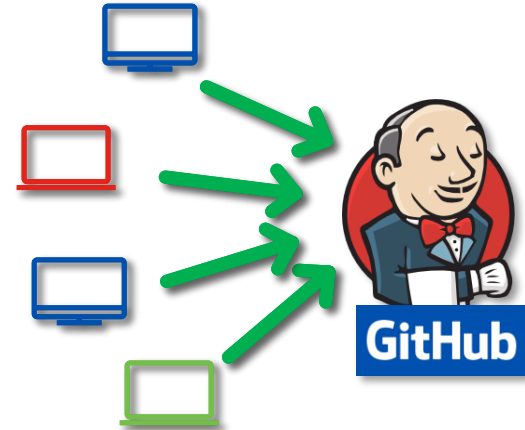
Runbook environments drift between developers, using different libraries and code versions

# Jenkins Pipeline Build

- Create a Docker container
- Clone, or pull, code from Version Control Platforms, GitHub or GitLab
- Install or upgrade Python Libraries - Cisco DNA Center SDK
- Provide the required params, or pull them from Version Control
- Run the application

*Note:*

*All steps are completely automated*



Runbook environments identical, always  
using the latest code versions and libraries  
Rich scheduling/triggering options

# Create a Basic Pipeline

The image shows the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and the user name 'Gabriel Zapodeanu' with a 'log out' button. The main content area is divided into a left sidebar with navigation links (People, Build History, Edit View, Delete View, Project Relationship, Check File Fingerprint, Manage Jenkins, My Views, Open Blue Ocean) and a main panel. The main panel shows a table of items with columns for status (S), warning (W), name, and a link. The 'New Item' button is highlighted in the sidebar. The 'Enter an item name' form is shown, with the 'Pipeline' option selected. The 'Copy from' field is highlighted, showing 'Jenkins Sandbox' as the selected item. A blue callout box points to the 'Pipeline' option with the text 'Name and type of pipeline'. A light blue callout box points to the 'Copy from' field with the text 'Note: Optional, copy from existing pipeline'.

**Jenkins**

Dashboard > LabView >

+ New Item

People

Build History

Edit View

Delete View

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Open Blue Ocean

All LabView OpsView +

| S   | W | Name ↓               |   |
|-----|---|----------------------|---|
| ✗   | ☁ | Dev Ready            | 1 |
| ✓   | ☀ | Jenkins Sandbox      | 1 |
| ... | ☀ | Jenkinsfile Template | N |

Icon: S M L

**Enter an item name**

Cisco DNA Center Jenkins Sandbox

\* Required field

**Freestyle project**

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**Pipeline**

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Multi-configuration project**

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Folder**

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**

Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**

Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Copy from

Jenkins Sandbox

OK Add to current view

**Note:**  
Optional, copy from existing pipeline

**Name and type of pipeline**

# Create a Basic Pipeline

Pipeline Script:  
1. Selecting the Agent –  
Docker image Python latest

## Notes:

1. Jenkinsfile is a text file representing the definition of a pipeline. It includes multiple stages and steps.
2. Steps may be constructed using:
  - Shell commands
  - Run code developed using the Python SDK, Ansible or Terraform
  - Jenkins Plugins

Dashboard > LabView > Cisco DNA Center Jenkins Sandbox > Configuration

Definition

Pipeline script

Script ?

```
1 pipeline {
2   agent {
3     {
4       docker
5       {
6         image 'python:latest'
7       }
8     }
9   }
10  options {
11    skipStagesAfterUnstable()
12    timeout(time: 15, unit: 'MINUTES')
13  }
14  stages {
15    stage ('Build Python Environment') {
16      steps {
17        echo('\n\nBuilding the Docker container:.....')
18      }
19    }
20  }
21  post {
22    always {
23      echo '\n\nCisco DNA Center Jenkins Sandbox end'
24    }
25  }
26 }
27
```

try sample Pipeline...

Select a Docker image

Configure Stages and Steps

Define the post activities

☒ Use Groovy Sandbox ?

Save Apply

# Build the Basic Pipeline

The image displays three overlapping screenshots of the Jenkins web interface, illustrating the process of building and viewing a pipeline.

- Top Left Screenshot:** Shows the Jenkins dashboard for the "Cisco DNA Center Jenkins Sandbox" pipeline. The "Build Now" button is highlighted with a green circle.
- Top Right Screenshot:** Shows the "Console Output" view for the pipeline. The "Console Output" link in the left sidebar is highlighted with a green circle. The console output shows the pipeline execution details, including the user "Gabriel Zapodeanu" and the command "docker inspect -f, python:latest".
- Bottom Screenshot:** Shows the "Stage View" for the pipeline. The "Stage View" link in the left sidebar is highlighted with a green circle. The stage view displays the pipeline stages and their execution times. A table shows the average stage times and full run time.

| Average stage times:      |      |
|---------------------------|------|
| Build Python Environment  | 73ms |
| Declarative: Post Actions | 57ms |

Average full run time: ~3s

Permalink: #1 May 2, 2023, 8:07 PM

Atom feed for all Atom feed for failures

**Note:**  
The console view is very valuable to view all details of the pipeline execution, specially when troubleshooting

# Pipeline Configuration – Adding Stages/Steps

Dashboard > GitHub Sync > Configuration

## Configure

- General
- Advanced Project Options
- Pipeline**

Script ?

```
1 pipeline {
2   agent
3     { docker
4       { {}
5     }
6   }
7   environment {}
8   options {}
9   stages {
10     stage ('Environment variables') {
11       steps {}
12     }
13     stage ('Build Python Environment') {
14       steps {}
15     }
16     stage ('Pull or Clone the source code') {
17       steps {}
18     }
19     stage ('Install Python libraries') {
20       steps {}
21     }
22     stage ('Execute Python application') {
23       steps {}
24     }
25   }
26   post {
27     always {}
28   }
29 }
```

### Jenkinsfile Declarative Pipeline

We need to include all tasks to build the environment running the automation workflow:

- Pull or clone code
- Install libraries
- Run the app

# Clone or Pull Code from GitHub

## Configure your GitHub Personal Access Token

The image shows the Jenkins web interface. The 'Manage Jenkins' page is visible, with the 'Manage Credentials' option highlighted in the 'Security' section. A modal window titled 'Global credentials (unrestricted)' is open, showing a table of credentials. The table has columns for ID, Name, Kind, and Description. Two credentials are listed: 'GITHUB\_TOKEN' and 'GITLAB\_TOKEN', both of kind 'Secret text'. A green box highlights the 'GITHUB\_TOKEN' entry. In the background, the 'Pipeline' definition is visible, showing a script that uses the 'GITHUB\_TOKEN' credential.

| ID           | Name         | Kind        | Description  |
|--------------|--------------|-------------|--------------|
| GITHUB_TOKEN | GitHub token | Secret text | GitHub token |
| GITLAB_TOKEN | GitLab token | Secret text | GitLab token |

```
10 environment {
11   REPO_NAME = 'dnocenter_github_integration'
12   GITHUB_TOKEN = credentials('GITHUB_TOKEN')
13   PIPELINE_NAME = 'GITHUB_SYNC'
14   PYTHON_APP_NAME = 'dnocenter_github_sync.py'
15   HOME = "${WORKSPACE}"
16 }
17
18
19
20 options {
```

- Jenkins configured credentials are encrypted by the Jenkins instance ID
- Can not be copied from a Jenkins instance
- Could be used in pipeline projects via their credential IDs

# Clone or Pull Code from GitHub

## Pipeline Stages and Steps

### Pipeline Script:

1. Selecting the Agent – Docker image Python latest
2. Pull or Clone app code and params files

The screenshot shows the GitHub Actions configuration page for a workflow named 'GitHub Sync'. The 'Pipeline' tab is selected in the left sidebar. The 'Definition' dropdown is set to 'Pipeline script'. The 'Script' section contains a Groovy script for a stage named 'Pull or Clone the source code'. The script defines a step that sets the environment variable 'HOME' to the workspace and then runs a script. This script checks if the repository exists and either pulls or clones the code from GitHub. The 'Use Groovy Sandbox' checkbox is checked. The 'Pipeline Syntax' section is visible at the bottom.

```
38 }
39 }
40 stage('Pull or Clone the source code') {
41   steps {
42     withEnv(["HOME=${env.WORKSPACE}"]) {
43       script {
44         if (fileExists(REPO_name)) {
45           echo('\n\nPull code from GitHub:.....')
46           dir(REPO_NAME) {
47             sh 'git pull --ff-only https://'+ GITHUB_USERNAME + ':' + GITHUB_TOKEN + '@github.com/cisco-en-p
48           }
49         }
50         if (!fileExists(REPO_NAME)) {
51           echo('\n\nClone code from GitHub:.....')
52           sh 'git clone https://'+ GITHUB_USERNAME + ':' + GITHUB_TOKEN + '@github.com/cisco-en-programmabilit
53         }
54       }
55     }
56   }
57 }
```

☒ Use Groovy Sandbox ?

Pipeline Syntax

Save Apply

- Verify if source code exists and pull or clone code from GitHub

# Install Python Libraries

Use "requirements.txt", Included with the Repo

## Pipeline Script:

1. Selecting the Agent – Docker image Python latest
2. Pull or Clone app code and params files
3. Install all required libraries and verify

### Definition

Pipeline script

### Script ?

```
49     }
50     if (!fileExists(REPO_NAME)) {
51         echo('\n\nClone code from GitHub:.....')
52         sh 'git clone https://' + GITHUB_USERNAME + ':' + GITHUB_TOKEN + '@github.com/cisco-en-programmability'
53     }
54 }
55 }
56 }
57 }
58 stage ('Install Python libraries') {
59     steps {
60         withEnv(["HOME=${env.WORKSPACE}"]) {
61             sh 'pip install -r ' + REPO_NAME + '/requirements.txt --no-warn-script-location'
62             echo('\n\nVerify Python version and Libraries:.....')
63             sh 'python --version'
64             sh 'pip3 list'
65             echo('\n\nVerify Application Files:.....')
66             sh 'ls ' + REPO_NAME + '/'
67         }
68     }
69 }
```

- PIP install "requirements.txt" or each library, one at a time
- Verify Python version, modules versions, pulled repo files

# Automation Workflow Run

## Shell Commands to Execute Python App

Dashboard > GitHub Sync > Configuration

### Configure

- General
- Advanced Project Options
- Pipeline**

Definition

Pipeline script

```
Script ?
61 sh 'pip install -r ' + REPO_NAME + '/requirements.txt --no-warn-script-location'
62 echo('\n\nVerify Python version and Libraries:.....')
63 sh 'python --version'
64 sh 'pip3 list'
65 echo('\n\nVerify Application Files:.....')
66 sh 'ls ' + REPO_NAME + '/'
67
68 }
69
70 stage ('Execute Python application') {
71   steps {
72     echo('\n\nApp ' + PIPELINE_NAME + ' run:.....')
73     withEnv(["HOME=${env.WORKSPACE}"]) {
74       echo('\n\nVerify Working Path:.....')
75       sh 'pwd'
76       dir(REPO_NAME) {
77         sh 'python3 ' + PYTHON_APP_NAME
78       }
79     }
80   }
81 }
82
83 }
84
85 post {
86   always {
87     echo '\n\nJenkins ' + PIPELINE_NAME + ' build end'
88   }
89 }
90
91 }
```

### Pipeline Script:

1. Selecting the Agent – Docker image Python latest
2. Pull or Clone app code
3. Install all required libraries and verify
4. Application run
5. Post tasks – destroy container, clean up files, create reports

Shell commands to run the Python App in the working directory

Optional post tasks

# Jenkins Pipeline Template

Dashboard > Jenkinsfile Template > Configuration

## Configure

- General
- Advanced Project Options
- Pipeline**

```
1 pipeline {
2   agent
3     { docker
4       {
5         reuseNode true
6         image 'python:latest'
7         args '--name DEV -v $HOME/DockerFiles:/root/DockerFiles -u root:sudo'
8       }
9     }
10  }
11
12  environment {
13    JENKINS_REPO_NAME = 'GitHub repo'
14    GITHUB_USERNAME = 'your GitHub user Id'
15    GITHUB_TOKEN = credentials('GITHUB_TOKEN')
16    PIPELINE_NAME = 'Pipeline name'
17    PYTHON_APP_NAME = 'python_app.py'
18    HOME = "${WORKSPACE}"
19  }
20
21  options {
22    skipStagesAfterUnstable()
23    timeout(time: 15, unit: 'MINUTES')
24  }
25
26  stages {
27    stage ('Environment variables') { }
28    stage ('Build Python Environment') { }
29    stage ('Pull or Clone the source code') { }
30    stage ('Install Python libraries') { }
31    stage ('Execute Python application') { }
32    stage ('Output folder content') { }
33  }
34
35  post {
36    always {
37      echo '\n\nJenkins ' + PIPELINE_NAME + ' build end'
38    }
39  }
40 }
```

Template to simplify development of new Jenkins pipelines

- Developer will customize:
  - The docker agent
  - The environment variables
- Using typical stages and steps:
  - Git pull or clone the source code from GitHub
  - Verification for every stage/step
  - Execute the application
  - Create reports/outputs

- You may use the “Jenkinsfile Template”, to start building your pipelines.
- Easy to get started in few simple steps!

# Agenda

- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
  - How to Build a Pipeline
  - Device Inventory
  - GitHub Integration App
  - CLI Templates Deployment
  - SDA-as-Code
  - PoE Sustainability
- Developer Resources
- Summary

# Device Inventory Pipeline

- Runs on a schedule, every 15 minutes
- Will collect the Cisco DNA Center device inventory
- Push the inventory files to GitHub
- Always up-to-date rich inventory data may be used by other automations tools and platforms
- It includes device and AP inventories, software non-compliant devices list, formatted as JSON and YAML files

The screenshot shows the Jenkins dashboard with the 'OpsView' tab selected. The 'Device Inventory' build is highlighted with a green box. The build queue is empty.

| S | W | Name ↓                   | Last Success     | Last Failure     | Last Duration | Fav |
|---|---|--------------------------|------------------|------------------|---------------|-----|
| ✓ | ☁ | CLI Templates Deployment | 1 mo 22 days #14 | 1 mo 22 days #13 | 2 min 30 sec  | ▶ ☆ |
| ✓ | ☁ | Device Inventory         | 7 min 29 sec #24 | 2 hr 17 min #21  | 48 sec        | ▶ ☆ |
| ✓ | ☀ | GitHub Sync              | 21 hr #87        | N/A              | 28 sec        | ▶ ☆ |
| ✓ | ☀ | GitLab Sync              | 21 hr #67        | N/A              | 23 sec        | ▶ ☆ |
| ✓ | ☀ | PoE Sustainability       | 1 mo 24 days #56 | N/A              | 16 sec        | ▶ ☆ |
| ✓ | ☀ | SDA-as-Code              | 1 mo 2 days #5   | N/A              | 1 min 19 sec  | ▶ ☆ |

Build Queue: No builds in the queue.

# Device Inventory Pipeline

## Schedule

### Jenkins triggers:

- Trigger builds remotely (API call)
- Periodically
- GitHub actions
- After another build is finished

Dashboard > OpsView > Device Inventory > Configuration

### Configure

- General
- Advanced Project Options
- Pipeline

#### Build Triggers

☐ Build after other projects are built ?

☒ Build periodically ?

Schedule ?

H/15 \* \* \* \*

Would last have run at Monday, May 22, 2023 at 6:59:10 PM Pacific Daylight Time; would next run at Monday, May 22, 2023 at 7:14:10 PM Pacific Daylight Time.

☐ GitHub Branches

☐ GitHub Pull Requests ?

☐ GitHub hook trigger for GITScm polling ?

☐ Poll SCM ?

☐ Quiet period ?

☐ Trigger builds remotely (e.g., from scripts) ?

#### Note:

*This build will run every 15 minutes*

General Build Triggers Advanced Project Options Pipeline

☒ Build periodically ?

Schedule ?

H/15 \* \* \* \*

Would last have run at Tuesday, June 7, 2022 at 3:14:03 PM Pacific Daylight Time; would next run at Tuesday, June 7, 2022 at 3:29:03 PM Pacific Daylight Time.

This field follows the syntax of cron (with minor differences). Specifically, each line consists of 5 fields separated by TAB or whitespace:

| MINUTE | HOUR                                                | DOM | MONTH | DOW |
|--------|-----------------------------------------------------|-----|-------|-----|
| MINUTE | Minutes within the hour (0-59)                      |     |       |     |
| HOUR   | The hour of the day (0-23)                          |     |       |     |
| DOM    | The day of the month (1-31)                         |     |       |     |
| MONTH  | The month (1-12)                                    |     |       |     |
| DOW    | The day of the week (0-7) where 0 and 7 are Sunday. |     |       |     |

To specify multiple values for one field, the following operators are available. In the order of precedence,

- \* specifies all valid values
- N-N specifies a range of values
- N-N/X or \*/X steps by intervals of X through the specified range or whole valid range
- A,B,...,Z enumerates multiple values

To allow periodically scheduled tasks to produce even load on the system, the symbol H (for "hash") should be used wherever possible. For example, using `H * * * *` for a dozen daily jobs will cause a large spike at midnight. In contrast, using `H H * * *` would still execute each job once a day, but not all at the same time, better using limited resources.

The H symbol can be used with a range. For example, `H H(8-7) * * *` means some time between 12:00 AM (midnight) to 7:59 AM. You can also use step intervals with H, with or without ranges.

