



Python API

Python プログラマビリティは、Python API をサポートしています。

- [Python について \(1 ページ\)](#)
- [Python API に関するその他の参考資料 \(10 ページ\)](#)
- [Python API の機能情報 \(10 ページ\)](#)

Python について

Cisco IOS XE デバイスは、ゲストシェル内でインタラクティブおよび非インタラクティブ（スクリプト）の両方のモードで Python バージョン 2.7 をサポートします。Python スクリプト機能により、デバイスの CLI にプログラムを使用してアクセスして、さまざまなタスク、およびゼロ タッチ プロビジョニングまたは Embedded Event Manager (EEM) アクションを実行することができます。

Cisco Python モジュール

シスコが提供する Python モジュールでは、EXEC および設定コマンドを実行するアクセス権が提供されます。**help()** コマンドを入力すると、Cisco Python モジュールの詳細が表示されます。**help()** コマンドは Cisco CLI モジュールのプロパティを表示します。

次の例は、Cisco Python モジュールに関する情報を示します。

```
Device# guestshell run python

Python 2.7.5 (default, Jun 17 2014, 18:11:42)
[GCC 4.8.2 20140120 (Red Hat 4.8.2-16)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> >>> from cli import cli,clip,configure,configurep, execute, executep
>>> help(configure)
Help on function configure in module cli:

configure(configuration)
Apply a configuration (set of Cisco IOS CLI config-mode commands) to the device
and return a list of results.

configuration = '''interface gigabitEthernet 0/0
```

```

no shutdown'''

# push it through the Cisco IOS CLI.
try:
    results = cli.configure(configuration)
    print "Success!"
except CLIConfigurationError as e:
    print "Failed configurations:"
    for failure in e.failed:
        print failure

Args:
configuration (str or iterable): Configuration commands, separated by newlines.

Returns:
list(ConfigResult): A list of results, one for each line.

Raises:
CLISyntaxError: If there is a syntax error in the configuration.

>>> help (configurep)
Help on function configurep in module cli:

configurep(configuration)
Apply a configuration (set of Cisco IOS CLI config-mode commands) to the device
and prints the result.

configuration = '''interface gigabitEthernet 0/0
no shutdown'''

# push it through the Cisco IOS CLI.
configurep(configuration)

Args:
configuration (str or iterable): Configuration commands, separated by newlines.
>>> help (execute)
Help on function execute in module cli:

execute(command)
Execute Cisco IOS CLI exec-mode command and return the result.

command_output = execute("show version")

Args:
command (str): The exec-mode command to run.

Returns:
str: The output of the command.

Raises:
CLISyntaxError: If there is a syntax error in the command.

>>> help (executep)
Help on function executep in module cli:

executep(command)
Execute Cisco IOS CLI exec-mode command and print the result.

executep("show version")

Args:
command (str): The exec-mode command to run.

```

```
>>> help(cli)
Help on function cli in module cli:

cli(command)
    Execute Cisco IOS CLI command(s) and return the result.

    A single command or a delimited batch of commands may be run. The
    delimiter is a space and a semicolon, ";". Configuration commands must be
    in fully qualified form.

    output = cli("show version")
    output = cli("show version ; show ip interface brief")
    output = cli("configure terminal ; interface gigabitEthernet 0/0 ; no shutdown")

    Args:
        command (str): The exec or config CLI command(s) to be run.

    Returns:
        string: CLI output for show commands and an empty string for
        configuration commands.

    Raises:
        errors.cli_syntax_error: if the command is not valid.
        errors.cli_exec_error: if the execution of command is not successful.
```

```
>>> help(clip)
Help on function clip in module cli:

clip(command)
    Execute Cisco IOS CLI command(s) and print the result.

    A single command or a delimited batch of commands may be run. The
    delimiter is a space and a semicolon, ";". Configuration commands must be
    in fully qualified form.

    clip("show version")
    clip("show version ; show ip interface brief")
    clip("configure terminal ; interface gigabitEthernet 0/0 ; no shutdown")

    Args:
        command (str): The exec or config CLI command(s) to be run.
```

IOS CLI コマンドを実行するための Cisco Python モジュール



(注) Python を実行するには、ゲストシェルが有効である必要があります。詳細については、「ゲストシェル」の章を参照してください。

Python プログラミング言語は CLI コマンドを実行できる 6 つの関数を使用します。これらの関数は、Python CLI モジュールから利用できます。これらの関数を使用するには、**import cli** コマンドを実行します。これらの関数が機能するには、**ip http server** コマンドが有効になっている必要があります。

これらの関数の引数は CLI コマンドの文字列です。Python インタープリタ経由で CLI コマンドを実行するには、次の 6 つの関数のいずれかの引数文字列として CLI コマンドを入力します。

- **cli.cli(command)** : この関数は IOS コマンドを引数として取り、IOS パーサーからコマンドを実行し、結果のテキストを返します。このコマンドの形式が正しくない場合、Python の例外が発生します。次に、**cli.cli(command)** 関数の出力例を示します。

```
>>> import cli
>>> cli.clip('configure terminal; interface loopback 10; ip address
10.10.10.10 255.255.255.255')
*Mar 13 18:39:48.518: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback10,
changed state to up
>>> cli.clip('show clock')
'\n*18:11:53.989 UTC Mon Mar 13 2017\n'
>>> output=cli.cli('show clock')
>>> print(output)
*18:12:04.705 UTC Mon Mar 13 2017
```

