



The bridge to possible

# Learning YANG Data Modeling

By Playing in the NSO Playground

Jason Belk  
@renobelk  
DEVNET - 1085

CISCO *Live!*

#CiscoLive

# Cisco Webex App

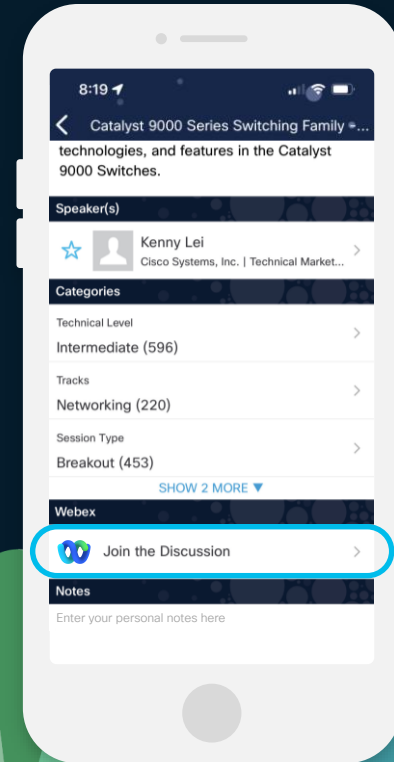
## Questions?

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 7, 2024.





# Agenda

- Structured vs Unstructured Data
- YANG Statements
- YANG Data Types
- Cisco NSO Playground Overview
- Live Demo

# CLI vs API: Audience Matters

```
router name      rno4-gw-1
router address   10.20.30.40
router operational-status up
```

Easy to **Read**, Hard to **Parse**

Hard to **Read**, Easy to **Parse**

```
<config xmlns="http://tail-f.com/ns/config/1.0">
  <router xmlns="http://com/example/routermodel">
    <name>rno4-gw-1</name>
    <address>10.20.30.40</address>
    <operational-status>up</operational-status>
  </router>
</config>
```

# Unstructured Data: Not Always Easy to Read

```
access-list 101 permit tcp any host 192.168.1.10 eq 80
```

What do these commands **mean**?

```
ip route 192.168.10.0 255.255.255.0 10.10.20.1
```

How would a machine know the **data types** and **relationships** between them?

# Structured Data: Who defines the structure?

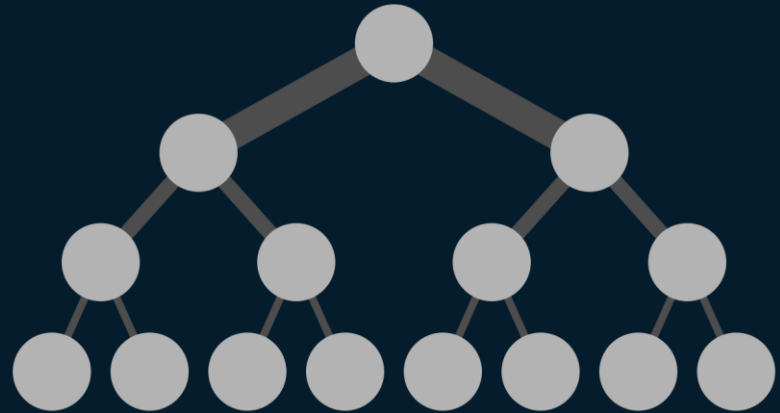
```
module router-model {  
  namespace "http://com/example/routermodel";  
  prefix router-model;  
  import ietf-inet-types {  
    prefix inet;  
  }  
  container router {  
    leaf name {  
      type string;  
    }  
    leaf address {  
      type inet:ipv4-address;  
    }  
    leaf operational-status {  
      type enumeration {  
        enum up;  
        enum down;  
      }  
    }  
  }  
}
```

```
<config xmlns="http://tail-f.com/ns/config/1.0">  
  <router xmlns="http://com/example/routermodel">  
    <name>rno4-gw-1</name>  
    <address>10.20.30.40</address>  
    <operational-status>up</operational-status>  
  </router>  
</config>
```

YANG defines the structure of the data  
(data modeling language)

# Things to Know about YANG

- Yang is a tree hierarchy
  - Start at the root (top)
- Defines the structure
- Agnostic whether the data is XML, JSON, something else
- Helps automation tooling know
  - What features are available
  - Expected input / output data types