# Device Inventory Pipeline

## Configuration – Copy and Update from Jenkins Pipeline Template

Dashboard > OpsView > Device Inventory > Configuration

### Configure

- General
- Advanced Project Options
- Pipeline**

Script ?

```
1 pipeline {
2
3
4   agent
5   { docker
6     { image 'python:latest' }
7   }
8
9   environment {
10     JENKINS_REPO_NAME = 'dnacenter_jenkins_automations'
11     GITHUB_USERNAME = 'gzapodea'
12     GITHUB_TOKEN = credentials('GITHUB_TOKEN')
13     PIPELINE_NAME = 'Device Inventory'
14     PYTHON_APP_NAME = 'device_inventory.py'
15     HOME = "${WORKSPACE}"
16   }
17
18   options {
19     skipStagesAfterUnstable()
20     timeout(time: 15, unit: 'MINUTES')
21   }
22
23   stages {
24
25     stage('Build Python Environment') {
26       steps {
27         echo '\n\nJenkins ' + PIPELINE_NAME + ' build start.....'
28         echo('\n\nBuilding the Docker container:.....')
29       }
30     }
31   }
32 }
```

Pipeline Template updated with:

- Docker image
- GitHub repo, username
- Pipeline and app names
- GitHub credential

# Device Inventory Pipeline

## Configuration – Verify Stages and Steps

Dashboard > OpsView > Device Inventory > Configuration

### Configure

- General
- Advanced Project Options
- Pipeline**

```
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
stage('Install Python libraries') {
    steps {
        echo('\n\nInstalling Python libraries:.....')
        sh 'git clone https://'+ GITHUB_USERNAME + ':' + GITHUB_TOKEN + '@github.com:zapodeanu/' + JENKINS_REPO_NAME
        withEnv(["HOME=${env.WORKSPACE}"]) {
            sh 'pip install -r '+ JENKINS_REPO_NAME + '/requirements.txt --no-warn-script-location'
            echo('\n\nVerify Python version and Libraries:.....')
            sh 'python --version'
            sh 'pip3 list'
            echo('\n\nVerify Application Files:.....')
            sh 'ls '+ JENKINS_REPO_NAME + '/'
        }
    }
}

stage('Execute Python application') {
    steps {
        echo('\n\nCollect the inventory of devices managed by Cisco DNA Center:.....')
        withEnv(["HOME=${env.WORKSPACE}"]) {
            echo('\n\nVerify Working Path:.....')
            sh 'pwd'
            dir(JENKINS_REPO_NAME) {
                sh 'python3 '+ PYTHON_APP_NAME
            }
        }
    }
}

stage('Verify collected data - local folder') {
    steps {
        echo('\n\nThe inventory files that have been created:.....')
        withEnv(["HOME=${env.WORKSPACE}"]) {
            echo('\n\nVerify Working Path:.....')
            sh 'pwd'
            dir(JENKINS_REPO_NAME + '/inventory') {
                sh 'ls -l -A'
            }
        }
    }
}
```

Verify and update pre-configured, pipeline template stages and steps:

- Pull code from GitHub
- Install libraries, verify
- Execute the Python App
- Verify the files saved in local folder

# Device Inventory Pipeline

## Logging

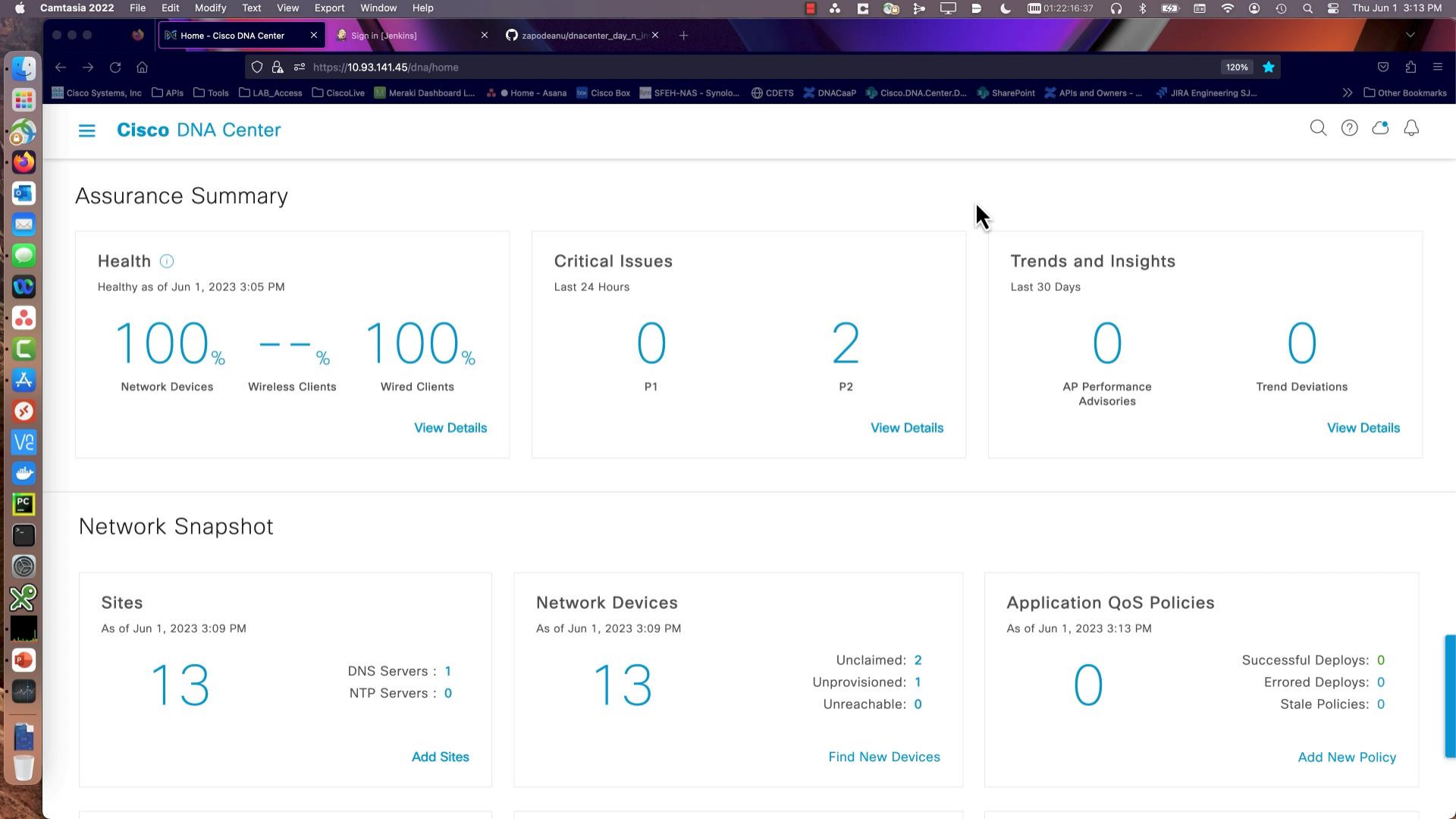
```
device_inventory.py x
152
153 logging.info(' Collected the device inventory from Cisco DNA Center')
154
155 # save device inventory to json and yaml formatted files
156 with open('inventory/device_inventory.json', 'w') as f:
157     f.write(json.dumps(device_inventory))
158 logging.info(' Saved the device inventory to file "device_inventory.json"')
159
160 with open('inventory/device_inventory.yaml', 'w') as f:
161     f.write('device_inventory:\n' + yaml.dump(device_inventory, sort_keys=False))
162 logging.info(' Saved the device inventory to file "device_inventory.yaml"')
163
164 # save ap inventory to json and yaml formatted files
165 with open('inventory/ap_inventory.json', 'w') as f:
166     f.write(json.dumps(ap_inventory))
167 logging.info(' Saved the device inventory to file "ap_inventory.json"')
168
169 with open('inventory/ap_inventory.yaml', 'w') as f:
170     f.write('ap_inventory:\n' + yaml.dump(ap_inventory, sort_keys=False))
171 logging.info(' Saved the device inventory to file "ap_inventory.yaml"')
172
```

Enable Python App logging  
to view them in the  
Pipeline Console, when building

Dashboard > OpsView > Device Inventory > #27

```
+ pwd
/var/lib/jenkins/workspace/Device Inventory
[Pipeline] dir
Running in /var/lib/jenkins/workspace/Device Inventory/dnacenter_jenkins_automations
[Pipeline] {
[Pipeline] sh
+ python3 device_inventory.py
INFO:root: App "device_inventory.py" run start, 2023-05-22 19:44:25
INFO:root: Number of devices managed by Cisco DNA Center: 13
INFO:root: Collected the device list from Cisco DNA Center
INFO:root: Collected the device inventory from Cisco DNA Center
INFO:root: Saved the device inventory to file "device_inventory.json"
INFO:root: Saved the device inventory to file "device_inventory.yaml"
INFO:root: Saved the device inventory to file "ap_inventory.json"
INFO:root: Saved the device inventory to file "ap_inventory.yaml"
INFO:root: Number of devices image non-compliant: 2
INFO:root: Image non-compliant devices:
INFO:root: PDX-STACK, Site Hierarchy: Global/OR/PDX/Floor-2
INFO:root: LO-BN, Site Hierarchy: Global/OR/LO/Floor-3
INFO:root: Saved the image non-compliant device inventory to file "non_compliant_devices.json"
INFO:root: Saved the image non-compliant device inventory to file "non_compliant_devices.yaml"
INFO:root: GitHub push for file: ap_inventory.yaml
INFO:root: GitHub push for file: device_inventory.yaml
INFO:root: GitHub push for file: device_inventory.json
INFO:root: GitHub push for file: ap_inventory.json
INFO:root: GitHub push for file: non_compliant_devices.yaml
INFO:root: GitHub push for file: non_compliant_devices.json
INFO:root: App "device_inventory.py" run end: 2023-05-22 19:44:39
```

# Demo Device Inventory



## Cisco DNA Center

### Assurance Summary

#### Health ⓘ

Healthy as of Jun 1, 2023 3:05 PM

100%

--%

100%

Network Devices

Wireless Clients

Wired Clients

[View Details](#)

#### Critical Issues

Last 24 Hours

0

2

P1

P2

[View Details](#)

#### Trends and Insights

Last 30 Days

0

0

AP Performance Advisories

Trend Deviations

[View Details](#)

### Network Snapshot

#### Sites

As of Jun 1, 2023 3:09 PM

13

DNS Servers : 1

NTP Servers : 0

[Add Sites](#)

#### Network Devices

As of Jun 1, 2023 3:09 PM

13

Unclaimed: 2

Unprovisioned: 1

Unreachable: 0

[Find New Devices](#)

#### Application QoS Policies

As of Jun 1, 2023 3:13 PM

0

Successful Deploys: 0

Errored Deploys: 0

Stale Policies: 0

[Add New Policy](#)

# Agenda

- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
  - How to Build a Pipeline
  - Device Inventory
  - GitHub Integration App
  - CLI Templates Deployment
  - SDA-as-Code
  - PoE Sustainability
- Developer Resources
- Summary

# GitHub Integration App Workflow



**GitHub**

**GitHub  
Integration  
App**

zapodeanu / dnacenter\_github\_templates Public

updated ntp source

master

zapodeanu committed 1 minute

Showing 1 changed file with 2 additions

snmp\_ntp.txt

```
@@ -5,5 +5,6 @@ snmp-server host 10.93.130.50 version 2c RW
5 ntp server 171.68.48.78
6 ntp server 171.68.38.53
7 no ntp server 171.68.48.80
8 - ntp source Loopback1
9 !
```

```
INFO:root: Template name: snmp_ntp
INFO:root: Template content:
{#
This template has been pulled from GitHub.
Uploaded to Cisco DNA Center by GitHub_Sync App
Author: gzapodea@cisco.com
Date: 2023-04-19T22:28:26Z
Commit message: updated ntp source
Commit URL: https://github.com/zapodeanu/dnacenter_github_templates/commit/82ed7d6
Commit Diff: @@ -5,5 +5,6 @@ snmp-server host 10.93.130.50 version 2c RW
ntp server 171.68.48.78
ntp server 171.68.38.53
no ntp server 171.68.48.80
-ntp source Loopback1
+no ntp source Loopback1
+ntp source Loopback0
!
\ No newline at end of file
#}
!
snmp-server host 10.93.135.30 version 2c RO
snmp-server host 10.93.130.50 version 2c RW
!
ntp server 171.68.48.78
ntp server 171.68.38.53
no ntp server 171.68.48.80
no ntp source Loopback1
ntp source Loopback0
!
INFO:root: Template "snmp_ntp" has changed, different template on Cisco DNA Center
INFO:root: Updating existing template "snmp_ntp" id: 27c20f95-b66b-4884-8d82-94c841805c9f
INFO:root: Template "snmp_ntp" committed
```

snmp\_ntp x

Actions | Edit | Properties

Template System Variables

Template

```
1 {#
2 This template has been pulled from GitHub.
3 Uploaded to Cisco DNA Center by GitHub_Sync App
4 Author: gzapodea@cisco.com
5 Date: 2023-04-19T22:28:26Z
6 Commit message: updated ntp source
7 Commit URL: https://github.com/zapodeanu/dnacenter_github_templates/commit/82ed7d6
8 Commit Diff: @@ -5,5 +5,6 @@ snmp-server host 10.93.130.50 version 2c RW
9 ntp server 171.68.48.78
10 ntp server 171.68.38.53
11 no ntp server 171.68.48.80
12 -ntp source Loopback1
13 +no ntp source Loopback1
14 +ntp source Loopback0
15 !
16 \ No newline at end of file
17 #}
18 !
19 !
20 snmp-server host 10.93.135.30 version 2c RO
21 snmp-server host 10.93.130.50 version 2c RW
22 !
23 ntp server 171.68.48.78
24 ntp server 171.68.38.53
25 no ntp server 171.68.48.80
26 no ntp source Loopback1
27 ntp source Loopback0
28 !
```

**Note:**  
Will call Cisco DNA Center  
and GitHub APIs

- Running on Jenkins, every evening
- Synchronize GitHub Repos with Cisco DNA Center projects and templates
- Includes the commit details: comments, author, date, diff
- Create CLI template, or update, if changes
- Commit the template
- Create report

# GitHub Sync Pipeline

## Copy and Update from Jenkins Pipeline Template

Dashboard > OpsView > GitHub Sync > Configuration

### Configure

- General
- Advanced Project Options
- Pipeline**

```
1 pipeline {
2   agent
3     { docker
4       {
5         reuseNode true
6         image 'python:latest'
7       }
8     }
9 }
10
11 environment {
12   REPO_NAME = 'dnacenter_github_integration'
13   GITHUB_USERNAME = 'zapodeanu'
14   GITHUB_TOKEN = credentials('GITHUB_TOKEN')
15   PIPELINE_NAME = 'GitHub Sync'
16   PYTHON_APP_NAME = 'dnacenter_github_sync.py'
17   HOME = "${WORKSPACE}"
18 }
19
20 options {
21   skipStagesAfterUnstable()
22   timeout(time: 15, unit: 'MINUTES')
23 }
24
25 stages {
26   stage ('Environment variables') { }
27   stage ('Build Python Environment') { }
28   stage ('Pull or Clone the source code') { }
29   stage ('Install Python libraries') { }
30   stage ('Execute Python application') { }
31 }
32
33 post {
34   always {
35     echo '\n\nJenkins ' + PIPELINE_NAME + ' build end'
36   }
37 }
38 }
```

- Agent – Docker and Reuse
- Update the environment variables
- Typical Stages and Steps

# Jenkins Pipeline Build – Scheduled

Dashboard > OpsView > GitHub Sync >

Status

</> Changes

▶ Build Now

⚙️ Configure

🗑️ Delete Pipeline

🔍 Full Stage View

🌐 Open Blue Ocean

✎ Rename

❓ Pipeline Syntax

Build History

trend ▾

🔍 Filter builds...

/

✓ #87 May 21, 2023, 9:42 PM

✓ #86 May 20, 2023, 9:42 PM

✓ #85 May 19, 2023, 9:42 PM

✓ #84 May 18, 2023, 9:42 PM

✓ #83 May 17, 2023, 9:42 PM

✓ #82 May 16, 2023, 9:42 PM

Pipeline GitHub Sync

Open-source integration synchronizing GitHub repos as single source of truth with Cisco DNA Center projects/templates.

