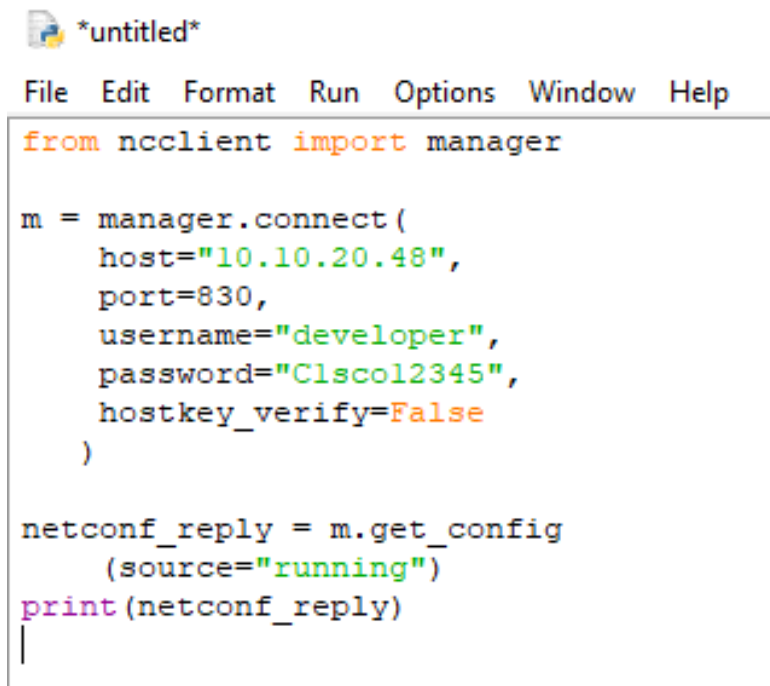


## 2.9 Lab - NETCONF wPython Get Operational Data

En este laboratorio, aprenderá a usar el ncclient de NETCONF para recuperar la configuración del dispositivo y actualizar y crear una nueva configuración de interfaz. También aprenderá por qué el soporte transaccional de NETCONF es importante para obtener cambios consistentes en la red.