# The Basic Building Blocks of YANG

```
module router-model {  
  namespace "http://com/example/routermode1";  
  prefix router-model;  
  import ietf-inet-types {  
    prefix inet;  
  }  
  container router {  
    leaf name {  
      type string;  
    }  
    leaf address {  
      type inet:ipv4-address;  
    }  
    leaf operational-status {  
      type enumeration {  
        enum up;  
        enum down;  
      }  
    }  
  }  
}
```

- Container statement
  - Group stuff together (no data)
- Leaf statements
  - A single unit of data
  - (Lists will be covered later)



# YANG Data Types

```
module router-model {  
  namespace "http://com/example/routermode";  
  prefix router-model;  
  import ietf-inet-types {  
    prefix inet;  
  }  
  container router {  
    leaf name {  
      type string;  
    }  
    leaf address {  
      type inet:ipv4-address;  
    }  
    leaf operational-status {  
      type enumeration {  
        enum up;  
        enum down;  
      }  
    }  
  }  
}
```

- String

- A flexible mix of numbers, letters & punctuation

- Custom Imported Types

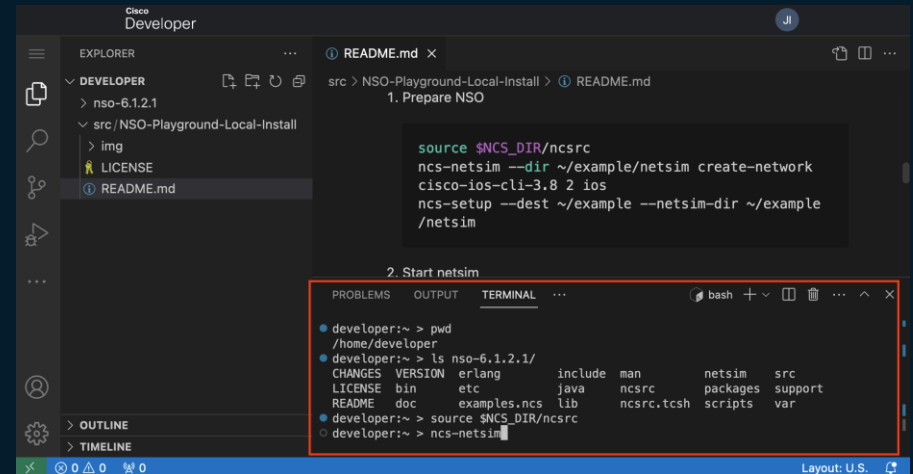
- Hidden regular expressions enforcing ipv4 syntax

- Enumeration

- Choose one from pre-defined set of values
- Has additional **enum** for each value

# The Cisco NSO Playground

- Cisco NSO uses YANG
- Using Cisco NSO to visualize YANG
- Cisco NSO Playground
  - Faster than DevNet sandbox to learn
  - Tied to specific Code Exchange Repos



# YANG Statements Visualized: GUI

```
module router-model {  
  namespace "http://com/example/routermode";  
  prefix router-model;  
  import ietf-inet-types {  
    prefix inet;  
  }  
  container router {  
    leaf name {  
      type string;  
    }  
    leaf address {  
      type inet:ipv4-address;  
    }  
    leaf operational-status {  
      type enumeration {  
        enum up;  
        enum down;  
      }  
    }  
  }  
}
```

Home /router-model:router/

name

address

operational-status ⓘ

✓ Select...

up

down

# YANG Statements Visualized: CLI

```
module router-model {  
  namespace "http://com/example/routermode1";  
  prefix router-model;  
  import ietf-inet-types {  
    prefix inet;  
  }  
  container router {  
    leaf name {  
      type string;  
    }  
    leaf address {  
      type inet:ipv4-address;  
    }  
    leaf operational-status {  
      type enumeration {  
        enum up;  
        enum down;  
      }  
    }  
  }  
}
```

admin@ncs(config)# router ?  
Possible completions:  
address name operational-status  
admin@ncs(config)# router operational-status ?  
Possible completions:  
down up  
admin@ncs(config)# router operational-status

# YANG Data Types

```
module router-model {  
  namespace "http://com/example/routermode";  
  prefix router-model;  
  import ietf-inet-types {  
    prefix inet;  
  }  
  container router {  
    leaf name {  
      type string;  
    }  
    leaf address {  
      type inet:ipv4-address;  
    }  
    leaf operational-status {  
      type enumeration {  
        enum up;  
        enum down;  
      }  
    }  
  }  
}
```