[https://github.com/cisco-en-programmability/dnacenter\\_git\\_integrations](https://github.com/cisco-en-programmability/dnacenter_git_integrations)

Stage View

Average stage times:  
(Average full run time: ~28s)

|                                      | Environment variables | Build Python Environment | Pull or Clone the source code | Install Python libraries | Execute Python application | Declarative: Post Actions |
|--------------------------------------|-----------------------|--------------------------|-------------------------------|--------------------------|----------------------------|---------------------------|
| #87<br>May 21<br>21:42<br>No Changes | 91ms                  | 65ms                     | 1s                            | 6s                       | 16s                        | 60ms                      |
| #86<br>May 20<br>21:42<br>No Changes | 126ms                 | 70ms                     | 1s                            | 6s                       | 16s                        | 56ms                      |
| #85<br>May 19<br>21:42<br>No Changes | 82ms                  | 62ms                     | 1s                            | 6s                       | 16s                        | 56ms                      |
| #84<br>May 18<br>21:42<br>No Changes | 86ms                  | 81ms                     | 1s                            | 6s                       | 16s                        | 72ms                      |

- Build history – running on a schedule, or on-demand
- Stage View – average stage and each build time

CISCO *Live!*

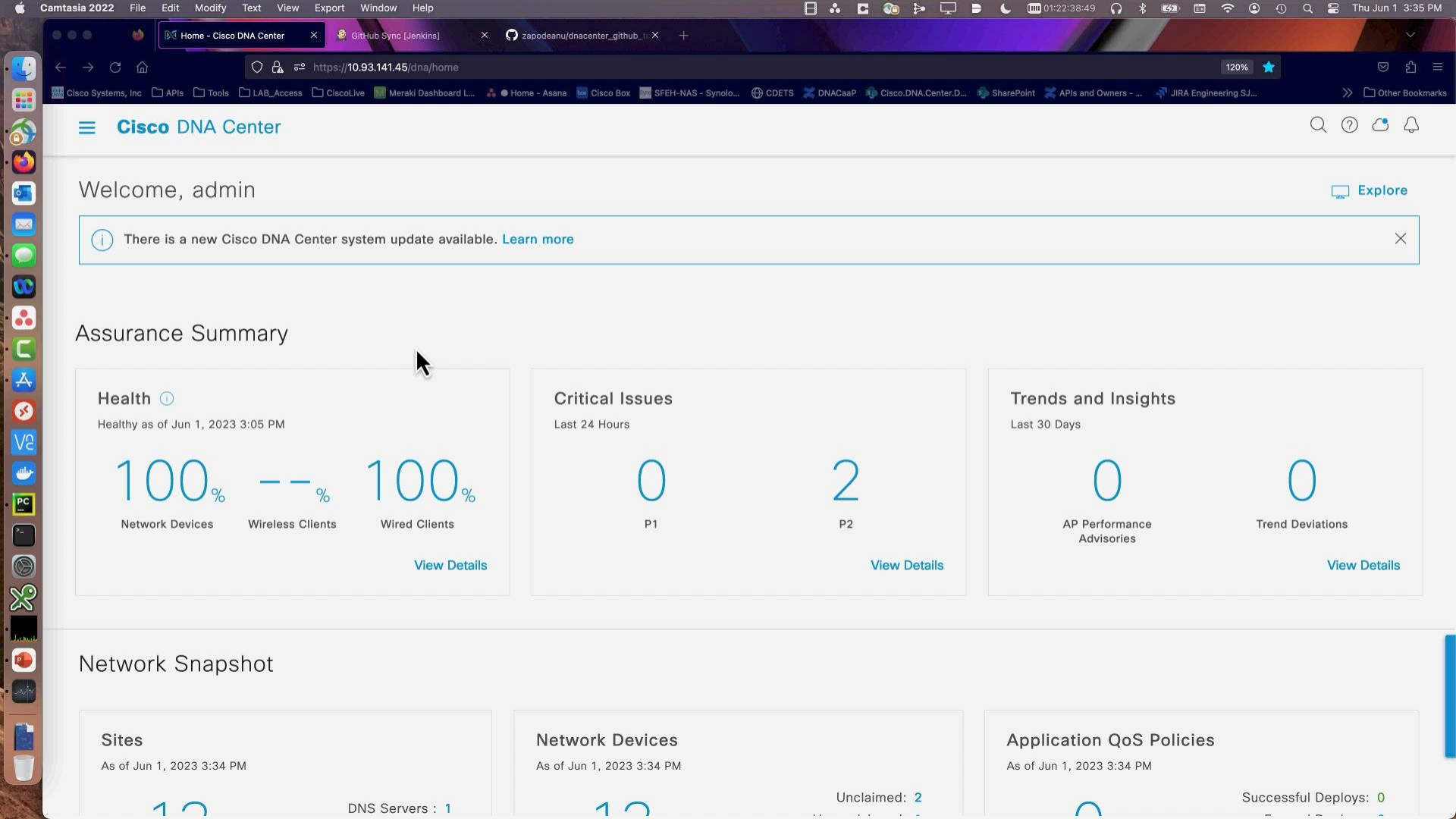
#CiscoLive

DEVNET-2151

© 2023 Cisco and/or its affiliates. All rights reserved. Cisco Public

48

# Demo GitHub Integration App



## Cisco DNA Center

Welcome, admin

Explore

There is a new Cisco DNA Center system update available. [Learn more](#)

### Assurance Summary

#### Health ⓘ

Healthy as of Jun 1, 2023 3:05 PM

100%

Network Devices

--%

Wireless Clients

100%

Wired Clients

[View Details](#)

#### Critical Issues

Last 24 Hours

0

P1

2

P2

[View Details](#)

#### Trends and Insights

Last 30 Days

0

AP Performance  
Advisories

0

Trend Deviations

[View Details](#)

### Network Snapshot

#### Sites

As of Jun 1, 2023 3:34 PM

#### Network Devices

As of Jun 1, 2023 3:34 PM

#### Application QoS Policies

As of Jun 1, 2023 3:34 PM

DNS Servers : 1

Unclaimed: 2

Successful Deploys: 0

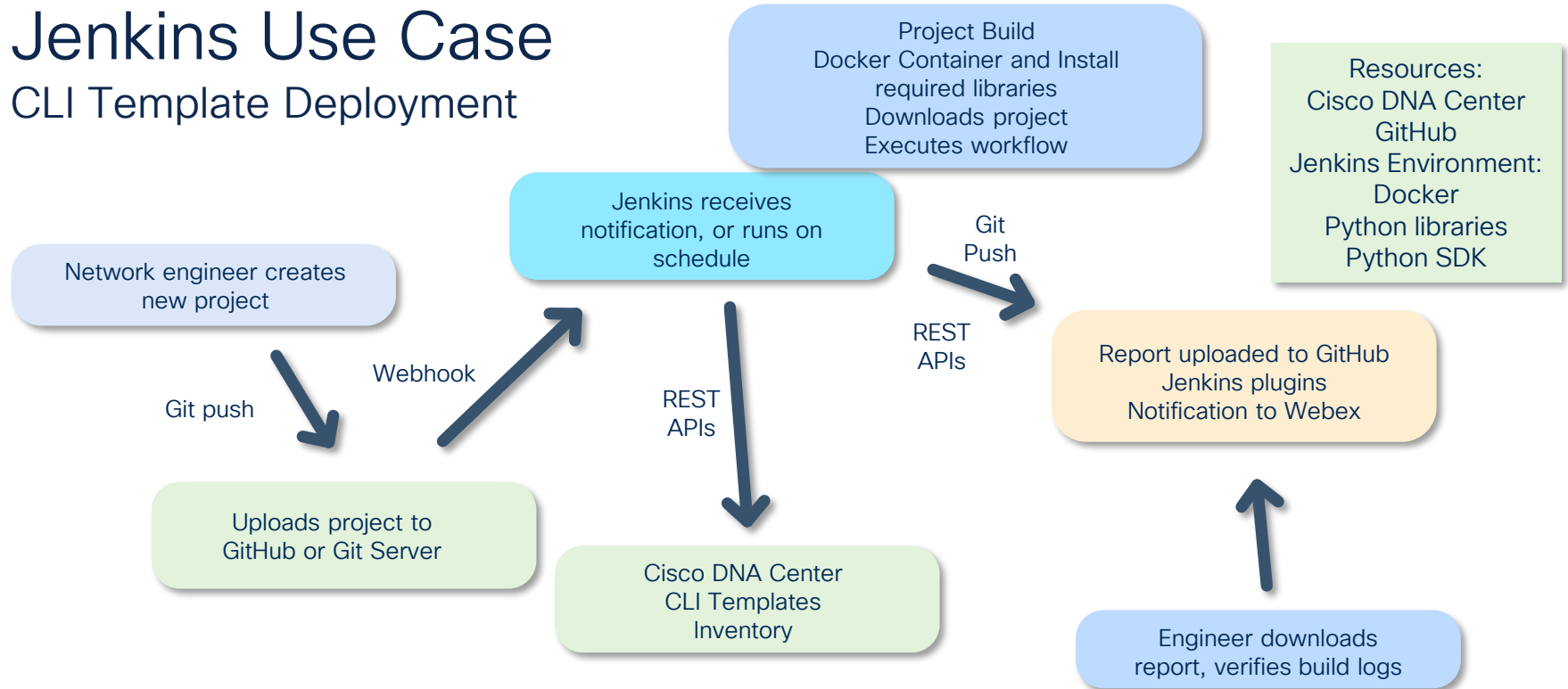
# Agenda



- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
  - How to Build a Pipeline
  - Device Inventory
  - GitHub Integration App
  - CLI Templates Deployment
  - SDA-as-Code
  - PoE Sustainability
- Developer Resources
- Summary

# Jenkins Use Case

## CLI Template Deployment



# CLI Deployment – Lab & Production

## Project Details

```
dna_center:
  ip_address: 10.93.141.45

devices_lab:
  hostname: [PDX-RN]

devices_info:
  hostname: [PDX-RO, SP, NYC-RO]

project:
  name: Jenkins

template:
  name: cli_template.txt
```

## CLI Template

```
!
interface loopback{{interface_number}}
description management interface
ip address {{ip_address}}
!
```

## Network Engineer

### Pipelines and Apps:

- Pre-defined and tested
  - Meet enterprise standards
- Network + DevOps Engineer

## Pipeline Build

CLI Templates Deployment > #16

+ python3 lab\_deploy\_templates.py

INFO:root: App "lab\_deploy\_templates.py" Start, 2023-06-01 16:01:23

INFO:root: Project name: Jenkins

CLI Templates Deployment > #16

INFO:root: Project name: Jenkins

INFO:root: Project id: ae404603-f2a8-43d5-bd81-68f6e24abce9

INFO:root: Template name: cli\_template

INFO:root: Template found and deleted

INFO:root: Device list: ['PDX-RO', 'SP', 'NYC-RO']

INFO:root: CLI Template:

!

!

interface loopback 500

description Interface - Jenkins\_config

!

ip access-list extended RETAIL

remark Access List for RETAIL - Jenkins\_config

10 permit ip any host 10.94.141.90

20 permit ip any host 10.94.141.91

!

!

INFO:root: Created template with the name: cli\_template

INFO:root: Template committed

INFO:root: Deploying template to device: PDX-RO

INFO:root: Deploying template to device: SP

INFO:root: Deploying template to device: NYC-RO

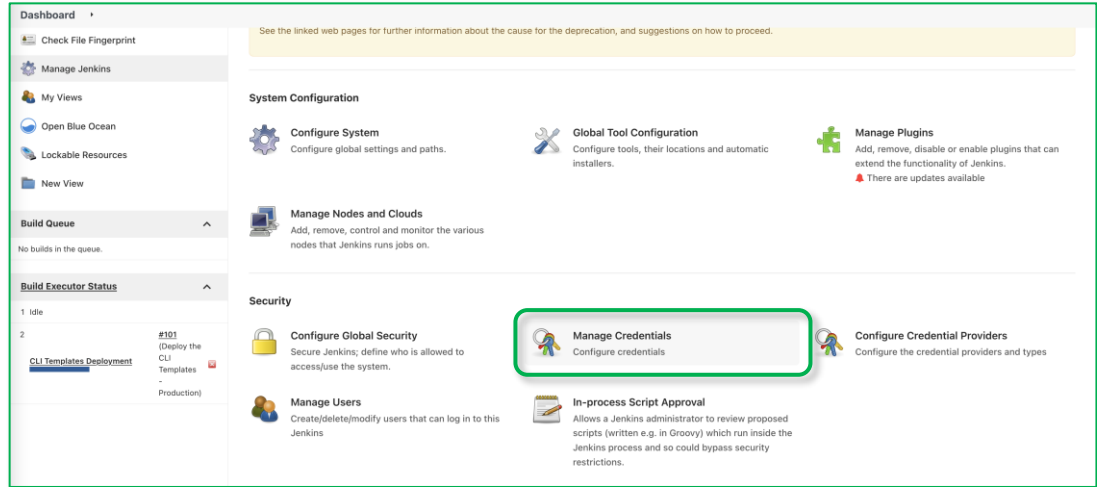
INFO:root: Deployment Report:

INFO:root: {"timestamp": "2023-06-01 16:02:16", "template\_content": "!\\n\\ninterface loopback 500\\n description Interface - Jenkins\_config\\n\\nip access-list extended RETAIL\\n remark Access List for RETAIL - Jenkins\_config\\n 10 permit ip any host 10.94.141.90\\n 20 permit ip any host 10.94.141.91\\n\\n\\n", "report": [{"hostname": "PDX-RO", "status": "successful"}, {"hostname": "SP", "status": "successful"}, {"hostname": "NYC-RO", "status": "successful"}]}

# Create a Cisco DNA Center Jenkins Pipeline

## Assumptions

- You have Jenkins installed and running
- Docker is installed and running
- You have created a GitHub Token (if planning to use GitHub)
- Installation and quick start may be found here: <https://jenkins.io/>



### Note:

Add the GitHub Token to your Jenkins credentials

# Pipeline CLI Templates Deployment - Environment

Dashboard ▸ CLI Templates Deployment ▸

General Build Triggers Advanced Project Options **Pipeline**

Pipeline script ▾

**Script ?**

```
1
2 pipeline {
3
4   agent
5     { docker
6       { image 'python:latest' }
7     }
8
9   environment {
10     JENKINS_REPO_NAME = 'dnacenter_jenkins'
11     TEMPLATES_REPO_NAME = 'templates_jenkins'
12     GITHUB_USERNAME = 'gzapodea'
13     GITHUB_TOKEN = credentials('GITHUB_TOKEN')
14   }
15
16   options {
17     skipStagesAfterUnstable()
18     timeout(time: 15, unit: 'MINUTES')
19   }
20
21   stages {
22     stage ('Build Python Environment') {
23       steps {
24         echo('\n\nBuilding the Docker container and Installing Python libraries:.....')
25         sh 'git clone https://' + GITHUB_USERNAME + ':' + GITHUB_TOKEN + '@github.com/zapodeanu/' + JENKINS_REPO_NAME
26         withEnv(["HOME=${env.WORKSPACE}"]) {
27           sh 'pip install -r ' + JENKINS_REPO_NAME + '/requirements.txt --no-warn-script-location'
28           echo('\n\nVerify Python version and Libraries:.....')
29           sh 'python --version'
30           sh 'pip3 list'
31           echo('\n\nVerify Application Files:.....')
32           sh 'ls ' + JENKINS_REPO_NAME + '/'
33         }
34       }
35     }
36   }
37 }
```

Select a Docker image

Define the environment variables

Clone the GitHub repo with the app code  
Install and verify the required Python libraries

# Pipeline CLI Templates Deployment - Deploy

Dashboard > CLI Templates Deployment >

General Build Triggers Advanced Project Options **Pipeline**

Pipeline script

Script ?

```
38 stage ('Download the Project Repo') {
39   steps {
40     echo('\n\nDownloading the CLI Templates:.....')
41     sh 'git clone https://' + GITHUB_USERNAME + ':' + GITHUB_TOKEN + '@github.com/zapodeanu/' + TEMPLATES_REPO_NAME + '.git'
42     withEnv(["HOME=${env.WORKSPACE}"]) {
43       echo('\n\nVerify Templates Files:.....')
44       sh 'ls ' + TEMPLATES_REPO_NAME + '/'
45     }
46   }
47 }
48
49 stage ('Deploy the CLI Templates - Lab') {
50   steps {
51     echo('\n\nLab Deploying the CLI Templates:.....')
52     withEnv(["HOME=${env.WORKSPACE}"]) {
53       echo('\n\nVerify Working Path:.....')
54       sh 'pwd'
55       dir('dnacenter-jenkins') {
56         sh 'python3 lab_deploy_templates.py'
57       }
58     }
59   }
60 }
61
62 stage ('Deploy the CLI Templates - Production') {
63   steps {
64     echo('\n\nDeploying the CLI Templates:.....')
65     withEnv(["HOME=${env.WORKSPACE}"]) {
66       echo('\n\nVerify Working Path:.....')
67       sh 'pwd'
68       dir('dnacenter-jenkins') {
69         sh 'python3 deploy_templates.py'
70       }
71     }
72   }
73 }
```

Clone from GitHub the CLI templates repo

Deploy template to lab devices

Deploy template to production devices

# Pipeline CLI Templates Deployment - Report

Dashboard ▸ CLI Templates Deployment ▸

General Build Triggers Advanced Project Options **Pipeline**

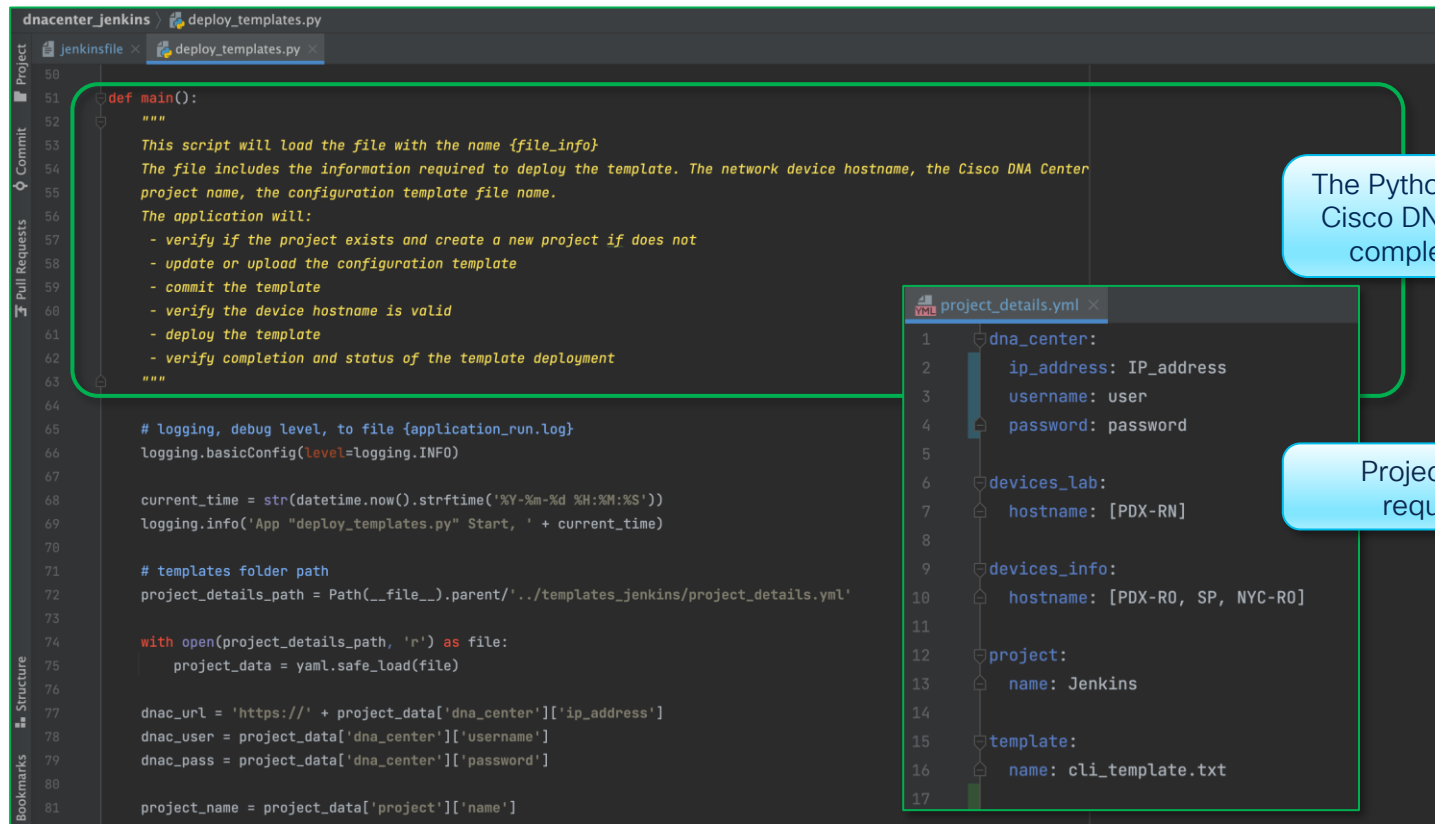
```
74
75 stage ('Upload Deployment Report') {
76     steps {
77         echo('\n\nVerify if the deployment report file exists:.....')
78         withEnv(["HOME=${env.WORKSPACE}"]) {
79             dir('templates_jenkins') {
80                 sh 'ls deployment_report.json'
81
82                 echo('\n\nUploading to GitHub the deployment report:.....')
83
84                 sh 'git config --global user.email "gzapodea@cisco.com"'
85                 sh 'git config --global user.name "gzapodea"'
86                 sh 'git add deployment_report.json'
87                 sh 'git commit -m "committed by Jenkins"'
88                 sh 'git push origin master --force'
89             }
90         }
91     }
92 }
93
94
95
96 post {
97     cleanup {
98         cleanWs()
99     }
100
101     always {
102         echo '\n\nJenkins CLI Templates Deployment end'
103     }
104 }
105
106 }
107
108
109
110
```

Use Groovy Sandbox ?