**Utilice ncclient para recuperar la configuración en ejecución del dispositivo.**

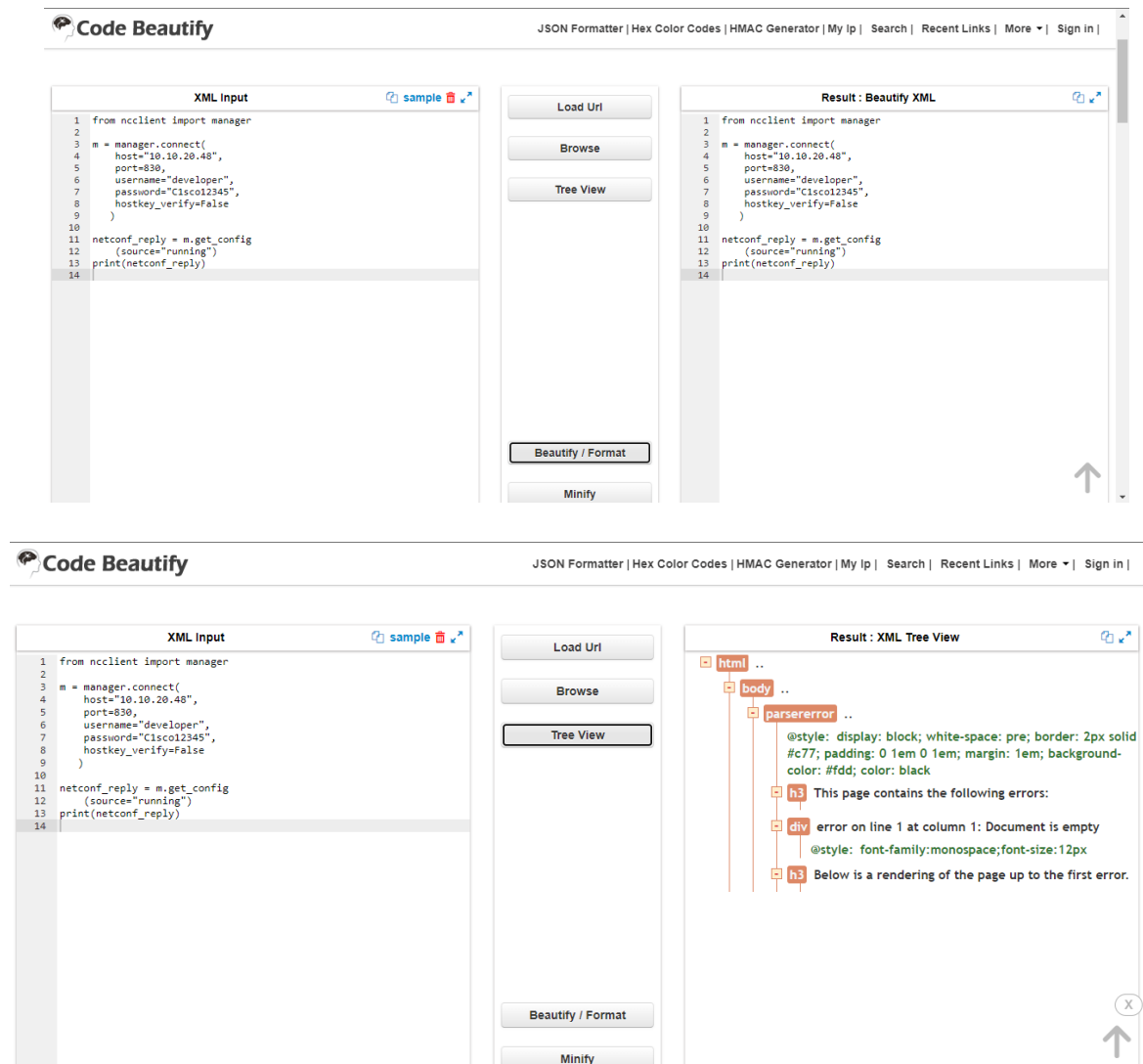


```
*untitled*
File Edit Format Run Options Window Help
from ncclient import manager

m = manager.connect(
    host="10.10.20.48",
    port=830,
    username="developer",
    password="Cisc012345",
    hostkey_verify=False
)

netconf_reply = m.get_config
    (source="running")
print(netconf_reply)
|
```

Utilice CodeBeautify.com para evaluar la respuesta.



The top screenshot shows the CodeBeautify.com interface. The 'XML Input' field contains a Python script for connecting to a device via ncclient. The 'Result: Beautify XML' field shows the formatted XML output. The bottom screenshot shows the same interface, but the 'Result: XML Tree View' field displays a tree view of the XML document, which includes a parser error message.

Actualizar la configuración del dispositivo

Cree un nuevo archivo de script de Python.

```
Lab 2.8 parte 2.py - D:/Laboratorios/Lab 2.8 parte 2.py (3.10.9)
File Edit Format Run Options Window Help
from ncclient import manager
import xml.dom.minidom

m = manager.connect(
    host="10.10.20.48",
    port=830,
    username="developer",
    password="Cisco12345",
    hostkey_verify=False
)
```

Cambie el nombre de host.

```
netconf_reply = m.edit_config(target="running", config=netconf_data)
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

Crear una interfaz de bucle invertido

```
Lab 2.8 parte 2.py - D:/Laboratortios/Lab 2.8 parte 2.py (3.10.9)
File Edit Format Run Options Window Help
from ncclient import manager
import xml.dom.minidom

m = manager.connect(
    host="10.10.20.48",
    port=830,
    username="developer",
    password="Cisc012345",
    hostkey_verify=False
)