- String

- A flexible mix of numbers, letters & punctuation

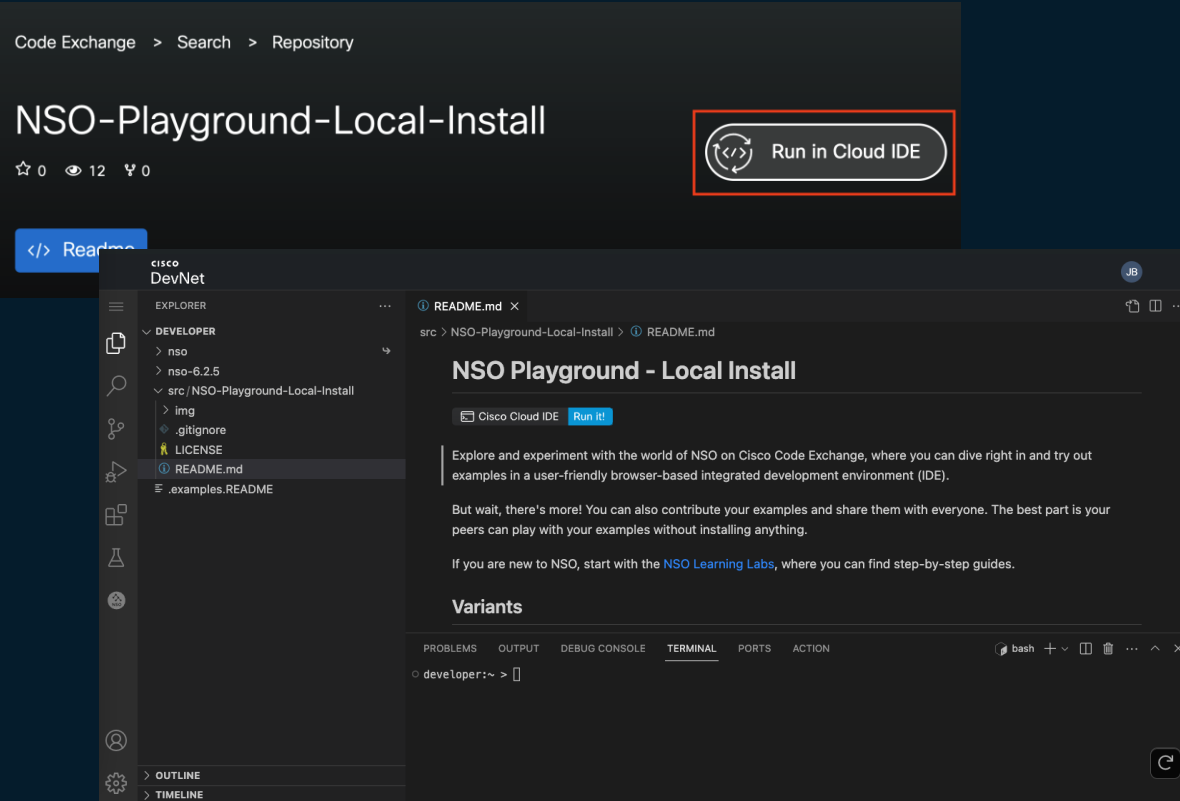
- Custom Imported Types

- Hidden regular expressions enforcing ipv4 syntax

- Enumeration

- Choose one from pre-defined set of values
- Has additional **enum** for each value

# Getting Started with NSO Playground



- Click to spin up
- Don't develop new work that isn't backed up somewhere else
- Still need to set up an instance of NSO, but it is installed in the container

# Creating an NSO Instance Steps

1. `source $NCS_DIR/ncsrc`
2. `ncs-setup --dest ~/nso-instance`
3. Copy over package directories to `~/nso-instance/packages` through VS Code GUI drag and drop
4. `cd ~/nso-instance`
5. `ncs`
6. `ncs_cli -C -u admin`

# Compile / Install YANG Packages

1. `source $NCS_DIR/ncsrc`
2. `cd ~/nso-instance/packages/learn-yang/src`
3. `make clean all`
4. `cd ~/nso-instance/packages/router-model/src`
5. `make clean all`
6. `ncs_cli -C -u admin`
7. `packages reload force`