Create report and push to GitHub

Remove the docker container

# CLI Templates Python Apps – Lab/Prod



The screenshot shows a code editor with two files open. The left file, `deploy_templates.py`, is a Python script with a `main()` function. The right file, `project_details.yml`, is a YAML configuration file. The Python script includes a docstring describing its purpose and a list of steps it will perform. It also shows logging setup, time tracking, and the loading of the YAML file. The YAML file defines parameters for the Cisco DNA Center, lab devices, and the project details.

```
def main():
    """
    This script will load the file with the name {file_info}
    The file includes the information required to deploy the template. The network device hostname, the Cisco DNA Center
    project name, the configuration template file name.
    The application will:
    - verify if the project exists and create a new project if does not
    - update or upload the configuration template
    - commit the template
    - verify the device hostname is valid
    - deploy the template
    - verify completion and status of the template deployment
    """

    # logging, debug level, to file {application_run.log}
    logging.basicConfig(level=logging.INFO)

    current_time = str(datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
    logging.info('App "deploy_templates.py" Start, ' + current_time)

    # templates folder path
    project_details_path = Path(__file__).parent/ '../templates_jenkins/project_details.yml'

    with open(project_details_path, 'r') as file:
        project_data = yaml.safe_load(file)

    dnac_url = 'https://' + project_data['dna_center']['ip_address']
    dnac_user = project_data['dna_center']['username']
    dnac_pass = project_data['dna_center']['password']

    project_name = project_data['project']['name']
```

```
1 dna_center:
2   ip_address: IP_address
3   username: user
4   password: password
5
6 devices_lab:
7   hostname: [PDX-RN]
8
9 devices_info:
10  hostname: [PDX-R0, SP, NYC-R0]
11
12 project:
13   name: Jenkins
14
15 template:
16   name: cli_template.txt
17
```

The Python app will call the Cisco DNA Center APIs to complete these steps

Project includes the required params

## Job Build Console Output - Start

Jenkins

Search ?

Gabi Zapodeanu log out

Dashboard > CLI Templates Deployment > #105

Back to Project

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#105'

Open Blue Ocean

Restart from Stage

Replay

Pipeline Steps

Workspaces

Previous Build

Timestamps View as plain text

System clock time

Use browser timezone

✓ Console Output

20:21:44 Started by user Gabi Zapodeanu

20:21:45 [Pipeline] Start of Pipeline

20:21:45 [Pipeline] node

20:21:45 Running on Jenkins in /Users/gzapodea/.jenkins/workspace/CLI Templates Deployment@2

20:21:45 [Pipeline] {

20:21:45 [Pipeline] isUnix

20:21:45 [Pipeline] sh

+ docker inspect -f . python:latest

.

[Pipeline] withDockerContainer

Jenkins does not seem to be running inside a container

\$ docker run -t -d -u 501:20 -w "/Users/gzapodea/.jenkins/workspace/CLI Templates Deployment@2" -v "/Users/gzapodea/.jenkins/workspace/CLI Templates Deployment@2:/tmp:/Users/gzapodea/.jenkins/workspace/CLI Templates Deployment@2tmp:rw,z" -e "python:latest cat"

\$ docker top 9f18dd7fa3fdc148e06dd3c522c32defc7be7a439606e9a39bfc225a587ee3c1 -eo pid,comm

[Pipeline] {

[Pipeline] withCredentials

Masking supported pattern matches of \$GITHUB\_TOKEN

[Pipeline] {

[Pipeline] withEnv

[Pipeline] {

[Pipeline] timeout

Timeout set to expire in 15 min

[Pipeline] {

[Pipeline] stage

[Pipeline] { (Build Python Environment)

[Pipeline] echo

20:21:48

20:21:48

# Console Output – Install Python Packages

```
Dashboard  ▸ CLI Templates Deployment  ▸ #105

System clock time
☒ Use browser timezone
☐ Elapsed time
☐ None

20:21:48
20:21:48 Building the Docker container and Installing Python libraries:.....
20:21:48 [Pipeline] sh
20:21:48 + git clone https://gzapodea:****@github.com/zapodeanu/dnacenter_jenkins.git
20:21:48 Cloning into 'dnacenter_jenkins'...
20:21:50 [Pipeline] withEnv
20:21:50 [Pipeline] {
20:21:50 [Pipeline] sh
20:21:51 + pip install -r dnacenter_jenkins/requirements.txt --no-warn-script-location
20:21:52 Defaulting to user installation because normal site-packages is not writeable
20:21:53 Collecting requests
20:21:53   Downloading requests-2.27.1-py3-none-any.whl (63 kB)
20:21:53     _____ 63.1/63.1 KB 977.0 kB/s eta 0:00:00
20:21:53 Collecting urllib3
20:21:53   Downloading urllib3-1.26.9-py2.py3-none-any.whl (138 kB)
20:21:53     _____ 139.0/139.0 KB 2.4 MB/s eta 0:00:00
20:21:53 Collecting dnacentersdk
20:21:53   Downloading dnacentersdk-2.4.10-py3-none-any.whl (2.4 MB)
20:21:53     _____ 2.4/2.4 MB 5.2 MB/s eta 0:00:00
20:21:54 Collecting pyopenssl
20:21:54   Downloading pyOpenSSL-22.0.0-py2.py3-none-any.whl (55 kB)
20:21:54     _____ 55.8/55.8 KB 1.9 MB/s eta 0:00:00
20:21:54 Collecting python-dotenv
20:21:54   Downloading python_dotenv-0.20.0-py3-none-any.whl (17 kB)
20:21:54 Collecting PyYAML
20:21:54   Downloading PyYAML-6.0-cp310-cp310-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux2_12_x86_64.manylinux2010_x86_64.whl (682 kB)
20:21:55     _____ 682.2/682.2 KB 2.7 MB/s eta 0:00:00
20:21:55 Collecting charset-normalizer~2.0.0
20:21:55   Downloading charset_normalizer-2.0.12-py3-none-any.whl (39 kB)
20:21:55 Collecting idna<4,>=2.5
20:21:55   Downloading idna-3.3-py3-none-any.whl (61 kB)
20:21:55     _____ 61.2/61.2 KB 2.1 MB/s eta 0:00:00
20:21:55 Collecting certifi==2017.4.17
20:21:55   Downloading certifi-2022.5.18.1-py3-none-any.whl (155 kB)
20:21:55     _____ 155.2/155.2 KB 2.9 MB/s eta 0:00:00
20:21:55 Collecting fastjsonschema<3.0.0,>=2.14.5
20:21:55   Downloading fastjsonschema-2.15.3-py3-none-any.whl (22 kB)
20:21:55 Collecting requests-toolbelt<0.10.0,>=0.9.1
20:21:55   Downloading requests_toolbelt-0.9.1-py2.py3-none-any.whl (54 kB)
```

# Console Output – Verify Python, App Files

```
Dashboard > CLI Templates Deployment > #105

20:22:28 Verify Python version and Libraries:.....
20:22:28 [Pipeline] sh
20:22:29 + python --version
20:22:29 Python 3.10.4
20:22:29 [Pipeline] sh
20:22:30 + pip3 list
20:22:32 Package Version
20:22:32 -----
20:22:32 certifi 2022.5.18.1
20:22:32 cffi 1.15.0
20:22:32 charset-normalizer 2.0.12
20:22:32 cryptography 37.0.2
20:22:32 dnacentersdk 2.4.10
20:22:32 fastjsonschema 2.15.3
20:22:32 future 0.18.2
20:22:32 idna 3.3
20:22:32 pip 22.0.4
20:22:32 pycparser 2.21
20:22:32 pyOpenSSL 22.0.0
20:22:32 python-dotenv 0.20.0
20:22:32 PyYAML 6.0
20:22:32 requests 2.27.1
20:22:32 requests-toolbelt 0.9.1
20:22:32 setuptools 58.1.0
20:22:32 urllib3 1.26.9
20:22:32 wheel 0.37.1
20:22:32 WARNING: You are using pip version 22.0.4; however, version 22.1.2 is available.
20:22:32 You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
20:22:32 [Pipeline] echo
20:22:32
20:22:32
20:22:32 Verify Application Files:.....
20:22:32 [Pipeline] sh
20:22:33 + ls dnacenter_jenkins/
20:22:33 cli_template.txt
20:22:33 deploy_templates.py
20:22:33 jenkinsfile
20:22:33 lab_deploy_templates.py
20:22:33 project_details.yml
20:22:33 requirements.txt
```

# Console Output – Deploy Templates to Lab

```
Dashboard ▸ CLI Templates Deployment ▸ #105

20:22:33 Downloading the CLI Templates:.....
20:22:33 [Pipeline] sh
20:22:34 + git clone https://gzapodea:****@github.com/zapodeanu/templates_jenkins.git
20:22:34 Cloning into 'templates_jenkins'...
20:22:36 [Pipeline] withEnv
20:22:36 [Pipeline] {
20:22:36 [Pipeline] echo
20:22:36
20:22:36 Verify Templates Files:.....
20:22:36 [Pipeline] sh
20:22:37 + ls templates_jenkins/
20:22:37 cli_template.txt
20:22:37 deployment_report.json
20:22:37 project_details.yml
20:22:37 [Pipeline] }
20:22:37 [Pipeline] // withEnv
20:22:37 [Pipeline] }
20:22:37 [Pipeline] // stage
20:22:38 [Pipeline] stage
20:22:38 [Pipeline] { (Deploy the CLI Templates – Lab)
20:22:38 [Pipeline] echo
20:22:38
20:22:38 Lab Deploying the CLI Templates:.....
20:22:38 [Pipeline] withEnv
20:22:38 [Pipeline] {
20:22:38 [Pipeline] echo
20:22:38
20:22:38 Verify Working Path:.....
20:22:38 [Pipeline] sh
20:22:39 + pwd
20:22:39 /Users/gzapodea/.jenkins/workspace/CLI Templates Deployment@2
20:22:39 [Pipeline] dir
20:22:39 Running in /Users/gzapodea/.jenkins/workspace/CLI Templates Deployment@2/dnacentr_jenkins
20:22:39 [Pipeline] {
20:22:39 [Pipeline] sh
20:22:40 + python3 lab_deploy_templates.py
20:22:44 INFO:root:App "lab_deploy_templates.py" Start, 2022-05-31 20:22:43
```

# Console Output – Templates to Prod + Report

```
Dashboard ▸ CLI Templates Deployment ▸ #105

20:23:37
20:23:37 Deploying the CLI Templates:.....
20:23:37 [Pipeline] withEnv
20:23:37 [Pipeline] {
20:23:37 [Pipeline] echo
20:23:37
20:23:37 Verify Working Path:.....
20:23:37 [Pipeline] sh
20:23:39 + pwd
20:23:39 /Users/gzapodea/.jenkins/workspace/CLI Templates Deployment@2
20:23:39 [Pipeline] dir
20:23:39 Running in /Users/gzapodea/.jenkins/workspace/CLI Templates Deployment@2/dnacentr_jenkins
20:23:39 [Pipeline] {
20:23:39 [Pipeline] sh
20:23:40 + python3 deploy_templates.py
20:23:43 INFO:root:App "deploy_templates.py" Start, 2022-05-31 20:23:42
20:23:43 INFO:root:Project name: Jenkins
20:23:43 INFO:root:Project id: 24c12756-46df-4de2-8f2f-c0a449eba67f
20:23:43 INFO:root:Template found and deleted
20:23:59 INFO:root:Created template with the name: cli_template
20:24:15 INFO:root:Template committed
20:24:19 INFO:root:Deploying template to device: PDX-R0
20:24:35 INFO:root:Deploying template to device: SP
20:24:50 INFO:root:Deploying template to device: NYC-R0
20:25:06 INFO:root:Deployment Report:
20:25:06 INFO:root:{"timestamp": "2022-05-31 20:23:42", "template_content": "!\\n\\ninterface loopback 500\\n description Interface - Jenkins_config\\n\\nip
access-list extended POS\\n remark Access List for POS - Jenkins_config\\n 10 permit ip any host 10.94.141.90\\n 20 permit ip any host 10.94.141.91\\n!",
"report": [{"hostname": "PDX-R0", "status": "successful"}, {"hostname": "SP", "status": "successful"}, {"hostname": "NYC-R0", "status": "successful"}]}
20:25:06 INFO:root:End of Application "deploy_templates.py" Run: 2022-05-31 20:25:04
20:25:06 [Pipeline] }
20:25:06 [Pipeline] // dir
20:25:06 [Pipeline] }
20:25:07 [Pipeline] // withEnv
20:25:07 [Pipeline] }
20:25:07 [Pipeline] // stage
20:25:07 [Pipeline] stage
20:25:07 [Pipeline] { (Upload Deployment Report)
20:25:07 [Pipeline] echo
20:25:07
```

# Jenkins Pipeline Dashboard

Dashboard > CLI Templates Deployment >

Status

</> Changes

Build Now

Configure

Delete Pipeline

Full Stage View

Open Blue Ocean

Rename

Pipeline Syntax

Pipeline CLI Templates Deployment

Automation to deploy GitHub hosted CLI templates to a list of network devices managed by Cisco D

Stage View

Average stage times:  
(Average full run time: ~4min 23s)

|                                    | Build Python Environment | Download the Project Repo | Deploy the CLI Templates - Lab | Deploy the CLI Templates - Production | Upload Deployment Report | Declarative: Post Actions |
|------------------------------------|--------------------------|---------------------------|--------------------------------|---------------------------------------|--------------------------|---------------------------|
| #112<br>Dec 08 10:16<br>No Changes | 2min 35s                 | 7s                        | 1min 9s                        | 1min 32s                              | 10s                      | 934ms                     |
| #111<br>Dec 07 17:30<br>No Changes | 1min 2s                  | 5s                        | 1min 2s                        | 1min 34s                              | 7s                       | 596ms                     |
| #110<br>Dec 07 17:02<br>No Changes | 50s                      | 5s                        | 1min 2s                        | 1min 31s                              | 8s                       | 596ms                     |
| #109<br>Dec 07 16:56<br>No Changes | 48s                      | 3s                        | 1min 15s<br>failed             | 315ms<br>failed                       | 329ms<br>failed          | 638ms                     |

Build History

trend

Filter builds...

#112 Dec 8, 2022, 10:16 AM

#111 Dec 7, 2022, 5:30 PM

#110 Dec 7, 2022, 5:02 PM

#109 Dec 7, 2022, 4:56 PM

#99 Nov 9, 2021, 8:58 AM

Atom feed for all

Atom feed for failures

- Build history
- Last job build and time for each stage
- Status for each build
- Failed stages will trigger skipping all subsequent stages

# Demo CLI Templates Pipeline

Camtasia 2022FileEditModifyTextViewExportWindowHelp

Template Editor - Cisco DNA CenterOpsView [Jenkins]dnacenter\_jenkins/project\_det

10.93.141.45/templateEditor120%

Cisco Systems, IncAPIsToolsLAB\_AccessCiscoLiveMeraki Dashboard L...Home - AsanaCisco BoxSFEH-NAS - Synolo...CDETSDNACaaPCisco.DNA.Center.D...SharePointAPIs and Owners - ...JIRA Engineering SJ...Other Bookmarks

Cisco DNA Center

Template Editor

Find template...+

Ansible\_Project

> Cloud DayN Templates

> Compliance

> Custom Profiles

> E911

> GitHub\_Project

> GitLab\_Project

> GitOps\_Project

> Jinja\_Templates

> NDB Configurations

> Onboarding Configuration

> Sample Jinja Templates

> Sample Velocity Templates

> Terraform\_Project

## Introduction

Templates are organized under a project folder structure.

Use the predefined Onboarding Configuration project to define templates for day-0 provisioning.

Click **>** **Create Project** to create a new project.  
Hover over the project, click the **⚙️** icon, and click **Add Template** to add a new template under that project.

As you work on templates, choose **Actions > Save** to save your in-progress work.  
When you're ready to commit and deploy a template, choose **Actions > Commit**.

A regular template is a single template.  
A composite sequence is a group of templates. Templates in a composite sequence are deployed in the order in which they are added.

