



NX-API Developer Sandbox

This chapter contains the following topics:

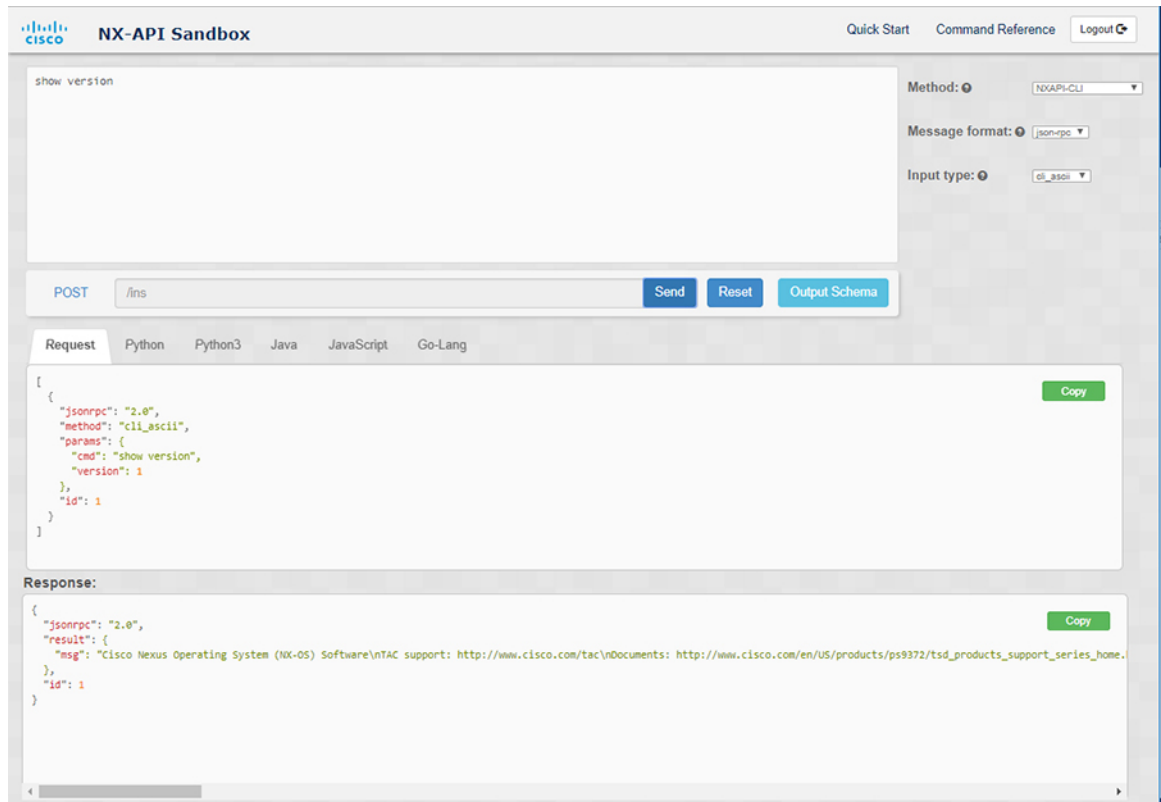
- [About the NX-API Developer Sandbox, on page 1](#)
- [Guidelines and Limitations, on page 2](#)
- [Enabling and Accessing the Developer Sandbox, on page 3](#)
- [Configuring the Message Format and Input Type, on page 3](#)
- [Using the Developer Sandbox, on page 5](#)

About the NX-API Developer Sandbox

The NX-API Developer Sandbox is a web form hosted on the switch. It translates NX-OS CLI commands into equivalent XML or JSON payloads, and converts NX-API REST payloads into their CLI equivalents.

The web form is a single screen with three panes — Command (top pane), Request, and Response — as shown in the figure.

Figure 1: NX-API Developer Sandbox with Example Request and Output Response



Controls in the Command pane allow you to choose a message format for a supported API, such as NX-API REST, and a command type, such as XML or JSON. The available command type options vary depending on the selected message format.

When you type or paste one or more CLI commands into the Command pane, the web form converts the commands into an API payload, checking for configuration errors, and displays the resulting payload in the Request pane. If you then choose to post the payload directly from the Sandbox to the switch, using the POST button in the Command pane, the Response pane displays the API response.