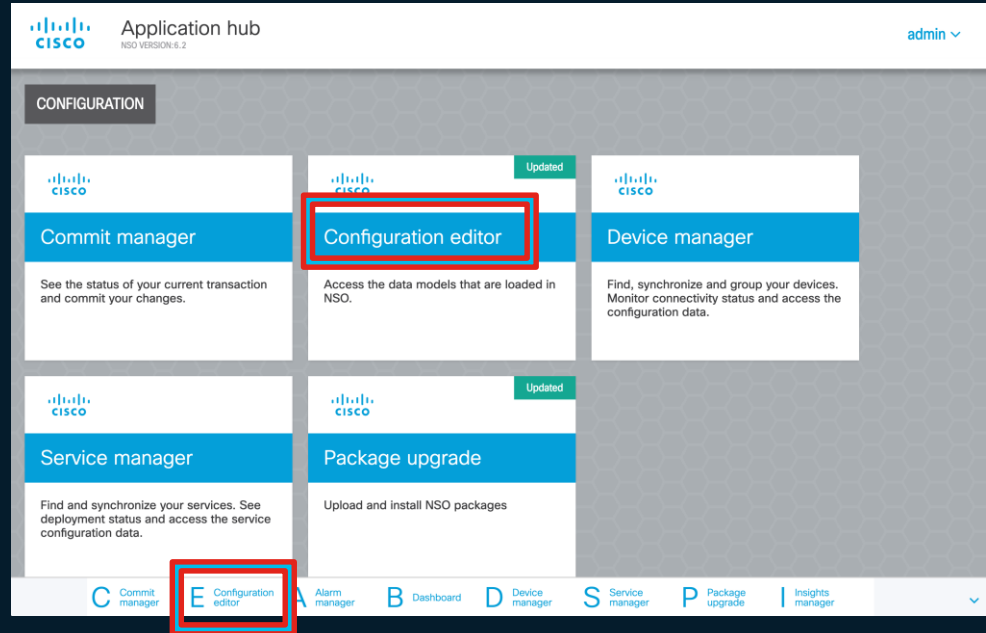
```
src$ ncs_cli -C -u admin

User admin last logged in 2024-05
om 127.0.0.1 using cli-console
admin connected from 127.0.0.1 us
admin@ncs# packages reload force
reload-result {
  package learn-yang
  result true
}
reload-result {
  package router-model
  result true
}
admin@ncs# exit
```