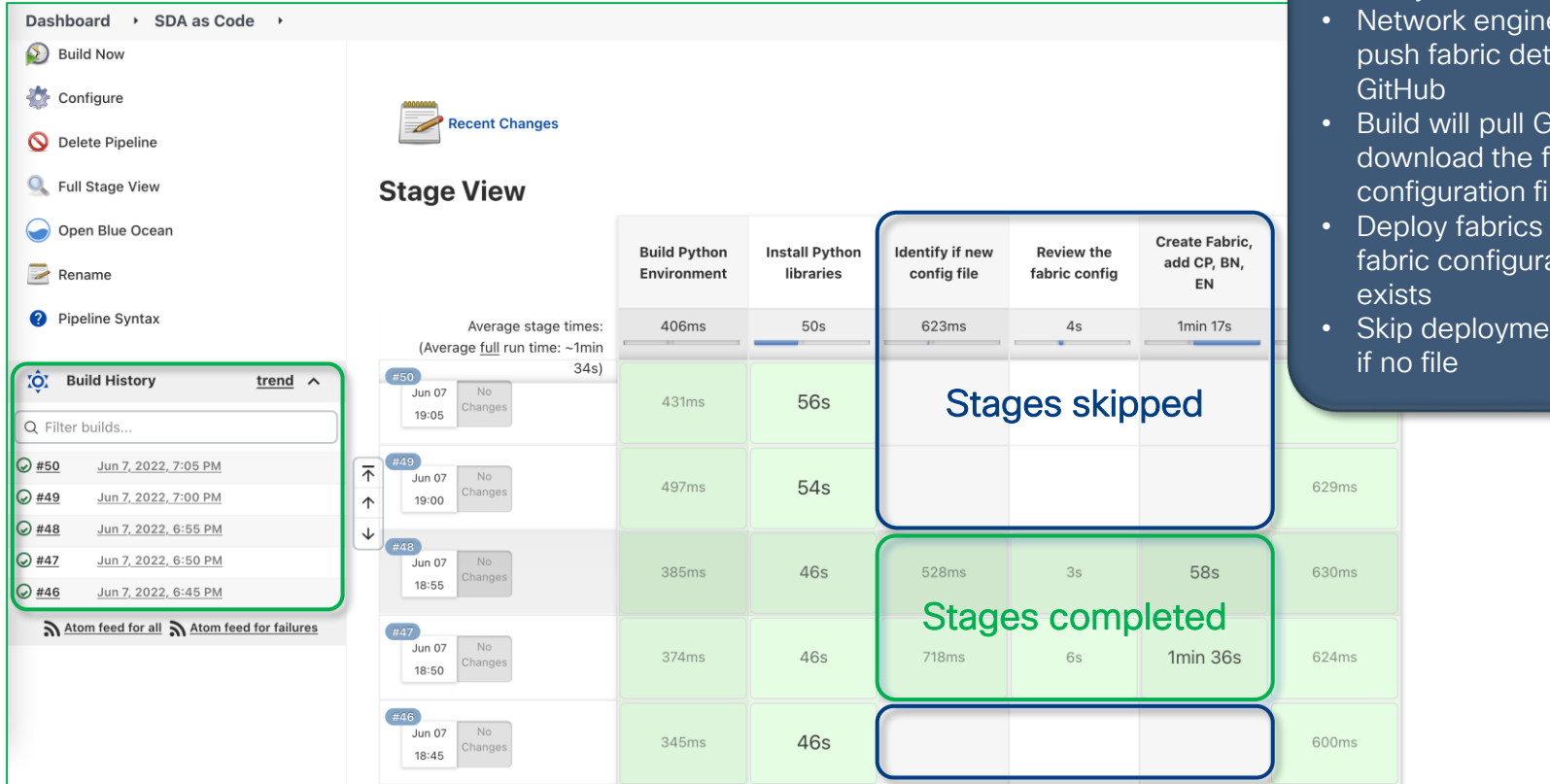
You can add committed templates to a network profile on the Design page.  
Use template tags only when you have to push different templates for the same device type based on the device tag.  
Templates added to a network profile are available for deployment as part of the provisioning workflow.  
You can only import templates that have been exported from Cisco DNA Center.

# Agenda



- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
  - How to Build a Pipeline
  - Device Inventory
  - GitHub Integration App
  - CLI Templates Deployment
  - SDA-as-Code
  - PoE Sustainability
- Developer Resources
- Summary

# Jenkins Pipeline – Create Fabric



- Runs on a schedule, every 5 minutes
- Network engineer will push fabric details to GitHub
- Build will pull GitHub to download the fabric configuration file
- Deploy fabrics only when fabric configuration file exists
- Skip deployment of fabric if no file

# Jenkins Fabric Build

Dashboard > SDA as Code >

General

Build Triggers

Advanced Project Options

Pipeline

Log the fabric config details  
when file exists

```
stage('Review the fabric config') {  
  when {  
    expression {  
      return fileExists (JENKINS_REPO_NAME + '/' + FABRIC_OPERATIONS_NAME)  
    }  
  }  
  steps {  
    echo('\n\nfabric_operations.yml:.....')  
    withEnv(["HOME=${env.WORKSPACE}"]) {  
      echo('\n\nVerify Working Path:.....')  
      sh 'pwd'  
      dir(JENKINS_REPO_NAME) {  
        sh 'cat $FABRIC_OPERATIONS_NAME'  
      }  
    }  
  }  
}
```

Run the Python App to create  
a fabric if file exists

```
stage('Create Fabric, add CP, BN, EN') {  
  when {  
    expression {  
      return fileExists (JENKINS_REPO_NAME + '/' + FABRIC_OPERATIONS_NAME)  
    }  
  }  
  steps {  
    echo('\n\nCreate a new fabric and add devices to the fabric:.....')  
    withEnv(["HOME=${env.WORKSPACE}"]) {  
      echo('\n\nVerify Working Path:.....')  
      sh 'pwd'  
      dir(JENKINS_REPO_NAME) {  
        sh 'python3 create_fabric.py'  
      }  
    }  
  }  
}
```

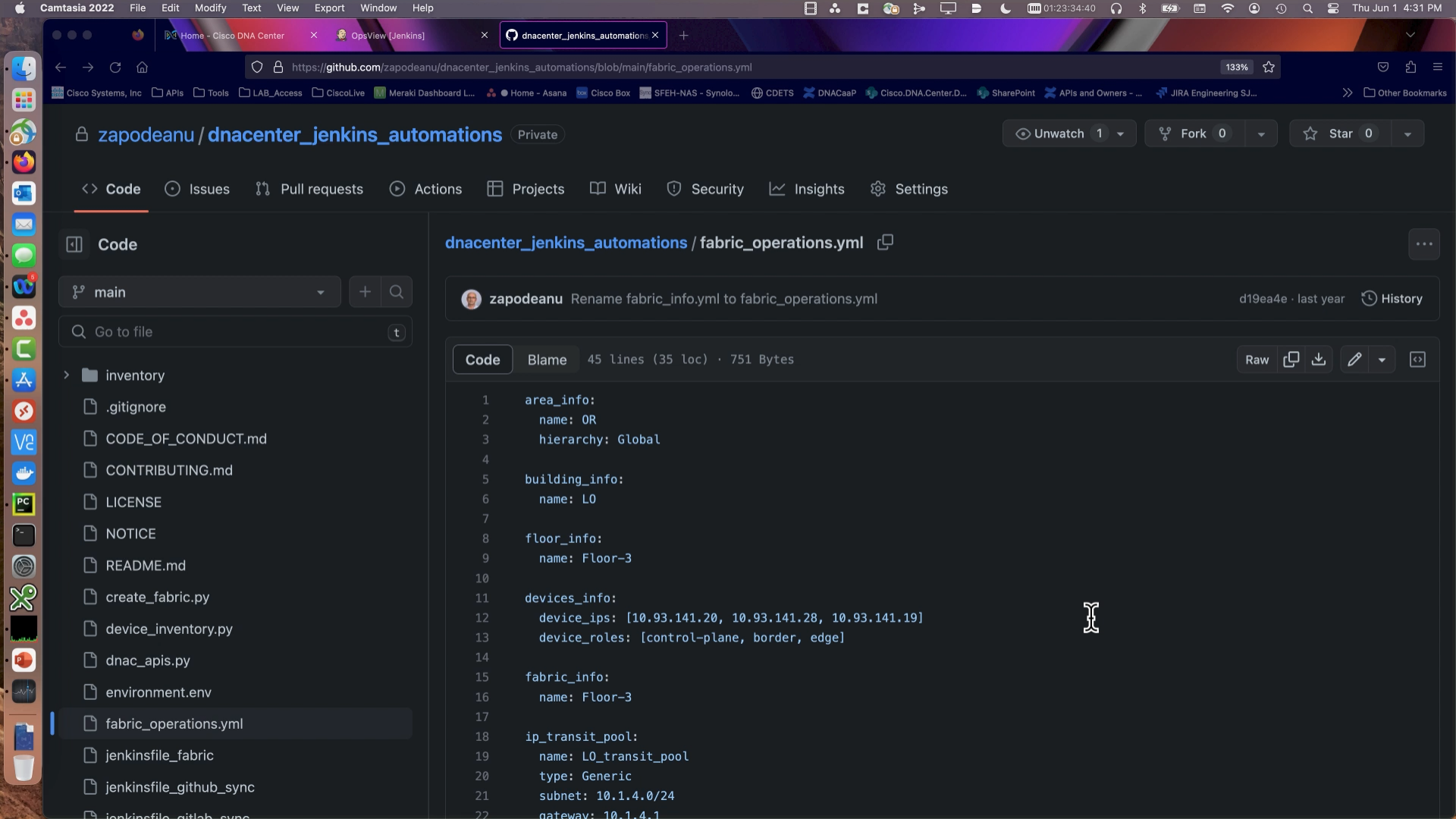
**Note:**

*Execute steps only when condition is true*

fabric\_operations.yml

```
11 devices_info:  
12   device_ips: [10.93.141.20, 10.93.141.28, 10.93.141.19]  
13   device_roles: [control-plane, border, edge]  
14  
15 fabric_info:  
16   name: Floor-3  
17  
18 ip_transit_pool:  
19   name: L0_transit_pool  
20   type: Generic  
21   subnet: 10.1.4.0/24  
22   gateway: 10.1.4.1  
23   dhcp_server: 10.93.141.46  
24   address_family: IPv4  
25  
26 l3_vn:  
27   name: Servers  
28  
29 control_plane_devices:  
30   ip: [10.93.141.20]  
31  
32 border_devices:  
33   ip: [10.93.141.28]  
34   routing_protocol: BGP  
35   internal_bgp_as: 65001  
36   external_bgp_as: 65002  
37   external_interface: TenGigabitEthernet1/1/1  
38   transit_network: IP_Transit  
39   transit_vlan: 602  
40  
41 edge_devices:  
42   ip: [10.93.141.19]
```

# Demo SDA-as-Code



# Agenda

- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
  - How to Build a Pipeline
  - Device Inventory
  - GitHub Integration App
  - CLI Templates Deployment
  - SDA-as-Code
  - PoE Sustainability
- Developer Resources
- Summary

# PoE Sustainability Pipeline

Dashboard > All >

**Enter an item name**

PoE Sustainability

» Required field

**Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used to build a project from a source code repository.

**Pipeline**  
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Multibranch Pipeline**  
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**  
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Copy from  
Jenkinsfile Template

OK

- Runs on a schedule every 12 hours, morning and evening
- Build will pull GitHub to download the PoE Automation App
- Operation UP/DOWN/NONE decided based on local time
- Disable/Enable access ports connected to APs, managed by both Cisco DNA Center and Meraki

**Note:**  
Copy from Jenkinsfile template to start building the new pipeline

REST API Jenkins 2.381

# Jenkins Pipeline – PoE Sustainability – Schedule

Dashboard > PoE Sustainability > Configuration

Configure

General

Advanced Project Options

Pipeline

Build Triggers

☐ Build after other projects are built ?

☒ Build periodically ?

Provides a [cron](#)-like feature to periodically execute this project.

This feature is primarily for using Jenkins as a cron replacement, and it is **not ideal for continuous integration**, they are often so used to the idea of regularly scheduled builds that they don't realize that the point of continuous integration is to start a build as soon as a change is made, to provide you with the most up-to-date version of your code. However, the point of continuous integration is to start a build as soon as a change is made, to provide you with the most up-to-date version of your code. That you need to [hook up SCM change notification to Jenkins](#).

So, before using this feature, stop and ask yourself if this is really what you want.

Schedule ?

H 6,18 \* \* \*

Would last have run at Friday, January 27, 2023 at 6:51:46 AM Pacific Standard Time; would next run at Friday, January 27, 2023 at 6:51:46 PM Pacific Standard Time.

This field follows the syntax of cron (with minor differences). Specifically, each line consists of 5 fields separated by TAB or whitespace:

Save

Apply

• Configure a trigger to build two times, every day:

- 6:00 AM to 6:59 AM
- 18:00 AM to 18:59 PM

# Jenkins Pipeline – PoE Sustainability – Configure

Dashboard > PoE Sustainability > Configuration

## Configure

- General
- Advanced Project Options**
- Pipeline

### Pipeline

Definition

Pipeline script

Script ?

```
1 pipeline {
2   agent
3   {
4     docker
5     {
6       reuseNode true
7       image 'python:latest'
8       args '--name DEV -v $HOME/DockerFiles:/root/DockerFiles -u root:sudo'
9     }
10  }
11
12  environment {
13    JENKINS_REPO_NAME = 'dnacenter_jenkins_automations'
14    GITHUB_USERNAME = 'gzapodea'
15    GITHUB_TOKEN = credentials('GITHUB_TOKEN')
16    PIPELINE_NAME = 'PoE Sustainability'
17    PYTHON_APP_NAME = 'poe_sustainability.py'
18    HOME = "${WORKSPACE}"
19  }
20
21  options {
22    skipStagesAfterUnstable()
23  }
24 }
```

Save Apply

- Configure the new pipeline:
  - Agent changes, if any
  - Jenkins repo name
  - Pipeline name
  - Python app name

# Jenkins Pipeline – PoE Sustainability – Output

Dashboard > PoE Sustainability > Configuration

## Configure

Definition

Pipeline script

Script ?

```
80 }
81 }
82 }
83 stage('Output folder content') {
84     steps {
85         echo("\n\nThe " + PIPELINE_NAME + " files that have been created:.....")
86         withEnv(["HOME=${env.WORKSPACE}"]) {
87             echo("\n\nVerify Working Path:.....")
88             sh 'pwd'
89             dir(JENKINS_REPO_NAME + '/inventory') {
90                 sh 'ls -l -A'
91             }
92         }
93     }
94 }
95 }
96 post {
97     always {
98         echo '\n\nJenkins ' + PIPELINE_NAME + ' build end'
99     }
100 }
101 }
102 }
```

☒ Use Groovy Sandbox ?

Pipeline Syntax

Save Apply

- Verify if the output folder is correct and you have an output stage/step
- Save

# PoE Sustainability – Build

 **Jenkins**

Dashboard > PoE Sustainability > #3

 Status

 Changes

 Console Output

 View as plain text

 Edit Build Information

 Open Blue Ocean

 Thread Dump

 Pause/resume

 Replay

 Pipeline Steps

 Workspaces

Timestamps

[View as plain text](#)

```
17:46:20 [Pipeline] {
17:46:20 [Pipeline] sh
17:46:21 + python3 poe_sustainability.py
17:46:21 INFO:root:2023-01-27 17:46:21 - App "poe_sustainability.py" Start
17:46:21 INFO:root:2023-01-27 17:46:21 - Cisco DNA Center PoE Sustainability
17:46:21 INFO:root:2023-01-27 17:46:21 - Local time hour: 17
17:46:21 INFO:root:2023-01-27 17:46:21 - Operation based on local time: DOWN
17:46:23 INFO:root:2023-01-27 17:46:23 - Collected the device list from Cisco DNA Center
17:46:23 INFO:root:2023-01-27 17:46:23 - Collected the device list from Cisco DNA Center
17:46:23 INFO:root:2023-01-27 17:46:23 - AP inventory from Cisco DNA Center:
17:46:23 INFO:root:2023-01-27 17:46:23 - AP name: AP3800i, AP Management IP Address: 10.93.141.69
17:46:23 INFO:root:2023-01-27 17:46:23 - Collected the AP inventory from Cisco DNA Center
17:46:24 INFO:root:2023-01-27 17:46:24 - PoE Operations for each AP
17:46:24 INFO:root:2023-01-27 17:46:24 - AP hostname: AP3800i
17:46:31 INFO:root:2023-01-27 17:46:30 - Saved the device inventory to file "ap_inventory.json"
17:46:31 INFO:root:2023-01-27 17:46:30 - Saved the device inventory to file "ap_inventory.yaml"
17:46:31 INFO:root:2023-01-27 17:46:30 - Meraki PoE Management
17:46:31 INFO:root:2023-01-27 17:46:30 - Meraki Switch Serial Number: Q2HP-ETU8-DKMF
17:46:31 INFO:root:2023-01-27 17:46:30 - Meraki AP Name: CAE_MR33
17:46:35 INFO:root:2023-01-27 17:46:34 - Meraki switchport admin status: False
17:46:35 INFO:root:2023-01-27 17:46:34 - End of Application "uno_cae_poe_management.py" Run
17:46:35 [Pipeline] }
17:46:35 [Pipeline] // dir
17:46:35 [Pipeline] }
17:46:35 [Pipeline] // withEnv
17:46:35 [Pipeline] }
17:46:35 [Pipeline] // stage
17:46:36 [Pipeline] stage
17:46:36 [Pipeline] { (Output folder content)
```

Manual build, to verify the new Pipeline

# PoE Sustainability Schedules

## Notes:

1. Run the same Python app anytime during the day, no need to specify the desired PoE operation
2. Protects against possible power down APs if a pipeline build would run during business hours

Application schedule logic:  
Identify if the intended PoE operation is UP, DOWN, or NONE

```
101
102     if 0 <= local_time_hour <= 7:
103         operation = 'UP'
104
105     if 18 <= local_time_hour <= 24:
106         operation = 'DOWN'
107
108     if operation == 'NONE':
109         logging.info(current_time() + ' - No PoE operation performed at this time, during business hours')
110     return
111
```

## Build Triggers

- ☐ Build after other projects are built ?
- ☒ Build periodically ?