Conversely, when you type an NX-API REST designated name (DN) and payload into the Command pane and select the **NX-API-REST (DME)** Method format and the **model** Input type, Developer Sandbox checks the payload for configuration errors, then the Response pane displays the equivalent CLIs.

Guidelines and Limitations

Following are the guidelines and limitations for the Developer Sandbox:

- Clicking **Send** in the Sandbox commits the command to the switch, which can result in a configuration or state change.
- Some feature configuration commands are not available until their associated feature has been enabled.
- Using Sandbox to convert with DN is supported only for finding the DN of a CLI config. Any other workflow, for example, using DME to convert DN for CLI configuration commands is not supported.

- The Command pane (the top pane) supports a maximum of 10,000 individual lines of input.

Enabling and Accessing the Developer Sandbox

By default, NX-API is disabled on the switch, as is the Sandbox. This procedure shows you how to enable both and how to access the Sandbox with a browser.

Before you begin

Configure the management port of the switch.

Procedure

-
- Step 1** Enable the NX-API **nxapi** feature and the sandbox using the following commands.
- ```
switch# configure terminal
switch(config)# feature nxapi
```
- Step 2** (Optional) To change the HTTP or HTTPS port number, enter the **nxapi http[s] port port-number** command.
- Example:**
- ```
switch# configure terminal
switch(config)# nxapi http port 80
switch(config)# nxapi https port 443
```
- Step 3** Open a browser and enter **http[s]://management-ip-address** (or **http[s]://management-ip-address:port-number**, if you configured with a specific port number in the previous step) to launch the NX-API Developer Sandbox.
- Example:**
- If the management IP address of the switch is 192.0.20.123, browse to `https://192.0.20.123`.
- Note** Cisco recommends that you use the Chrome browser, release 69.0.3497.100 (64-bit), or later, to access the NX-API Developer Sandbox.
-

Configuring the Message Format and Input Type

The **Method**, **Message format**, and **Input type** are configured in the upper right corner of the Command pane (the top pane). For **Method**, choose the format of the API protocol that you want to use. The Developer Sandbox supports the following API protocols:

Table 1: NX-OS API Protocols

Protocol	Description
NXAPI-CLI	Cisco NX-API proprietary protocol for delivering NX-OS CLI or bash commands in an XML or a JSON payload.

Protocol	Description
NXAPI-REST (DME)	Cisco NX-API proprietary protocol for manipulating and reading managed objects (MOs) and their properties in the internal NX-OS data management engine (DME) model. For more information about the Cisco Nexus 3400-S Series NX-API REST SDK, see Cisco Nexus NX-API References .
RESTCONF (Yang)	The YANG ("Yet Another Next Generation") data modeling language for configuration and state data.

When the **Method** has been chosen, a set of **Message format** and **Input type** options are presented. The **Message format** setting can constrain the input CLI and can determine the **Request** and **Response** format. The options vary depending on the **Method** selection.

For each **Message format**, the following table describes the **Input type** options:

Table 2: Command Types

Method	Message format	Input type
NXAPI-CLI	json-rpc	<ul style="list-style-type: none"> cli — show or configuration commands cli-ascii — show or configuration commands, output without formatting cli-array — show or configuration commands. Similar to cli, but with cli_array, data is returned as a list of one element, or an array, within square brackets, [].
NXAPI-CLI	xml	<ul style="list-style-type: none"> cli_show — show commands. If the command does not support XML output, an error message will be returned. cli_show_array — show commands. Similar to cli_show, but with cli_show_array, data is returned as a list of one element, or an array, within square brackets []. cli_show_ascii — show commands, output without formatting cli_conf — configuration commands. Interactive configuration commands are not supported. bash — bash commands. Most non-interactive bash commands are supported. <p>Note The bash shell must be enabled in the switch.</p>

Method	Message format	Input type
NXAPI-CLI	json	<ul style="list-style-type: none"> cli_show — show commands. If the command does not support XML output, an error message will be returned. cli_show_ascii — show commands, output without formatting cli_conf — configuration commands. Interactive configuration commands are not supported. bash — bash commands. Most non-interactive bash commands are supported. <p>Note The bash shell must be enabled in the switch.</p>
NXAPI-REST (DME)		<ul style="list-style-type: none"> cli — configuration commands model — DN and corresponding payload.
RESTCONF (Yang)	<ul style="list-style-type: none"> json — JSON structure is used for payload xml — XML structure is used for payload 	

Output Chunking

In order to handle large show command output, some NX-API message formats support output chunking for show commands. In this case, an **Enable chunk mode** checkbox appears below the **Command Type** control along with a session ID (**SID**) type-in box.