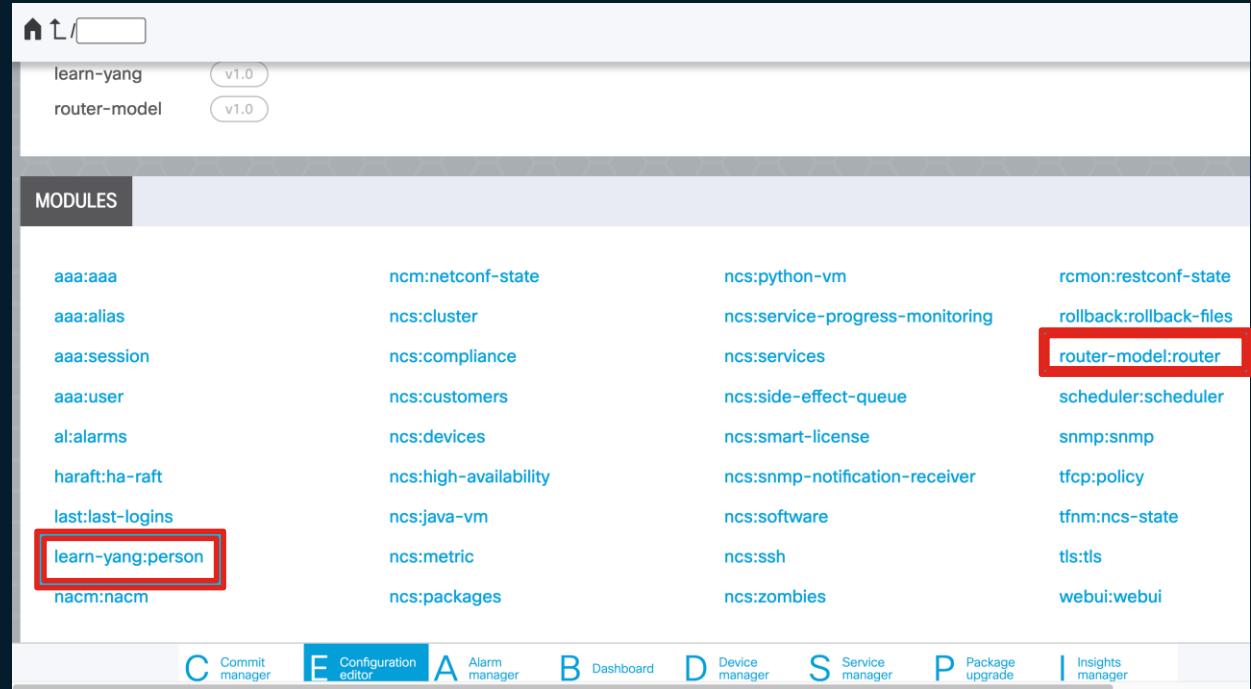
# Accessing the GUI

1. echo  
\$DEVENV\_APP\_8080\_URL
2. Open browser to above URL
3. Credentials are "admin" / "admin"
4. Click on "E" – "Configuration Editor" on the bottom left (or on the tile in middle of screen)



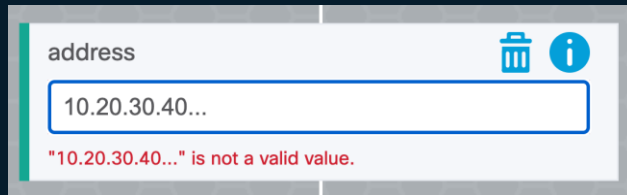
# Accessing the Packages on the GUI

- The packages have their namespace prefix (namespace:package\_name)
- Click on one of the two custom package names to edit the configuration
- There should be no values stored by default



# Adding Data

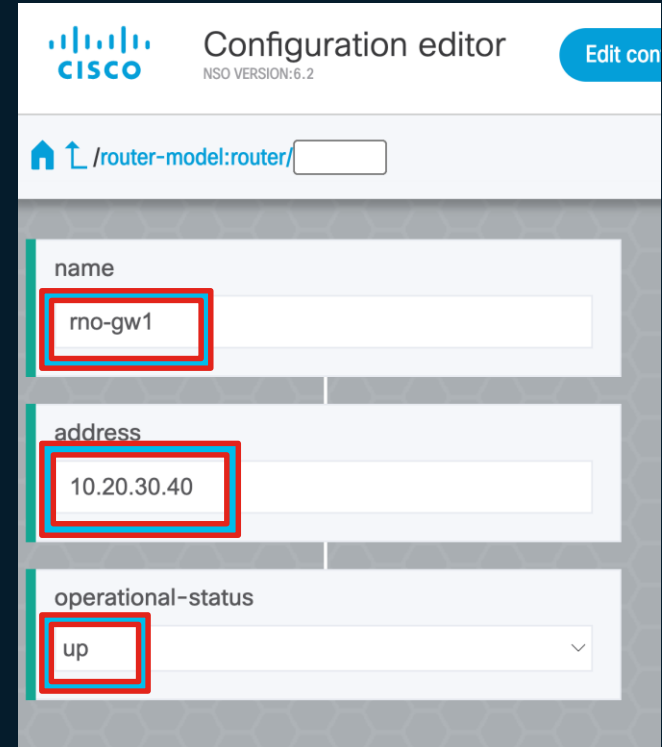
- Put a string for the name in “name”
- Put an IP address in the “address” field
- Select up or down for status
- Note:
  - Since the data model has a custom data type for IP Address validation, it will use RegEx to validate it is a valid IP address



address

10.20.30.40...

"10.20.30.40..." is not a valid value.



Configuration editor  
NSO VERSION: 6.2

/router-model:router/

name  
rno-gw1

address  
10.20.30.40

operational-status  
up

# Saving the Data (and Yes, commit) to pop-up

The screenshot displays the Cisco Commit Manager interface. At the top, the Cisco logo and 'Commit manager' title are visible, along with 'NSO VERSION:6.2'. A status bar indicates 'Current transaction (4 - webui-one) is VALID' and includes 'Revert' and 'Load/Save' buttons. A red box highlights the 'Commit' button in the top right. Below this is a table of changes with tabs for 'changes', 'errors', 'warnings', 'config', 'native config', and 'commit queue'. The table lists three configuration changes for a router model. At the bottom, a navigation bar contains icons for various tools, with a red box highlighting the 'C\*' Commit manager icon.