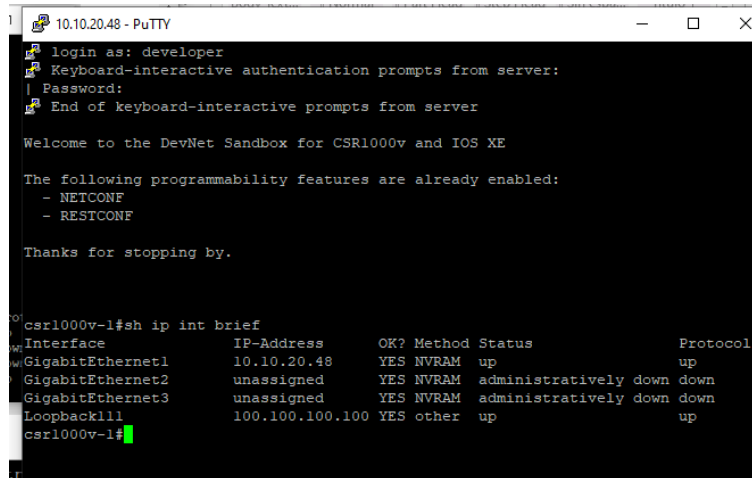
netconf_data = """
<config>
<native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
  <interface>
    <Loopback>
      <name>111</name>
      <description>TEST1</description>
      <ip>
        <address>
          <primary>
            <address>100.100.100.100</address>
            <mask>255.255.255.0</mask>
          </primary>
        </address>
      </ip>
    </Loopback>
  </interface>
</native>
</config>
"""

netconf_reply = m.edit_config(target="running", config=netconf_data)
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

```
===== RESTART: D:/Laboratortios/Lab 2.8 parte 2.py =====
<?xml version="1.0" ?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:64b489dd-4473-4676-aa09-ea2cfec2b890">
  <ok/>
</rpc-reply>

>>> |
```

## Ejecutando comando “sh ip int brief”



```
10.10.20.48 - PuTTY
login as: developer
Keyboard-interactive authentication prompts from server:
Password:
End of keyboard-interactive prompts from server

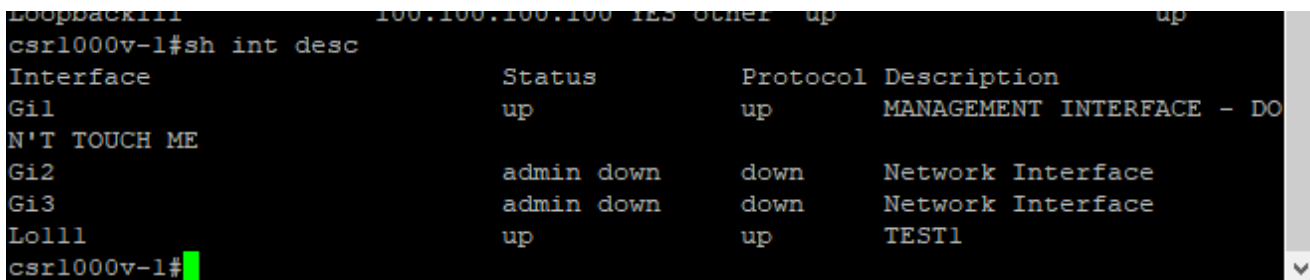
Welcome to the DevNet Sandbox for CSR1000v and IOS XE

The following programmability features are already enabled:
- NETCONF
- RESTCONF

Thanks for stopping by.

csr1000v-1#sh ip int brief
Interface      IP-Address      OK? Method Status      Protocol
GigabitEthernet1  10.10.20.48     YES NVRAM    up          up
GigabitEthernet2  unassigned      YES NVRAM    administratively down down
GigabitEthernet3  unassigned      YES NVRAM    administratively down down
Loopback111      100.100.100.100 YES other    up          up
csr1000v-1#
```

## Ejecutando el comando “sh int desc”



```
Loopback111      100.100.100.100 YES other    up          up
csr1000v-1#sh int desc
Interface      Status      Protocol Description
Gi1            up          up          MANAGEMENT INTERFACE - DO
N'T TOUCH ME
Gi2            admin down  down        Network Interface
Gi3            admin down  down        Network Interface
Lo111         up          up          TEST1
csr1000v-1#
```

Realizamos una práctica parecida a la anterior del laboratorio 2.7, aquí pusimos a prueba nuevamente ncclient de esta manera realizando una actividad más completa para llegar a comprender mejor como es que funciona y para que es.

Así como lo vimos antes ya sabemos que ncclient proporciona API's y estas API's pueden ser mapeadas de una forma inteligente de esta manera logran facilitar los trabajos de administración de las aplicaciones.