## Schedule ?

H 6,18 \*\*\*

Would last have run at Friday, January 27, 2023 at 6:51:56 AM Pacific Standard Time; would next run at Friday, January 27, 2023 at 6:51:56 PM Pacific Standard Time.

- ☐ GitHub hook
- ☐ Poll SCM
- ☐ Quiet period
- ☐ Trigger builds remotely (e.g., from scripts) ?

Jenkins schedule:  
When will the pipeline run, daily, two times

```
INFO:root:2023-01-30 11:01:38 - App "poe_sustainability.py" Start
INFO:root:2023-01-30 11:01:38 - Cisco DNA Center PoE Sustainability
INFO:root:2023-01-30 11:01:38 - Local time hour: 11
INFO:root:2023-01-30 11:01:38 - No PoE operation performed at this time, during business hours
```

```
INFO:root:2023-01-27 18:12:33 - App "poe_sustainability.py" Start
INFO:root:2023-01-27 18:12:33 - Cisco DNA Center PoE Sustainability
INFO:root:2023-01-27 18:12:33 - Local time hour: 18
INFO:root:2023-01-27 18:12:33 - Operation based on local time: DOWN
INFO:root:2023-01-27 18:12:34 - Collected the device list from Cisco DNA Center
```

# PoE Sustainability Build – Console

```
Dashboard > PoE Sustainability > #13

18:51:42 [Pipeline] {
18:51:42 [Pipeline] echo
18:51:42
18:51:42
18:51:42 Verify Working Path:.....
18:51:42 [Pipeline] sh
18:51:42 + pwd
18:51:42 /Users/gzapodeal/.jenkins/workspace/PoE Sustainability
18:51:42 [Pipeline] dir
18:51:42 Running in /Users/gzapodeal/.jenkins/workspace/PoE Sustainability/dnacenter_jenkins_automations
18:51:42 [Pipeline] {
18:51:43 [Pipeline] sh
18:51:43 + python3 poe_sustainability.py
18:51:43 INFO:root:2023-01-31 18:51:43 - App "poe_sustainability.py" Start
18:51:43 INFO:root:2023-01-31 18:51:43 - Cisco DNA Center PoE Sustainability
18:51:43 INFO:root:2023-01-31 18:51:43 - Local time hour: 18
18:51:43 INFO:root:2023-01-31 18:51:43 - Operation based on local time: DOWN
18:51:44 INFO:root:2023-01-31 18:51:44 - Collected the device list from Cisco DNA Center
18:51:44 INFO:root:2023-01-31 18:51:44 - AP inventory from Cisco DNA Center:
18:51:44 INFO:root:2023-01-31 18:51:44 - AP name: AP3800i, AP Management IP Address: 10.93.141.69
18:51:44 INFO:root:2023-01-31 18:51:44 - Collected the AP inventory from Cisco DNA Center
18:51:45 INFO:root:2023-01-31 18:51:45 - PoE Operations for each AP
18:51:45 INFO:root:2023-01-31 18:51:45 - AP hostname: AP3800i
18:51:50 INFO:root:2023-01-31 18:51:49 - Saved the device inventory to file "ap_inventory.json"
18:51:50 INFO:root:2023-01-31 18:51:49 - Saved the device inventory to file "ap_inventory.yaml"
18:51:50 INFO:root:2023-01-31 18:51:49 - Meraki PoE Management
18:51:50 INFO:root:2023-01-31 18:51:49 - Meraki Switch Serial Number: Q2HP-ETU8-DKMF
18:51:50 INFO:root:2023-01-31 18:51:49 - Meraki AP Name: CAE_MR33
18:51:51 INFO:root:2023-01-31 18:51:51 - Meraki switchport admin status: False
18:51:51 INFO:root:2023-01-31 18:51:51 - End of Application "uno_cae_poe_management.py" Run
18:51:52 [Pipeline] }
18:51:52 [Pipeline] // dir
18:51:52 [Pipeline] }
```

App logs

# Demo PoE Sustainability

FirefoxFileEditViewHistoryBookmarksToolsWindowHelp

Home - Cisco DNA CenterDashboard [Jenkins]

localhost:8080120%★

Cisco Systems, IncAPIsToolsLAB\_AccessCiscoLiveMeraki Dashboard L...Home - AsanaCisco BoxSFEH-NAS - Synolo...CDETSDNACaaPCisco.DNA.Center.D...SharePointAPIs and Owners - ...JIRA Engineering SJ...Other Bookmarks

Jenkins

Search (⌘+K)?

2

Gabi Zapodeanu

log out

Dashboard

+ New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Open Blue Ocean

Lockable Resources

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Cisco DNA Center Pipelines using:

- REST APIs

- Python SDK

- Ansible

- Terraform

All+

| S | W | Name ↓                   | Last Success      | Last Failure | Last Duration | Fav |
|---|---|--------------------------|-------------------|--------------|---------------|-----|
| ✓ | ☀ | CLI Templates Deployment | 1 day 18 hr #113  | N/A          | 4 min 21 sec  | ▶ ☆ |
| ✓ | ☀ | Dev Ready                | 1 day 18 hr #70   | N/A          | 1 min 56 sec  | ▶ ☆ |
| ✓ | ☀ | Device Inventory         | 6 days 18 hr #54  | N/A          | 2 min 28 sec  | ▶ ☆ |
| ✓ | ☀ | Jenkins Sandbox          | 4 mo 26 days #35  | N/A          | 7.6 sec       | ▶ ☆ |
| ✓ | ☀ | Jenkinsfile Template     | 4 days 13 hr #64  | N/A          | 1 min 16 sec  | ▶ ☆ |
| ✓ | ☁ | PoE Sustainability       | 11 hr #13         | 23 hr #12    | 40 sec        | ▶ ☆ |
| ✓ | ☀ | SDA as Code              | 7 mo 25 days #234 | N/A          | 2 min 6 sec   | ▶ ☆ |

Icon: SML

Icon legend

Atom feed for all

Atom feed for failures

Atom feed for just latest builds

Edit description

# Jenkins Optimizations – Reuse and Caching

## Pipeline Dev Ready

Multi language App container



Recent Changes

## Stage View

Average stage times:

(Average full run time: ~1min 7s)

|                                   | Build Python Environment | Install Cisco DNA Center SDK | Install Ansible | Install Terraform | Verify libraries | Post Actions |
|-----------------------------------|--------------------------|------------------------------|-----------------|-------------------|------------------|--------------|
|                                   | 529ms                    | 11s                          | 28s             | 17s               | 1s               | 347ms        |
| #63<br>Nov 18 16:29<br>No Changes | 541ms                    | 8s                           | 23s             | 18s               | 1s               | 400ms        |
| #62<br>Nov 18 16:28<br>No Changes | 490ms                    | 8s                           | 23s             | 16s               | 1s               | 336ms        |
| #61<br>Nov 18 16:26<br>No Changes | 558ms                    | 18s                          | 37s             | 16s               | 1s               | 305ms        |

Pipeline script

Script ?

```
1 pipeline {
2   agent {
3     docker {
4       reuseNode true
5       image 'python:latest'
6       args '--name PAT -v $HOME/DockerFiles:/root/DockerFiles -u root:sudo'
7     }
8   }
9 }
10
11
```

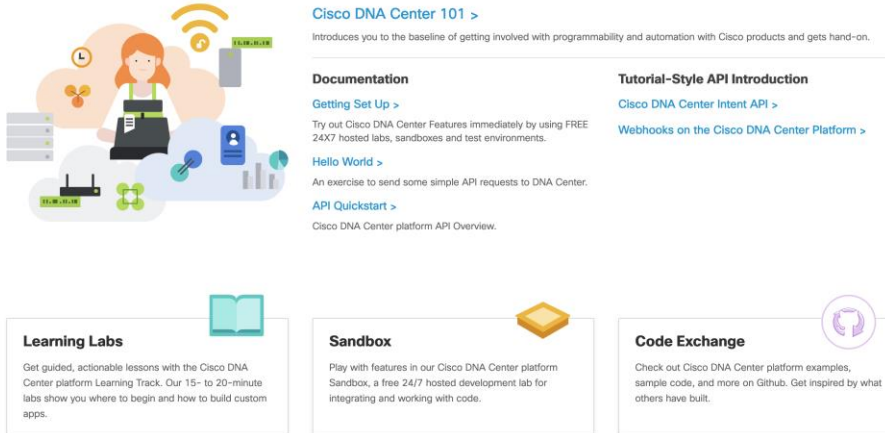
- Docker Agent “reuseNode” “true”
- Caching for all libraries and packages

- Approx. 23 seconds time savings to install Cisco DNA Center Python SDK, Ansible and Terraform when caching and reuse

# Agenda

- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
- Developer Resources
- Summary

# Cisco DNA Center – DevNet Resources



## How to Get Started?

[Cisco DNA Center 101 >](#)  
Introduces you to the baseline of getting involved with programmability and automation with Cisco products and gets hand-on.

### Documentation

[Getting Set Up >](#)  
Try out Cisco DNA Center Features immediately by using FREE 24X7 hosted labs, sandboxes and test environments.

[Hello World >](#)  
An exercise to send some simple API requests to DNA Center.

[API Quickstart >](#)  
Cisco DNA Center platform API Overview.

### Tutorial-Style API Introduction

[Cisco DNA Center Intent API >](#)  
[Webhooks on the Cisco DNA Center Platform >](#)

### 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.

- API Documentation
- User Guides
- Learning Labs
- Sandboxes
- Code Exchange

# Cisco DNA Center Jenkins Tutorial

<https://github.com/cisco-en-programmability/dnacenter-jenkins-tutorial>

The screenshot shows the GitHub repository page for 'cisco-en-programmability/dnacenter-jenkins-tutorial'. The repository is public and has 0 stars, 0 forks, and 2 watchers. The main branch is 'main' with 1 branch and 0 tags. The repository was last updated on Nov 8, 2021, with 62 commits.

The file list includes:

- `.vscode`: Configuration template deployment (8 months ago)
- `config`: Changed areas to sites (8 months ago)
- `images`: Documented the process (7 months ago)
- `templates`: Added site validation (8 months ago)
- `.DS_Store`: Documented the process (7 months ago)
- `.gitignore`: Initial commit (8 months ago)
- `Jenkinsfile`: Changed areas to sites (8 months ago)
- `LICENSE`: Initial commit (8 months ago)
- `README.md`: Fix repo link (7 months ago)
- `cli_templates.py`: Fixed errors with messages (8 months ago)
- `poetry.lock`: Included templates with variables/parms (8 months ago)

The right sidebar shows the repository's metadata:

- About**: Cisco DNA Center - Jenkins Integration tutorial
- Releases**: No releases published. [Create a new release](#)
- Packages**: No packages published. [Publish your first package](#)

# Cisco DNA Center Jenkins Automations

[https://github.com/zapodeanu/dnacenter\\_jenkins\\_automations\\_public](https://github.com/zapodeanu/dnacenter_jenkins_automations_public)

Search or jump to... Pull requests Issues Codespaces Marketplace Explore

zapodeanu / dnacenter\_jenkins\_automations\_public Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file <> Code

zapodeanu initial commit 65c6721 now 2 commits

|                     |                |                |
|---------------------|----------------|----------------|
| .gitignore          | Initial commit | 13 minutes ago |
| CODE_OF_CONDUCT.md  | initial commit | now            |
| CONTRIBUTING.md     | initial commit | now            |
| LICENSE             | initial commit | now            |
| NOTICE              | initial commit | now            |
| README.md           | initial commit | now            |
| cli_template.txt    | initial commit | now            |
| create_fabric.py    | initial commit | now            |
| deploy_templates.py | initial commit | now            |
| device_inventory.py | initial commit | now            |
| dnac_apis.py        | initial commit | now            |

About

Repo for sample Cisco DNA Center Jenkins Automations

- Readme
- View license
- Code of conduct
- 0 stars
- 1 watching
- 0 forks

Releases

No releases published  
[Create a new release](#)

Packages

No packages published

# Cisco DNA Center GitHub/GitLab Integrations

[https://github.com/cisco-en-programmability/dnacenter\\_git\\_integrations](https://github.com/cisco-en-programmability/dnacenter_git_integrations)

The screenshot shows the GitHub repository page for `cisco-en-programmability/dnacenter_git_integrations`. The repository is public and has 3 unwatchers, 0 forks, and 0 stars. The main branch is `main`, with 1 branch and 0 tags. The repository was updated with sample outputs 2 weeks ago, with 6 commits.

The file list includes:

| File                                      | Commit                      | Time        |
|-------------------------------------------|-----------------------------|-------------|
| <code>dnacenter_github_integration</code> | initial commit              | last month  |
| <code>dnacenter_gitlab_integration</code> | initial commit              | last month  |
| <code>.gitignore</code>                   | Initial commit              | last month  |
| <code>CODE_OF_CONDUCT.md</code>           | initial commit              | last month  |
| <code>CONTRIBUTING.md</code>              | initial commit              | last month  |
| <code>LICENSE</code>                      | initial commit              | last month  |
| <code>NOTICE</code>                       | initial commit              | last month  |
| <code>README.md</code>                    | updated with sample outputs | 2 weeks ago |

The right sidebar shows the repository's description: "Cisco DNA Center open-source integrations with GitHub and GitLab". It includes tags for `github`, `python`, `gitlab`, `dna-center`, and `dnacentersdk`. The sidebar also lists the README, View license, Code of conduct, 0 stars, 3 watching, 0 forks, and a link to Report repository.

# Python SDK

## Cisco DNA Center Version Support 1.2.10 to 2.3.5.x

### dnacentersdk

Simple, lightweight, scalable Python API wrapper for the DNA Center APIs

Welcome to the docs! dnacentersdk is a community developed Pythonic wrapping of the DNA Center APIs (for API versions: 1.2.10, 1.3.0, 1.3.1, 1.3.3, 2.1.1, 2.1.2, 2.2.1, 2.2.2.3, 2.2.3.3, 2.3.3.0 and 2.3.5.3). The package represents all of the Cisco DNA Center API interactions via native Python tools. Making working with the Cisco DNA Center APIs in Python a native and natural experience.

dnacentersdk helps you get things done faster. We take care of the API semantics, and you can focus on writing your code.

With dnacentersdk, you can easily:

Interact with the DNA Center APIs in an interactive Python session

Quickly create code that enables you get something done in DNA Center

Leverage the API wrapper to cleanly add DNA Center functionality to your project without having to write the boilerplate code for working with APIs

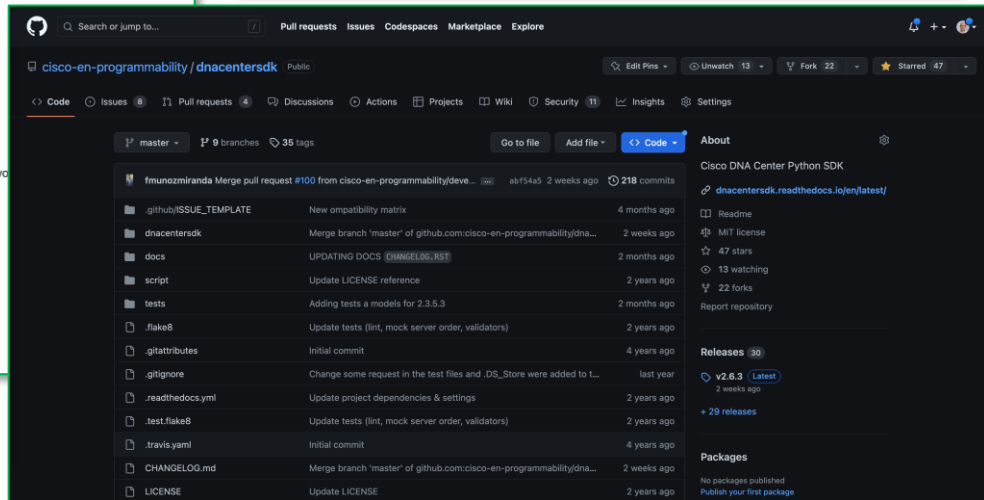
To dive in and see how dnacentersdk makes your life better, check out the [Quickstart!](#)

### The User Guide

- Installation
  - PIP Install
  - PIP Upgrade
  - Get the Source Code

<https://dnacentersdk.readthedocs.io>

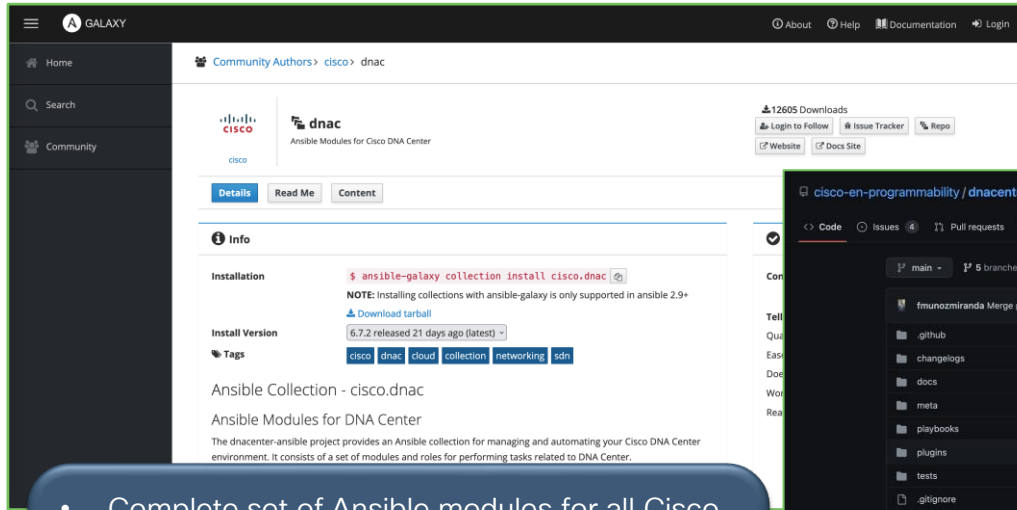
- Includes all Cisco DNA Center REST APIs
- Support for Cisco DNA Center version 2.3.5.x
- First SDK version August 2019
- Downloads last month 3,500+