When chunking is enabled, the response is sent in multiple "chunks," with the first chunk sent in the immediate command response. In order to retrieve the next chunk of the response message, you must send an NX-API request with **SID** set to the session ID of the previous response message.

Using the Developer Sandbox

You can use the Developer Sandbox to make multiple conversions, including the following:

- [Using the Developer Sandbox to Convert CLI Commands to Payloads, on page 6](#)
- [Using the Developer Sandbox to Convert from REST Payloads to CLI Commands, on page 9](#)
- [Using the Developer Sandbox to Convert from RESTCONF to json or XML, on page 13](#)

Using the Developer Sandbox to Convert CLI Commands to Payloads



Tip Online help is available by clicking **Quick Start** in the upper right corner of the Sandbox window. Additional details, such as response codes and security methods, can be found in the chapter "NX-API CLI". Only configuration commands are supported.

Procedure

- Step 1** Configure the **Method**, **Message Format**, and **Input Type** for the API protocol you want to use. For detailed instructions, see [Configuring the Message Format and Input Type, on page 3](#).
- Step 2** Type or paste NX-OS CLI configuration commands, one command per line, into the text entry box in the top pane. You can erase the contents of the text entry box (and the **Request** and **Response** panes) by clicking **Reset** at the bottom of the top pane.

- Step 3** Click the **Convert** at the bottom of the top pane. If the CLI commands contain no configuration errors, the payload appears in the **Request** pane. If errors are present, a descriptive error message appears in the **Response** pane.

The screenshot displays the NX-API Sandbox interface. At the top, there is a navigation bar with the Cisco logo, the title "NX-API Sandbox", and links for "Quick Start", "DME Documentation", "Model Browser", and "Logout". The main area is divided into several sections:

- Code Editor:** A text area containing a JSON payload:

```
{  "topSystem": {    "attributes": {      "name": "REST2CLI"    }  }}
```
- Method and Input Type:** A "Method" dropdown menu is set to "NXAPI-REST (DME)" and an "Input type" dropdown menu is set to "model".
- URL and Action Buttons:** A text input field contains the URL "/api/mo/sys.json". Below it are three buttons: "Send" (blue), "Reset" (orange), and "Convert" (blue).
- Request Panel:** A tabbed interface with "Request" selected. It shows the command "hostname REST2CLI" and a green "Copy" button.
- Response Panel:** A section labeled "Response:" with a large empty text area and a green "Copy" button.

Step 4 When a valid payload is present in the **Request** pane, you can click **Send** to send the payload as an API call to the switch.

The response from the switch appears in the **Response** pane.

Warning Clicking **Send** commits the command to the switch, which can result in a configuration or state change.

The screenshot displays the NX-API Developer Sandbox interface. At the top, there are navigation links: Quick Start, DME Documentation, Model Browser, and Logout. The main area is divided into several sections:

- Top Left:** A text area containing the command "Logging level netstack 6".
- Top Right:** Controls for "Method" (set to NX-API-REST (DME)) and "Input type" (set to cli).
- Request Bar:** A dropdown menu set to "POST" and a text input field containing "/api/mo/sys.json". To the right are buttons for "Send", "Reset", and "Convert".
- Request Pane:** A tabbed interface with tabs for "Request", "Python", "Python3", "Java", "JavaScript", and "Go-Lang". The "Request" tab is active, showing a JSON payload:


```
{
  "topSystem": {
    "children": [
      {
        "ipv4Entity": {
          "children": [
            {
              "ipv4Inst": {
                "attributes": {
                  "loggingLevel": "informational"
                }
              }
            }
          ]
        }
      }
    ]
  }
}
```

 A green "Copy" button is visible in the top right corner of this pane.
- Response Pane:** A section labeled "Response:" showing a JSON payload:


```
{
  "imdata": []
}
```

 A green "Copy" button is visible in the top right corner of this pane.

Step 5 You can copy the contents of the **Request** or **Response** pane to the clipboard by clicking **Copy** in the pane.