Commit manager  
NSO VERSION:6.2

admin

Current transaction (4 - webui-one) is VALID

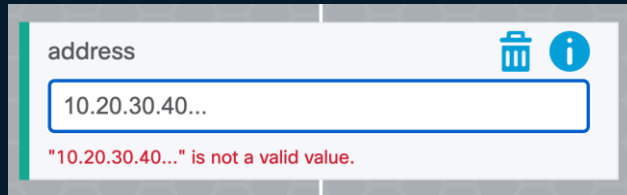
Revert Load/Save Commit

changes	errors	warnings	config	native config	commit queue
Path	Operation	Old value	New value		
/router-model:router/name	value_set		rno-gw1		
/router-model:router/address	value_set		10.20.30.40		
/router-model:router/operational-status	value_set		up		

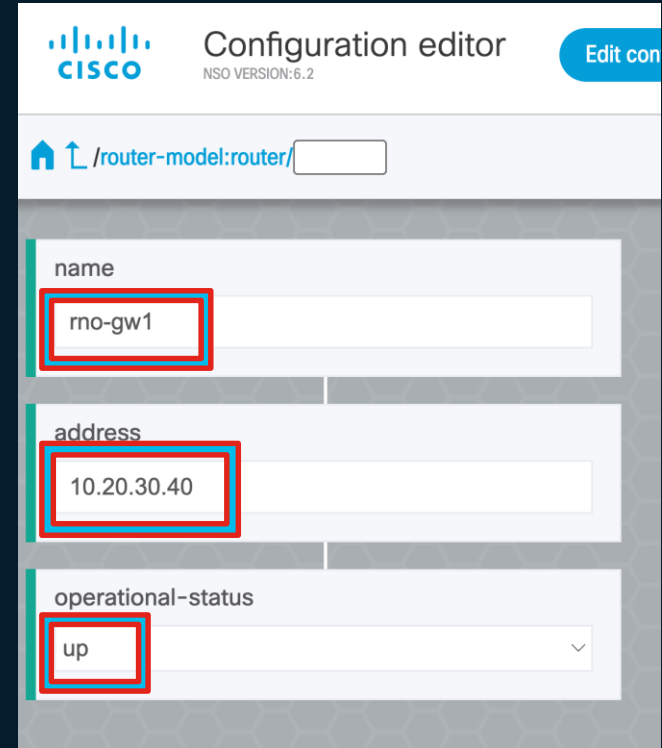
C\* Commit manager E Configuration editor A Alarm manager B Dashboard D Device manager S Service manager P Package upgrade I Insights manager

# Viewing the Data from the CLI

- NSO stores its data in a CDB, which both the GUI and CLI access
- Access the CLI from the terminal
  - Make sure to source the following if you haven't already, or it is a new terminal session
    - `source $NCS_DIR/ncsrc`



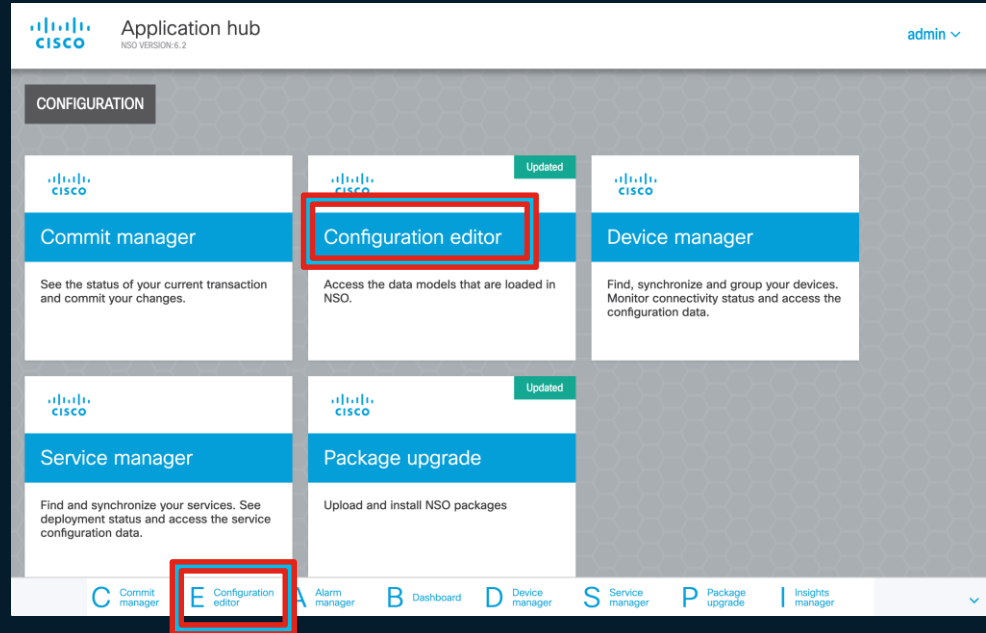
A terminal window showing a configuration field labeled 'address'. The input field contains '10.20.30.40...'. Below the input field, a red error message states: '"10.20.30.40..." is not a valid value.' To the right of the input field are icons for a trash can and an information symbol.