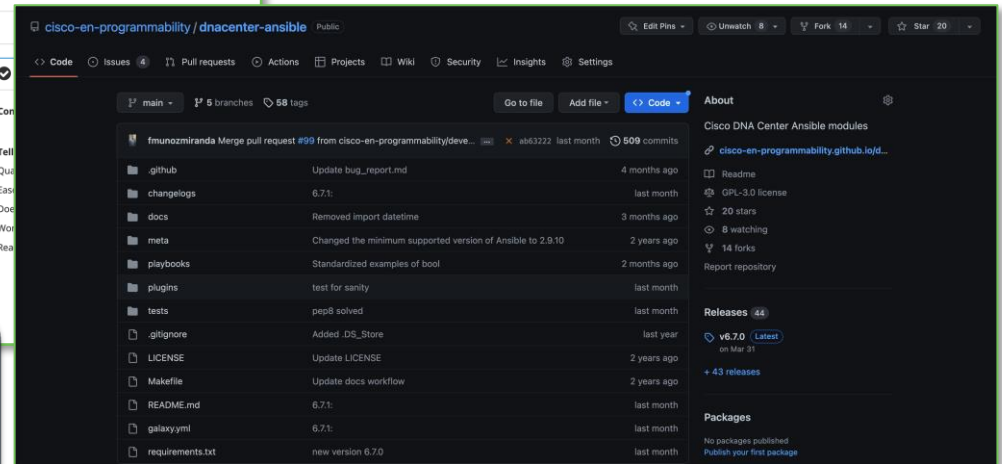
<https://github.com/cisco-en-programmability/dnacentersdk>

# Cisco DNA Center Ansible Modules

<https://galaxy.ansible.com/cisco/dnac>

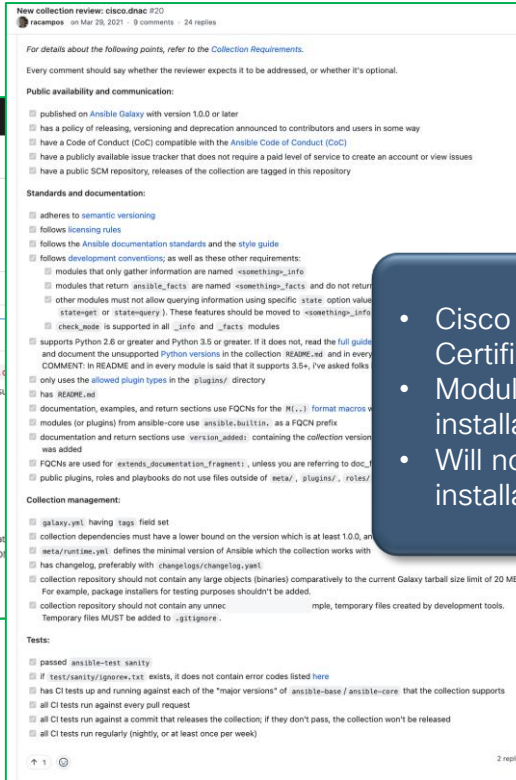
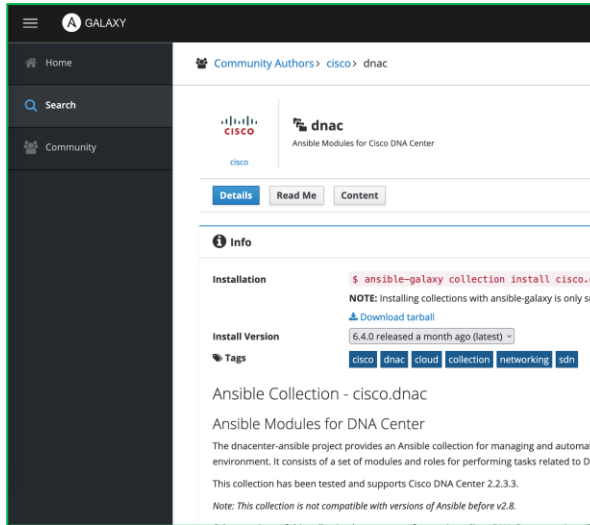


- Complete set of Ansible modules for all Cisco DNA Center REST APIs
- First library version Dec 2020
- Certified by RedHat v 2.9 – 04/2021
- Community certified – 05/2022
- Support for Cisco DNA Center version 2.3.5.3
- 12,600 + downloads



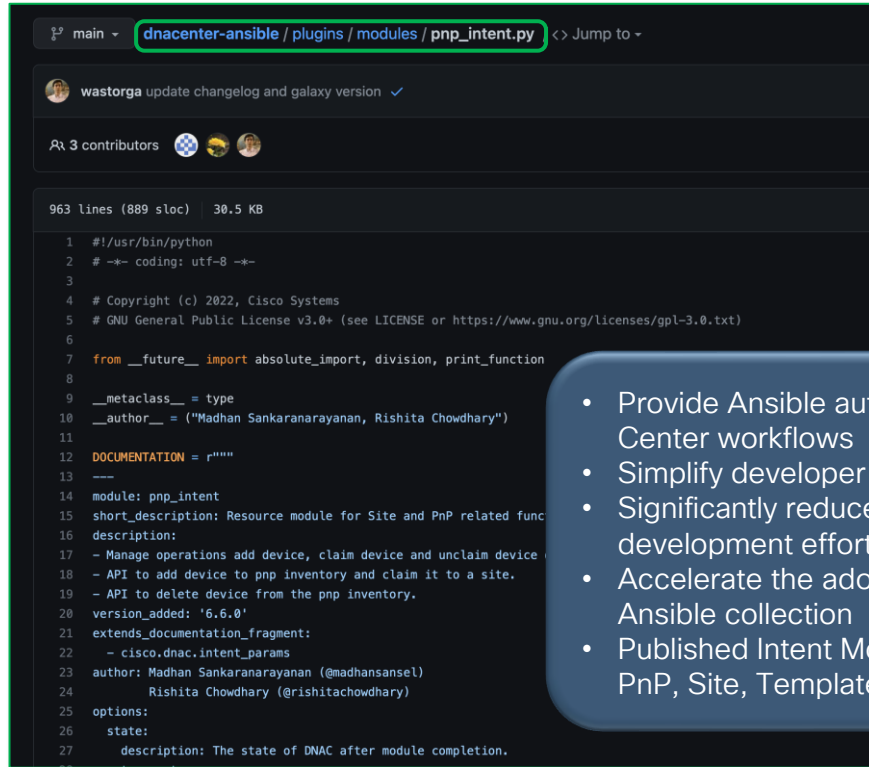
<https://github.com/cisco-en-programmability/dnacenter-ansible>

# Ansible Collection – Community Certification

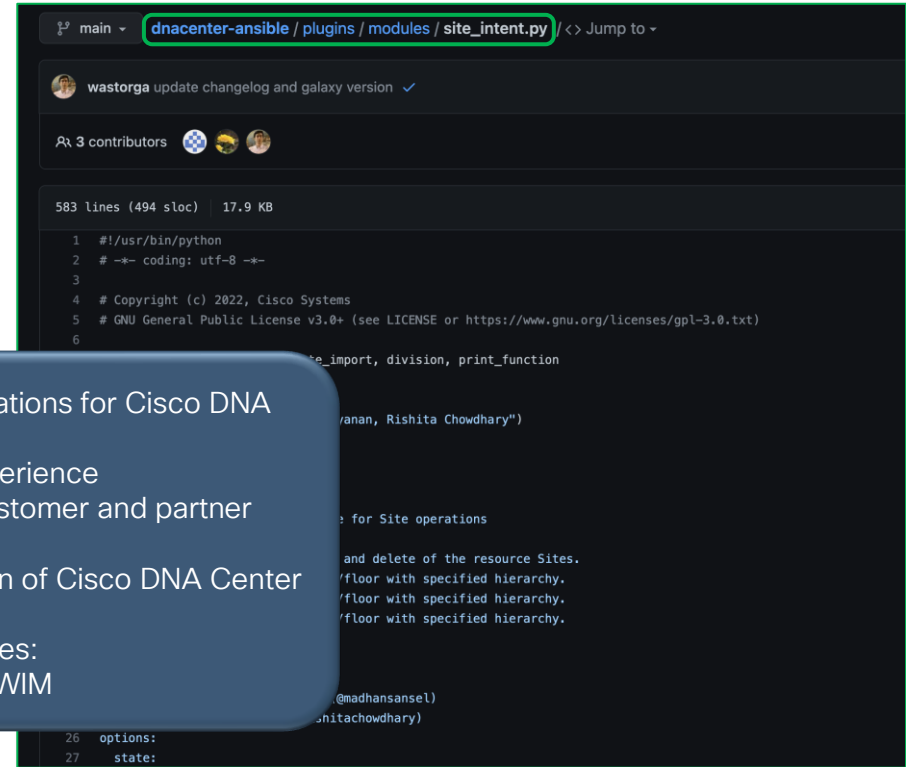


- Cisco DNA Center Collection Community Certified – 05/19/2022
- Modules are included with every Ansible installation
- Will not require additional collection installation steps

# Intent Modules – Cisco Developed and Supported



```
1 #!/usr/bin/python
2 # -*- coding: utf-8 -*-
3
4 # Copyright (c) 2022, Cisco Systems
5 # GNU General Public License v3.0+ (see LICENSE or https://www.gnu.org/licenses/gpl-3.0.txt)
6
7 from __future__ import absolute_import, division, print_function
8
9 __metaclass__ = type
10 __author__ = ("Madhan Sankaranarayanan, Rishita Chowdhary")
11
12 DOCUMENTATION = r"""
13 ---
14 module: pnp_intent
15 short_description: Resource module for Site and PnP related functions
16 description:
17 - Manage operations add device, claim device and unclaim device
18 - API to add device to pnp inventory and claim it to a site.
19 - API to delete device from the pnp inventory.
20 version_added: '6.6.0'
21 extends_documentation_fragment:
22 - cisco.dnac.intent_params
23 author: Madhan Sankaranarayanan (@madhansansel)
24       Rishita Chowdhary (@rishitachowdhary)
25 options:
26   state:
27     description: The state of DNAC after module completion.
```

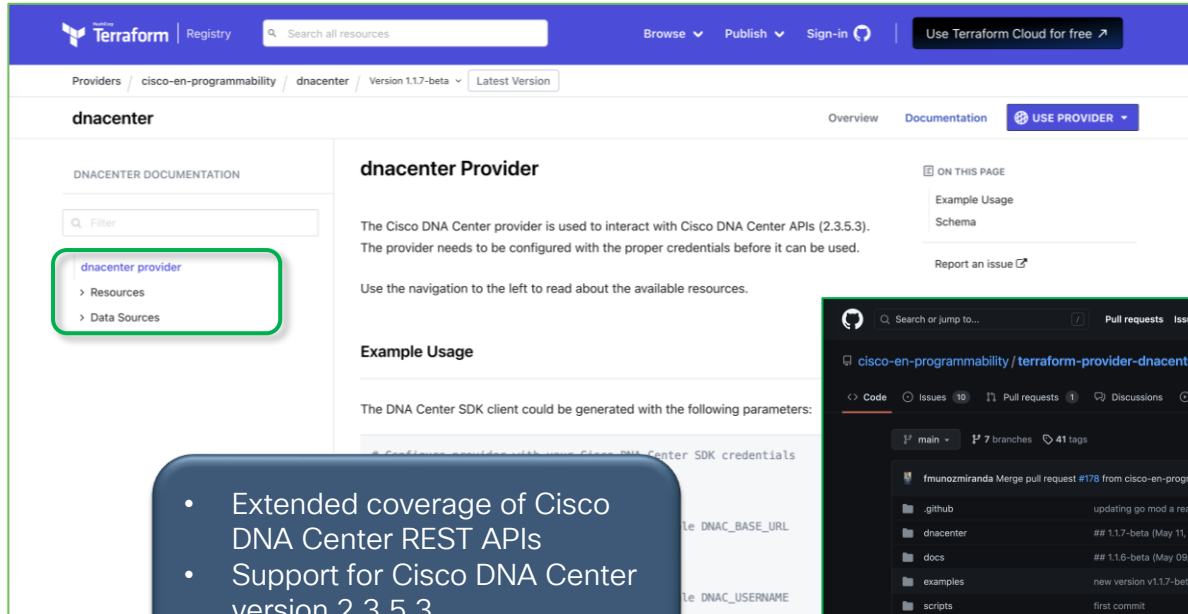


```
1 #!/usr/bin/python
2 # -*- coding: utf-8 -*-
3
4 # Copyright (c) 2022, Cisco Systems
5 # GNU General Public License v3.0+ (see LICENSE or https://www.gnu.org/licenses/gpl-3.0.txt)
6
7 from __future__ import absolute_import, division, print_function
8
9 __metaclass__ = type
10 __author__ = ("Madhan Sankaranarayanan, Rishita Chowdhary")
11
12 DOCUMENTATION = r"""
13 ---
14 module: site_intent
15 short_description: Resource module for Site operations
16 description:
17 - Manage operations add and delete of the resource Sites.
18 - API to add site to floor with specified hierarchy.
19 - API to delete site from floor with specified hierarchy.
20 version_added: '6.6.0'
21 extends_documentation_fragment:
22 - cisco.dnac.intent_params
23 author: Madhan Sankaranarayanan (@madhansansel)
24       Rishita Chowdhary (@rishitachowdhary)
25 options:
26   state:
27     description: The state of DNAC after module completion.
```

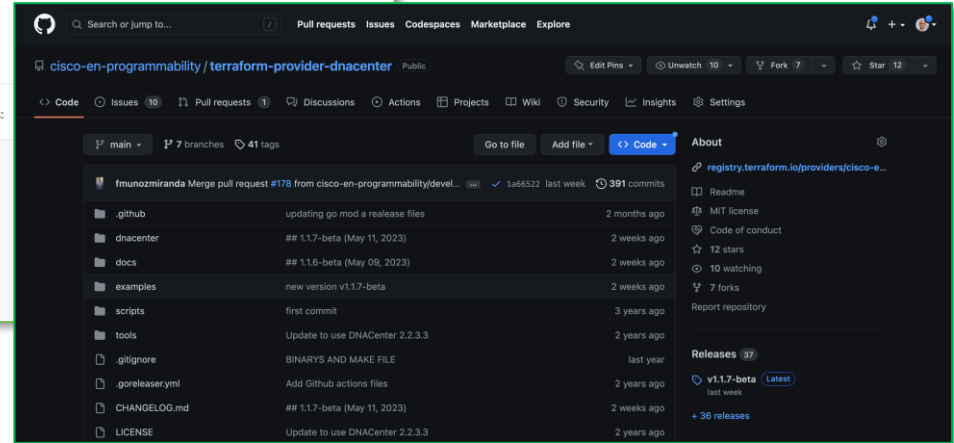
- Provide Ansible automations for Cisco DNA Center workflows
- Simplify developer experience
- Significantly reduce customer and partner development effort
- Accelerate the adoption of Cisco DNA Center Ansible collection
- Published Intent Modules: PnP, Site, Template, SWIM

# Terraform Provider and Go SDK

<https://registry.terraform.io/providers/cisco-en-programmability/dnacenter>



- Extended coverage of Cisco DNA Center REST APIs
- Support for Cisco DNA Center version 2.3.5.3
- Released January 2021
- 19,600+ downloads



<https://github.com/cisco-en-programmability/terraform-provider-dnacenter>

# Cisco DNA Center Apps – Splunkbase

<https://splunkbase.splunk.com/>

The screenshot shows the Splunkbase app marketplace interface. At the top, there's a navigation bar with 'splunkbase™', 'Collections', and 'Apps'. A search bar contains 'Find an app'. To the right are buttons for 'Submit an App' and a user profile 'GZ'. Below the navigation bar is a 'Discover Apps' section. On the left, there are filters for 'PLATFORM' (SPLUNK, SPLUNK SOAR) and 'CATEGORY'. The main content area shows 'Showing 1-2 of 2 Results for Cisco DNA/ X'. Two app cards are displayed: 'Cisco DNA Center App' and 'Cisco DNA Center Add-on', both by Gabriel Zapodeanu. The 'Cisco DNA Center App' card shows a description, platform (Splunk Enterprise, Splunk Cloud), a 5-star rating (2 reviews), and a 'DEVELOPER SUPPORTED APP' badge. The 'Cisco DNA Center Add-on' card shows a description, platform (Splunk Enterprise, Splunk Cloud), a 5-star rating (3 reviews), and a 'DEVELOPER SUPPORTED ADDON' badge. A 'Sort by' dropdown is set to 'Sort By Best Match'. A 'splunk> Certified' badge is visible on the right side of the interface.

**splunkbase™** Collections Apps

Find an app

Submit an App GZ

## Discover Apps

PLATFORM

Showing 1-2 of 2 Results for Cisco DNA/ X

Sort by  
Sort By Best Match

☐ SPLUNK

> PRODUCT

> VERSION

☐ SPLUNK SOAR

> PRODUCT

> VERSION

CATEGORY

**Cisco DNA Center App**  
By Gabriel Zapodeanu

Cisco DNA Center and Splunk have partnered to deliver an advanced security reporting and analysis tool. Th...

PLATFORM Splunk Enterprise, Splunk Cloud

RATING ★★★★★ (2)

DEVELOPER SUPPORTED APP

**Cisco DNA Center Add-on**  
By Gabriel Zapodeanu

The Cisco DNA Center Add-on for Splunk allows a Splunk® Enterprise administrator to collect data from Cisco...