Step 6 You can also convert the request into the following formats by clicking on the appropriate tab in the **Request** pane:

- Python
- Python3
- Java
- JavaScript
- Go-Lang

Using the Developer Sandbox to Convert from REST Payloads to CLI Commands



Tip Online help is available by clicking **Quick Start** in the upper right corner of the Sandbox window. Additional details, such as response codes and security methods, can be found in the chapter "NX-API CLI". Click on the **DME Documentation** link in the upper right corner of the Sandbox window to go to the NX-API DME Model Reference page.

Click on the **Model Browser** link in the upper right corner of the Sandbox window to access Visore, the Model Browser. Note that you might have to manually enter the IP address for your switch to access the Visore page:

`https://management-ip-address/visore.html.`

Procedure

Step 1 Select **NXAPI-REST (DME)** as the **Method** and **model** as the **Input Type**.

Example:

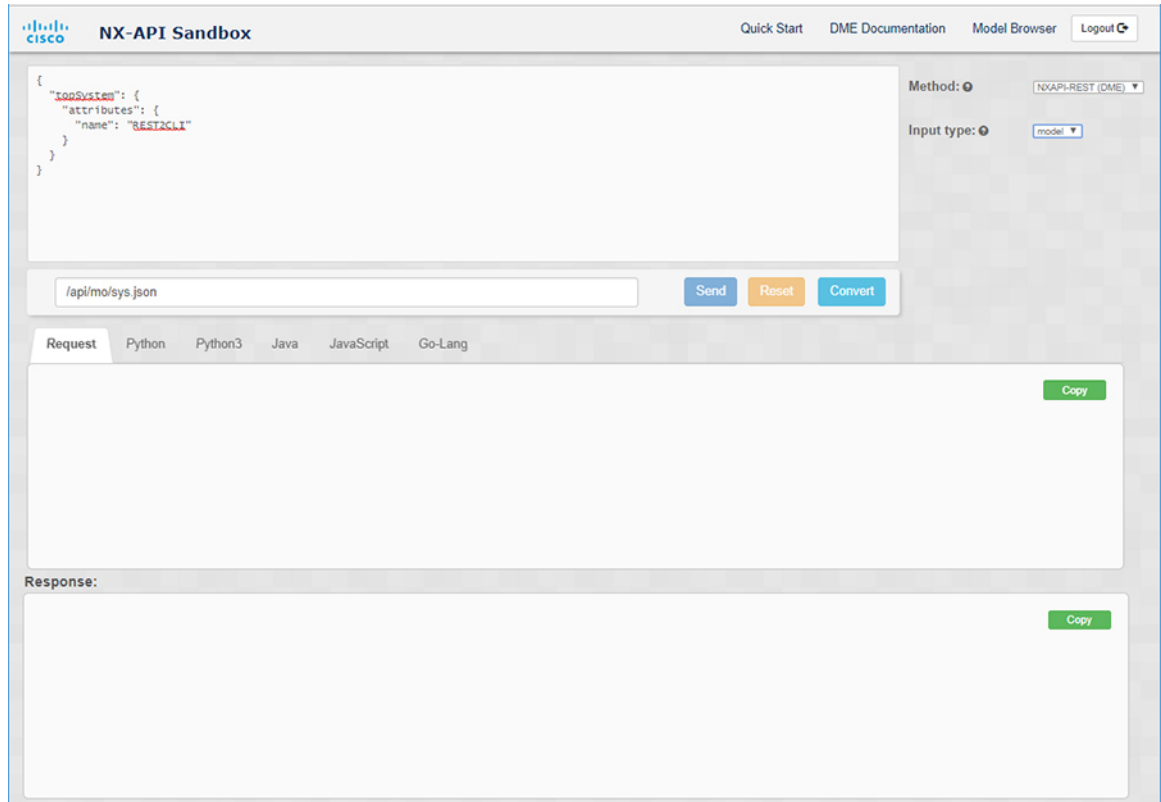
The screenshot shows the NX-API Sandbox interface. At the top, there is a navigation bar with the Cisco logo, the title "NX-API Sandbox", and links for "Quick Start", "DME Documentation", "Model Browser", and "Logout". Below the navigation bar is a large text area for entering the DME payload, with the placeholder text "Enter DME payload here.". To the right of this area are two dropdown menus: "Method:" set to "NXAPI-REST (DME)" and "Input type:" set to "model". Below these is a text input field containing the payload "/api/mo/sys.json" and three buttons: "Send", "Reset", and "Convert". Below the payload input is a row of tabs: "Request", "Python", "Python3", "Java", "JavaScript", and "Go-Lang". The "Request" tab is selected, showing a large empty area with a "Copy" button in the top right corner. Below this is a "Response:" section, also with a "Copy" button in the top right corner.