The screenshot shows the Cisco NSO Configuration editor interface. At the top, it says 'Configuration editor' and 'NSO VERSION: 6.2'. Below this is a breadcrumb navigation path: '/router-model:router/'. The main area displays three configuration fields, each with a red box around its value: 'name' with 'rno-gw1', 'address' with '10.20.30.40', and 'operational-status' with 'up'.

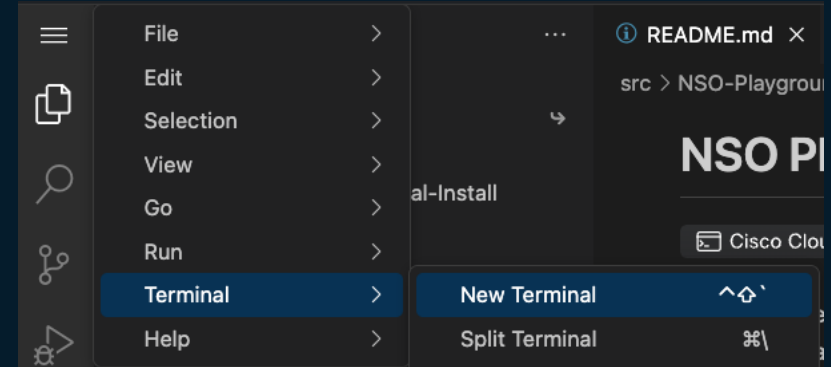
# Accessing the CLI

1. `echo $DEVENV_APP_8080_URL`
2. Open browser to above URL
3. Credentials are "admin" / "admin"
4. Click on "E" – "Configuration Editor" on the bottom left (or on the tile in middle of screen)



# Accessing the CLI in the Terminal in the Browser

1. Use the terminal on the bottom of the screen
2. If it is not there open a new one on the left-hand navigation



# Accessing the CLI in the Terminal in the Browser

1. `source $NCS_DIR/ncsrc`
2. `ncs_cli -C -u admin`
3. `show running-config router`
4. `show running-config router | display json`
5. `show running-config router | display xml`

```
admin@ncs# show running-config router
router name      rno-gw1
router address   10.20.30.40
router operational-status up
admin@ncs# show running-config router | display json
{
  "data": {
    "router-model:router": {
      "name": "rno-gw1",
      "address": "10.20.30.40",
      "operational-status": "up"
    }
  }
}
admin@ncs# show running-config router | display xml
<config xmlns="http://tail-f.com/ns/config/1.0">
  <router xmlns="http://com/example/routermodel">
    <name>rno-gw1</name>
    <address>10.20.30.40</address>
    <operational-status>up</operational-status>
  </router>
</config>
admin@ncs#
```



# Change the Config & Try out the ? options

1. conf
2. router ?
3. router name ?