PLATFORM Splunk Enterprise, Splunk Cloud

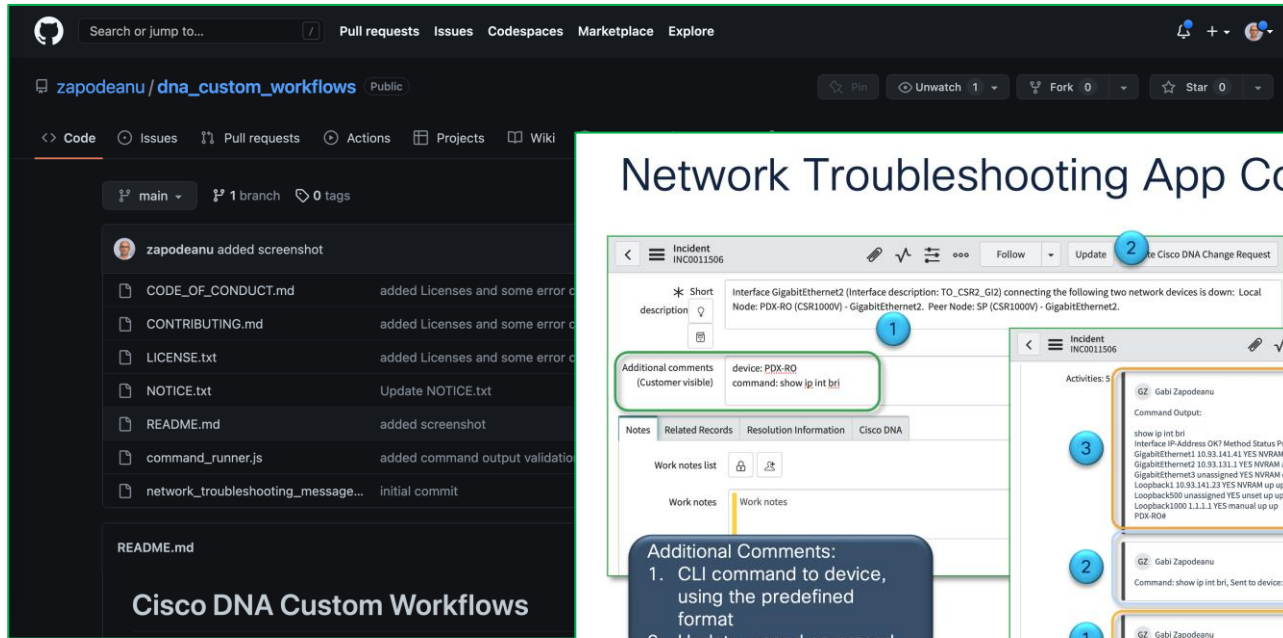
RATING ★★★★★ (3)

DEVELOPER SUPPORTED ADDON

- Two Apps:
  - Add-on – data collection
  - Visualizations
- Support via GitHub issues
- Fully customizable

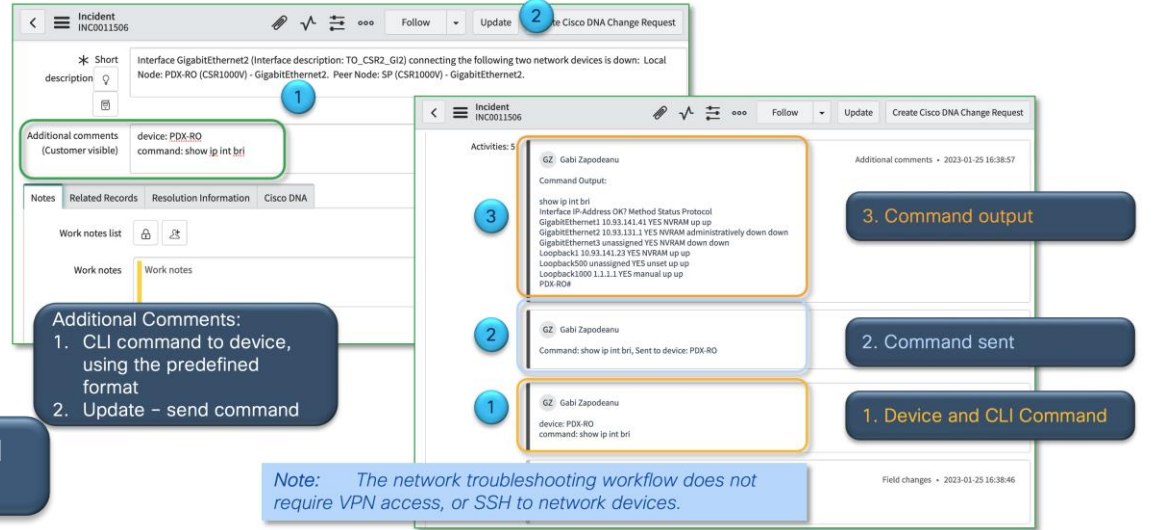
# Cisco DNA Center ServiceNow Custom Workflows

[https://github.com/zapodeanu/dna\\_custom\\_workflows](https://github.com/zapodeanu/dna_custom_workflows)

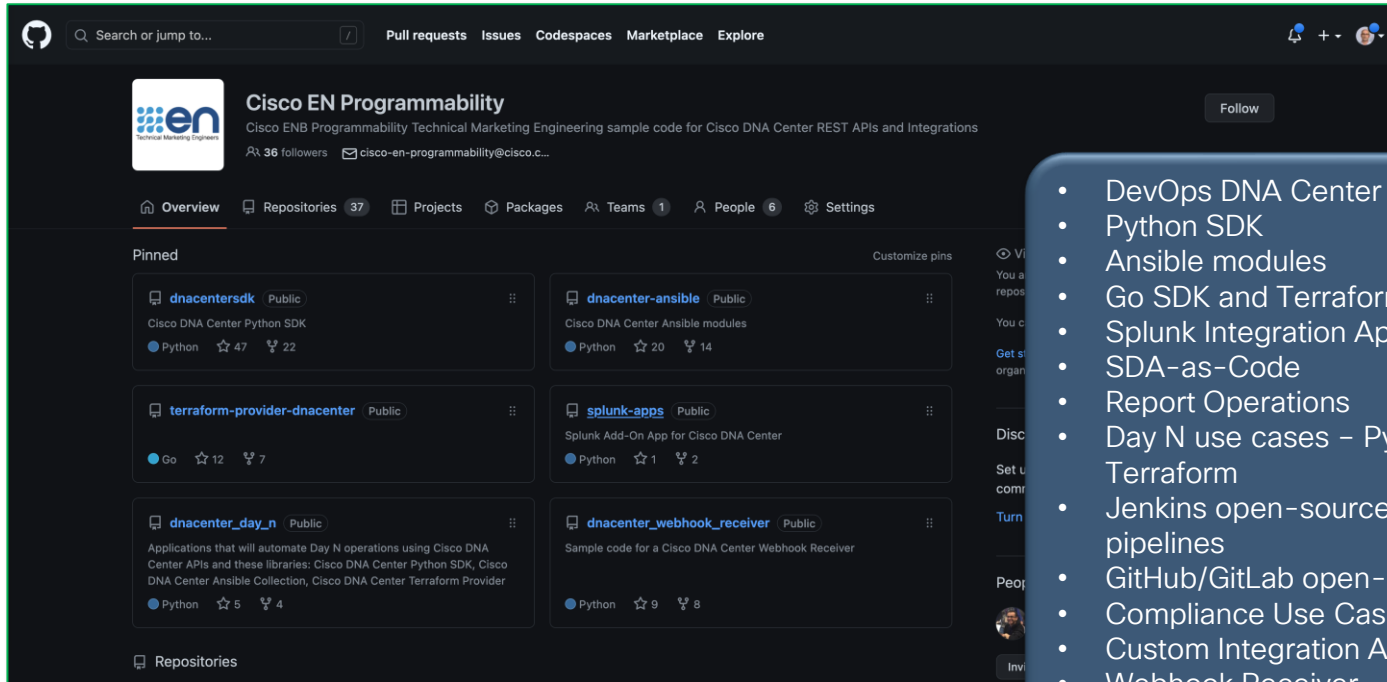


The open-source code is for demos and proof-of-concepts only.

## Network Troubleshooting App Command/Output



# Cisco EN Programmability GitHub Org



<https://github.com/cisco-en-programmability>

- DevOps DNA Center use cases
- Python SDK
- Ansible modules
- Go SDK and Terraform provider
- Splunk Integration Apps
- SDA-as-Code
- Report Operations
- Day N use cases – Python SDK, Ansible, Terraform
- Jenkins open-source Integration and pipelines
- GitHub/GitLab open-source Integrations
- Compliance Use Case
- Custom Integration App
- Webhook Receiver
- Other sample code

# YouTube Channel – EN Programmability

The screenshot displays the Cisco EN Programmability YouTube channel interface. At the top, there's a search bar and navigation links for Cisco DNA Center (DESIGN, POLICY, PROVISION, ASSURANCE, PLATFORM). The main header features the 'en' logo and 'Technical Marketing Engineers'. Below this, the channel name 'Cisco EN Programmability' is shown with 294 subscribers and buttons for 'CUSTOMIZE CHANNEL' and 'MANAGE VIDEOS'. The 'HOME' tab is selected, showing a grid of video uploads. The videos include:

- NETCONF with YANG Suite**: 68 views • 1 month ago (10:55)
- Cisco IOS XE Terraform provider introduction and...**: 117 views • 3 months ago (9:27)
- Cisco Catalyst 9300 IOS XE 17.5 GNMI GNOI OS.proto ...**: 139 views • 1 year ago (2:22)
- Cisco DNA Center Client Proximity API and Event**: 103 views • 1 year ago (13:26)
- Cisco DNA Center Real Time Notifications to Splunk...**: 2.1K views • 2 years ago (8:08)
- Cisco DNA Center IP Address Conflict Prevention**: 130 views • 2 years ago (5:57)

<https://YouTube.com/c/CiscoENProgrammability>

# Cisco DNA Center – Developer Resources



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



<https://github.com/cisco-en-programmability>



<http://cs.co/EN-Programmability-Videos>



<https://galaxy.ansible.com/cisco/dnac>



<https://registry.terraform.io/providers/cisco-en-programmability/dnacenter>



<https://dnacentersdk.readthedocs.io/en/latest/>



<https://github.com/cisco-en-programmability/dnacenter-go-sdk>

# Agenda

- Cisco DNA Center Platform
- Jenkins Open-Source Integration
- Jenkins Pipelines
- Developer Resources
- Summary

# Cisco DNA Center Jenkins Integration



```
1 configuration_templates:
2   project: Jenkins with variables
3   name: Variables testing
4   file: template_with_variables.j2
5   variables:
6     - parameterName: interface_name
7       dataType: STRING
8       required: True
9       order: 1
10    - parameterName: ip_address
11      dataType: STRING
12      required: True
13      order: 2
14   devices:
15     - name: C9K-Branch-SF0.clus-demo.com
16       variables:
17         interface_name: Loopback 200
18         ip_address: 172.30.200.1
19   product_family:
20     - Routers
21     - Switches and Hubs
22
```

REST  
APIs

- Simplified at-scale network management
- Consistent intent configurations, profiles and policies
- Use of approved intents and workflow automations
- Support for multi-Cisco DNA Center clusters and platforms
- Visibility, security and reporting

GitHub

ITSM, Splunk,  
Notifications, IPAM,  
Custom Integrations

Webhooks

# Other Cisco DNA Center Sessions:

| Day       | Time             | Room                        | Session                                                                                            |
|-----------|------------------|-----------------------------|----------------------------------------------------------------------------------------------------|
| On demand |                  |                             | BRKOPS-2854 Version Control Tools Integrations – Cisco DNA Center Infrastructure-as-Code Use Cases |
| June 5    | 9:30 – 10:30 AM  | Level 2, Mandalay Bay D     | BRKOPS-2032 3 Cisco DNA Center and ITSM Workflows: CMDB, Incident Management and SWIM              |
| June 6    | 2:30 – 3:30 PM   | Lower Level, Tradewinds ABC | BRKOPS-2471 Custom Workflows for the Cisco DNA Center Integration with ServiceNow                  |
| June 7    | 10:00 – 10:45 AM | DevNet Theater              | DEVNET-2739 ClickOps to GitOps – Cisco DNA Center Infrastructure-as-Code Use Cases                 |
| June 7    | 2:00 – 2:45 PM   | DevNet Classroom 2          | DEVNET-2151 Jenkins Automations for Cisco DNA Center                                               |

# Fill out your session surveys!



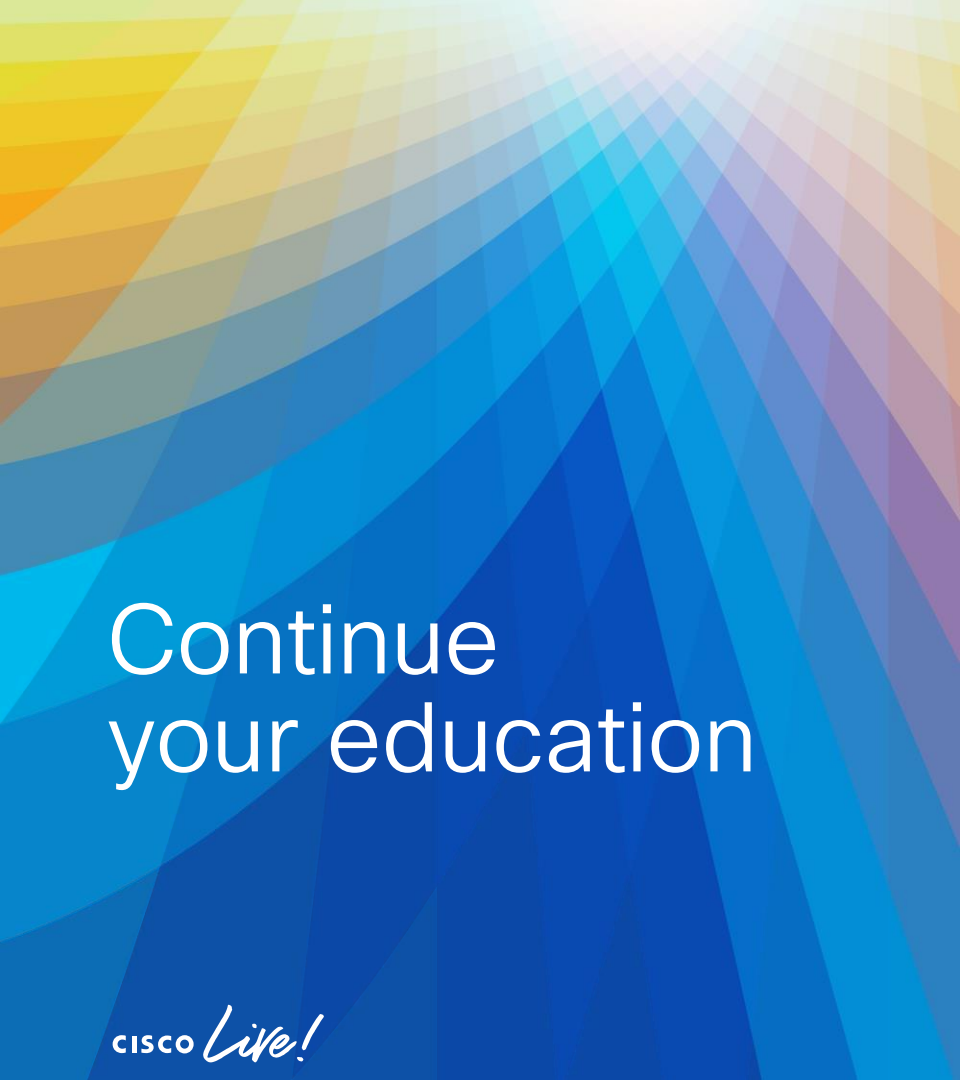
Attendees who fill out a minimum of four session surveys and the overall event survey will get **Cisco Live-branded socks** (while supplies last)!



Attendees will also earn 100 points in the **Cisco Live Game** for every survey completed.



**These points** help you get on the leaderboard and increase your chances of winning daily and grand prizes



# Continue your education

CISCO *Live!*

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Engineer meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at [www.CiscoLive.com/on-demand](https://www.CiscoLive.com/on-demand)



The bridge to possible

# Thank you

CISCO *Live!*

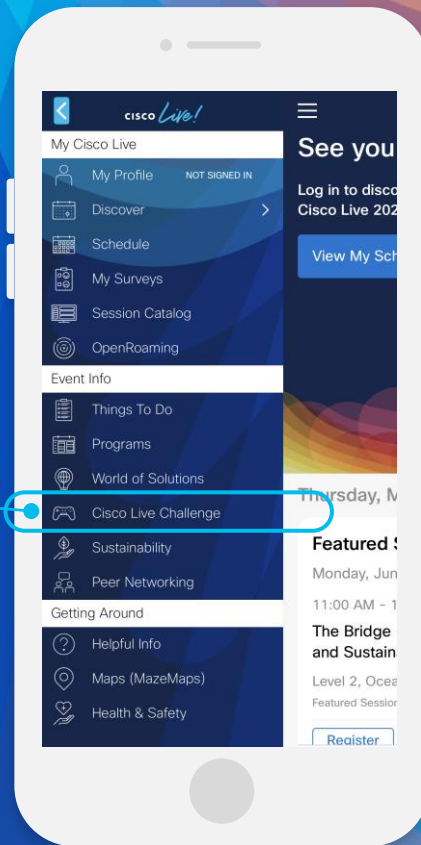
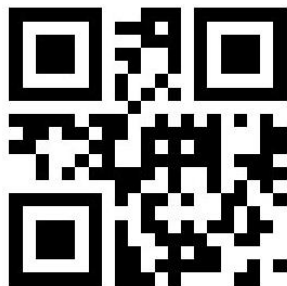
#CiscoLive

# Cisco Live Challenge

Gamify your Cisco Live experience!  
Get points for attending this session!

## How:

- 1 Open the Cisco Events App.
- 2 Click on 'Cisco Live Challenge' in the side menu.
- 3 Click on View Your Badges at the top.
- 4 Click the + at the bottom of the screen and scan the QR code:



The background is a vibrant, abstract graphic. It features a central bright white light source from which numerous colorful rays emanate, creating a sunburst or starburst effect. The rays transition through a spectrum of colors including yellow, orange, red, and various shades of blue and green. Overlaid on this are several large, semi-transparent, wavy shapes in similar color tones, giving the overall image a sense of motion and energy.

cisco *Live!*

Let's go

#CiscoLive