Step 2 Enter a DN and payload into the text entry box in the top pane. Then click on the **Convert** button below the top pane.

Example:

For this example, the DN is `/api/mo/sys.json` and the NX-API REST payload is:

```
{
  "topSystem": {
    "attributes": {
      "name": "REST2CLI"
    }
  }
}
```



When you click on the **Convert** button, the CLI equivalent appears in the **CLI** pane as shown in the following image.

NX-API Sandbox

[Quick Start](#)
[DME Documentation](#)
[Model Browser](#)
[Logout](#)

```
{
  "topSystem": {
    "attributes": {
      "name": "REST2CLI"
    }
  }
}
```

/api/mo/sys.json

Send
Reset
Convert

Request
Python
Python3
Java
JavaScript
Go-Lang

Copy

```
hostname REST2CLI
```

Response:

Copy

Note The Developer Sandbox cannot convert all payloads into equivalent CLIs, even if the Sandbox converted the CLIs to NX-API REST payloads. The following is a list of possible sources of error that can prevent a payload from completely converting to CLI commands:

Table 3: Sources of REST2CLI Errors

Payload Issue	Result
<p>The payload contains an attribute that does not exist in the MO.</p> <p>Example:</p> <pre>api/mo/sys.json { "topSystem": { "children": [{ "interfaceEntity": { "children": [{ "l1PhysIf": { "attributes": { "id": "eth1/1", "fakeattribute": "totallyFake" } } }] } }] } }</pre>	<p>The Error pane will return an error related to the attribute.</p> <p>Example:</p> <p>CLI</p> <p>Error unknown attribute 'fakeattribute' in element 'l1PhysIf'</p>
<p>The payload includes MOs that aren't yet supported for conversion:</p> <p>Example:</p> <pre>api/mo/sys.json { "topSystem": { "children": [{ "dhcpEntity": { "children": [{ "dhcpInst": { "attributes": { "SnoopingEnabled": "yes" } } }] } }] } }</pre>	<p>The Error Pane will return an error related to the unsupported MO.</p> <p>Example:</p> <p>CLI</p> <p>Error The entire subtree of "sys/dhcp" is not converted.</p>

Using the Developer Sandbox to Convert from RESTCONF to json or XML



- Tip** Online help is available by clicking **Quick Start** in the upper right corner of the Sandbox window.
- Click on the **Yang Documentation** link in the upper right corner of the Sandbox window to go to the Model Driven Programmability with Yang page.
- Click on the **Yang Models** link in the upper right corner of the Sandbox window to access the YangModels GitHub site.

Procedure

- Step 1** Select **RESTCONF (Yang)** as the **Method** and either **json** or **xml** as the **Message format**.