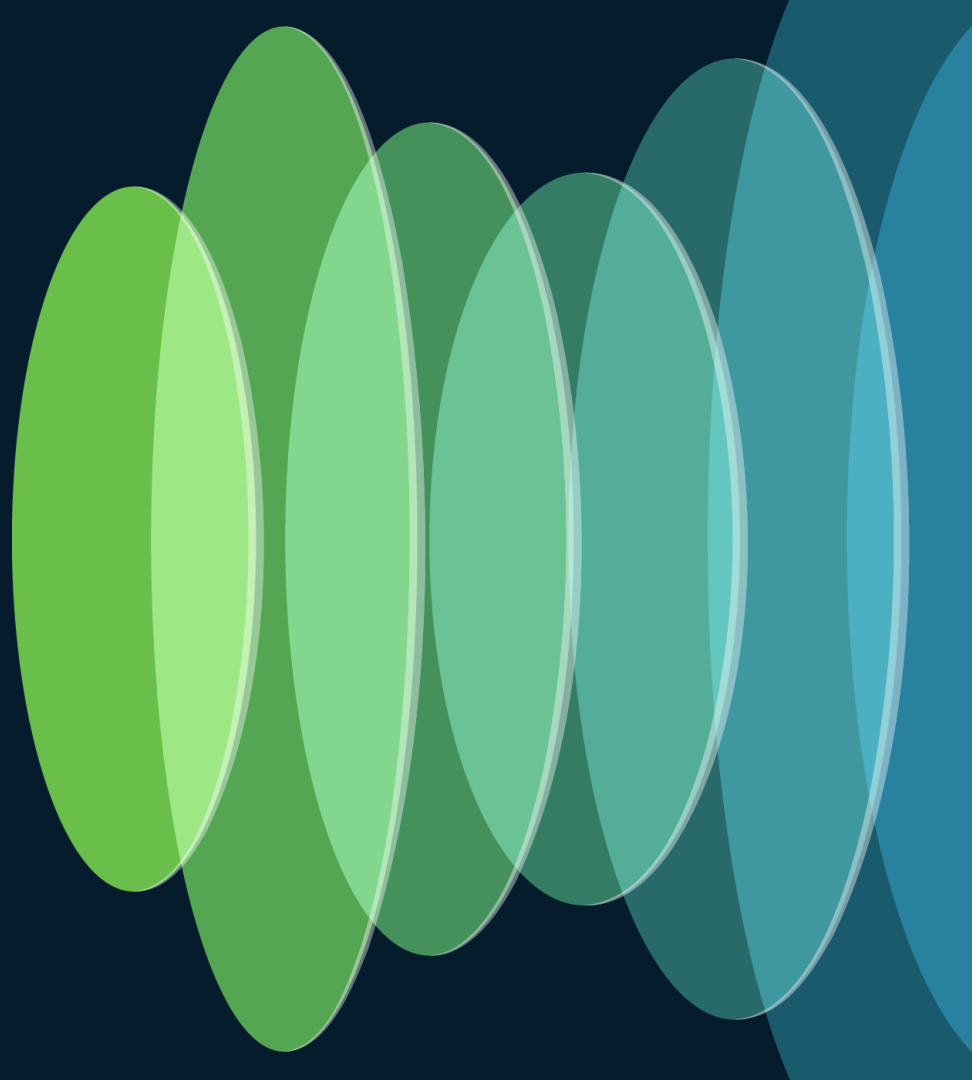
```
admin@ncs# conf
Entering configuration mode terminal
admin@ncs(config)# router ?
Possible completions:
  address  name  operational-status
admin@ncs(config)# router name ?
Possible completions:
  <string>[rno-gw1]
admin@ncs(config)# router name
```

# Change the Config & Try out the ? options

1. router name sjc-gw2
2. commit
3. end
4. show running-config router

```
admin@ncs# conf
Entering configuration mode terminal
admin@ncs(config)# router ?
Possible completions:
  address  name  operational-status
admin@ncs(config)# router name ?
Possible completions:
  <string>[rno-gw1]
admin@ncs(config)# router name sjc-gw2
admin@ncs(config)# commit
Commit complete.
admin@ncs(config)# end
admin@ncs# show running-config router
router name          sjc-gw2
router address       10.20.30.40
router operational-status up
admin@ncs#
```

# Live Demo



# Complete Your Session Evaluations



Complete a minimum of 4 session surveys and the Overall Event Survey to be entered in a drawing to **win 1 of 5 full conference passes** to Cisco Live 2025.

---



**Earn 100 points** per survey completed and compete on the Cisco Live Challenge leaderboard.

---



Level up and earn **exclusive prizes!**

---



Complete your surveys in the **Cisco Live mobile app**.

# Continue your education

- 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)

Contact me at: renobelk (Twitter / LinkedIn)

# Continue your learning journey after the event with Cisco DevNet!

- Personalized learning modules
- Live interactive tooling
- Other resources



**Scan to get started**



The bridge to possible

# Thank you

CISCO *Live!*

#CiscoLive