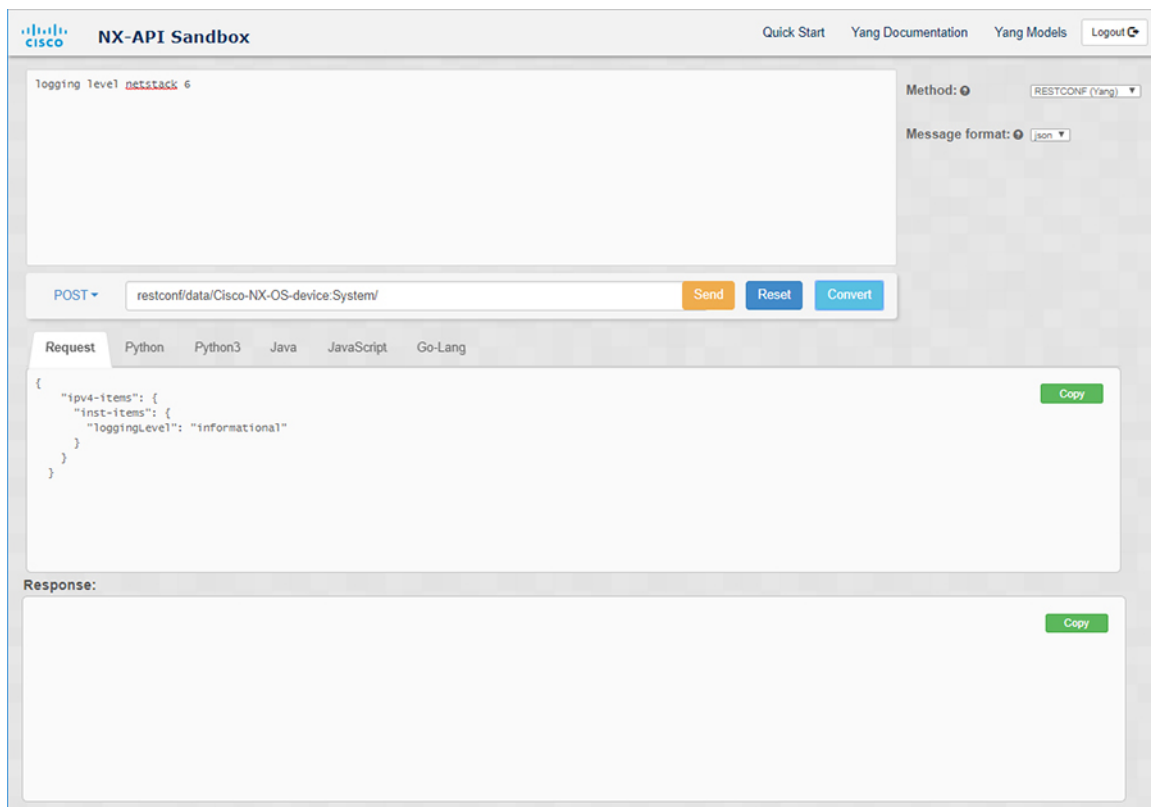
Example:

The screenshot displays the NX-API Sandbox interface. At the top, there are navigation links: "Quick Start", "Yang Documentation", "Yang Models", and "Logout". The main area is divided into two panes. The top pane contains a text entry box with the command "Logging level netstack 6". To the right of this box, there are two dropdown menus: "Method" set to "RESTCONF (Yang)" and "Message format" set to "json". Below the text entry box, there is a "POST" dropdown menu, a text input field containing the URL "restconf/data/Cisco-NX-OS-device:System/", and three buttons: "Send", "Reset", and "Convert". Below the top pane, there are tabs for "Request", "Python", "Python3", "Java", "JavaScript", and "Go-Lang". The "Request" tab is active, showing a large empty text area with a "Copy" button in the top right corner. Below the "Request" pane, there is a "Response:" label and another large empty text area with a "Copy" button in the top right corner.

- Step 2** Enter a command into the text entry box in the top pane, choose a message format, then click on the **Convert** button below the top pane.

Example:

For this example, the command is **logging level netstack 6** and the message format is json:



The screenshot displays the NX-API Developer Sandbox interface. At the top, the Cisco logo and "NX-API Sandbox" are visible, along with navigation links for "Quick Start", "Yang Documentation", "Yang Models", and "Logout". The main area is divided into several sections:

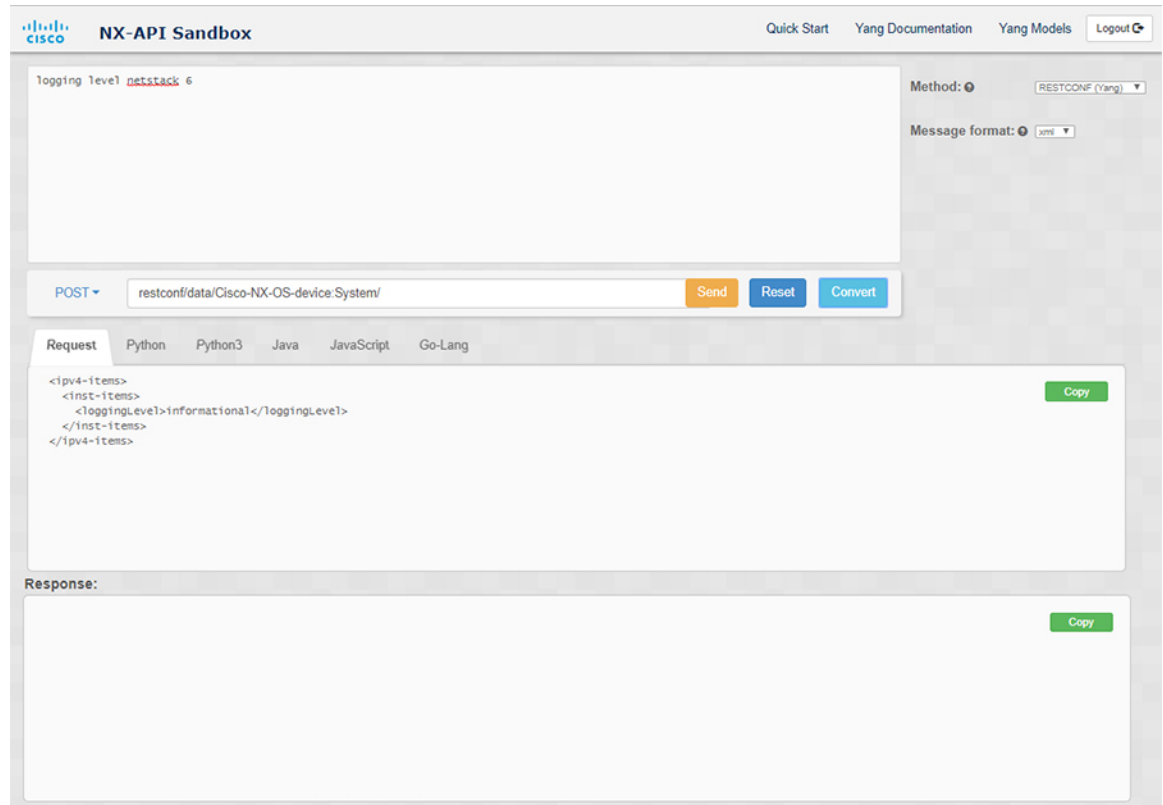
- Input Area:** A text box contains the command "Logging level netstack 6". To the right, there are dropdown menus for "Method" (set to "RESTCONF (Yang)") and "Message format" (set to "json").
- Request Form:** A dropdown menu is set to "POST" and the URL is "restconf/data/Cisco-NX-OS-device:System/". There are "Send", "Reset", and "Convert" buttons.
- Request Tab:** Below the form, there are tabs for "Request", "Python", "Python3", "Java", "JavaScript", and "Go-Lang". The "Request" tab is active, showing a JSON payload:

```
{  "ipv4-items": {    "inst-items": {      "loggingLevel": "informational"    }  }}
```

 A "Copy" button is located to the right of the JSON.
- Response Area:** A section labeled "Response:" is currently empty, with a "Copy" button to its right.

Example:

For this example, the command is `logging level netstack 6` and the message format is xml:



The screenshot displays the NX-API Sandbox interface. At the top, there is a header with the Cisco logo and the text "NX-API Sandbox". To the right of the header are links for "Quick Start", "Yang Documentation", "Yang Models", and a "Logout" button. The main area is divided into two sections. The top section contains a text input field with the text "logging level **netstack** 6". To the right of this field are two dropdown menus: "Method:" set to "RESTCONF (Yang)" and "Message format:" set to "xml". Below the input field is a "POST" dropdown menu, a text input field containing "restconf/data/Cisco-NX-OS-device:System/", and three buttons: "Send" (orange), "Reset" (blue), and "Convert" (blue). Below this is a "Request" tab, which is currently selected. It shows a tabbed interface with options for "Request", "Python", "Python3", "Java", "JavaScript", and "Go-Lang". The "Request" tab displays the following XML content:

```
<ipv4-items>
<inst-items>
  <loggingLevel>informational</loggingLevel>
</inst-items>
</ipv4-items>
```

 To the right of this content is a green "Copy" button. Below the "Request" tab is a "Response:" section, which is currently empty. To the right of this section is another green "Copy" button.

Step 3 You can also convert the request into the following formats by clicking on the appropriate tab in the **Request** pane:

- Python
- Python3
- Java
- JavaScript
- Go-Lang

Note The Java-generated script does not work if you choose the PATCH option from the drop-down menu in the area above the Request tab. This is a known limitation with Java and is expected behavior.