- **cli.clip(command)** : この関数は **cli.cli(command)** 関数と機能はまったく同じです。ただし結果のテキストを（返すのではなく）stdout に出力する点が異なります。次に、**cli.clip(command)** 関数の出力例を示します。

```
>>> cli
>>> cli.clip('configure terminal; interface loopback 11; ip address
10.11.11.11 255.255.255.255')
*Mar 13 18:42:35.954: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback11,
changed state to up
*Mar 13 18:42:35.954: %LINK-3-UPDOWN: Interface Loopback11, changed state to up
>>> cli.clip('show clock')
*18:13:35.313 UTC Mon Mar 13 2017
>>> output=cli.clip('show clock')
*18:19:26.824 UTC Mon Mar 13 2017
>>> print (output)
None
```

- **cli.execute(command)** : この関数は単一の EXEC コマンドを実行して出力を返します。ただし結果のテキストは出力しません。このコマンドの一部としてセミコロンまたは改行を使用することは許可されません。この関数を複数回実行するには、for-loop が指定された Python リストを使用します。次に、**cli.execute(command)** 関数の出力例を示します。

```
>>> cli.execute("show clock")
'15:11:20.816 UTC Thu Jun 8 2017'
>>>
>>> cli.execute('show clock'; 'show ip interface brief')
File "<stdin>", line 1
    cli.execute('show clock'; 'show ip interface brief')
    ^
SyntaxError: invalid syntax
```

```
>>>
```

- **cli.executep(command)** : この関数は単一のコマンドを実行して、結果のテキストを（返すのではなく）stdout に出力します。次に、**cli.executep(command)** 関数の出力例を示します。

```
>>> cli.executep('show clock')
*18:46:28.796 UTC Mon Mar 13 2017
>>> output=cli.executep('show clock')
*18:46:36.399 UTC Mon Mar 13 2017
>>> print(output)
None
```

- **cli.configure(command)** : この関数は、コマンドで使用できる設定によりデバイスを設定します。これは次に示すように、コマンドとその結果が含まれる名前付きタブルのリストを返します。

```
[Think: result = (bool(success), original_command, error_information)]
```

コマンドパラメータは複数行に入力することができ、**show running-config** コマンドの出力に表示されているのと同じ形式にすることができます。次に、**cli.configure(command)** 関数の出力例を示します。

```
>>>cli.configure(["interface GigabitEthernet1/0/7", "no shutdown",
"end"])
[ConfigResult(success=True, command='interface GigabitEthernet1/0/7',
line=1, output='', notes=None), ConfigResult(success=True, command='no shutdown',
line=2, output='', notes=None), ConfigResult(success=True, command='end',
line=3, output='', notes=None)]
```

- **cli.configurep(command)** : この関数は **cli.configure(command)** 関数と機能はまったく同じです。ただし結果のテキストを（返すのではなく）stdout に出力する点が異なります。次に、**cli.configurep(command)** 関数の出力例を示します。

```
>>> cli.configurep(["interface GigabitEthernet1/0/7", "no shutdown",
"end"])
Line 1 SUCCESS: interface GigabitEthernet1/0/7
Line 2 SUCCESS: no shut
Line 3 SUCCESS: end
```

Python スクリプトの概要

Python は、仮想化された Linux ベースの環境であるゲストシェルで実行されます。詳細については、「ゲストシェル」の章を参照してください。シスコが提供する Python モジュールは、ユーザの Python スクリプトがホスト デバイス上で IOS CLI コマンドを実行することを可能にします。

対話形式の Python プロンプト

デバイス上で **guestshell run python** コマンドを実行すると、ゲストシェル内で、対話形式の Python プロンプトが開きます。Python の対話モードでは、Cisco Python CLI モジュールから Python 機能を実行してデバイスを設定することができます。

次の例は、対話形式の Python プロンプトを有効にする方法を示しています。

```
Device# guestshell run python

Python 2.7.5 (default, Jun 17 2014, 18:11:42)
[GCC 4.8.2 20140120 (Red Hat 4.8.2-16)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>

Device#
```

Python スクリプト

Python スクリプト名を引数として Python コマンドで使用することで、Python スクリプトを非インタラクティブモードで実行できます。Python スクリプトは、ゲストシェル内からアクセス可能である必要があります。ゲストシェルから Python スクリプトにアクセスするには、ゲストシェル内にマウントされているブートフラッシュまたはフラッシュにスクリプトを保存します。



(注) Python で **import cli** が機能するように、**ip http server** コマンドを設定する必要があります。

次の Python スクリプトの例は、さまざまな CLI 関数を使用して **show** コマンドを設定および出力します。

```
Device# more flash:sample_script.py

import sys
import cli

intf= sys.argv[1:]
intf = ''.join(intf[0])

print "\n\n *** Configuring interface %s with 'configurep' function *** \n\n" %intf
cli.configurep(["interface loopback55","ip address 10.55.55.55 255.255.255.0","no
shut","end"])

print "\n\n *** Configuring interface %s with 'configure' function *** \n\n"
cmd='interface %s,logging event link-status ,end' % intf
cli.configure(cmd.split(', '))

print "\n\n *** Printing show cmd with 'executep' function *** \n\n"
cli.executep('show ip interface brief')

print "\n\n *** Printing show cmd with 'execute' function *** \n\n"
output= cli.execute('show run interface %s' %intf)
print (output)
```

```
print "\n\n *** Configuring interface %s with 'cli' function *** \n\n"
cli.cli('config terminal; interface %s; spanning-tree portfast edge default' %intf)

print "\n\n *** Printing show cmd with 'clip' function *** \n\n"
cli.clip('show run interface %s' %intf)
```

To run a Python script from the Guest Shell, execute the `guestshell run python /flash/script.py` command at the device prompt. The following example shows how to run a Python script from the Guest Shell:

次の例は、ゲストシェルから Python スクリプトを実行する方法を示しています。

```
Device# guestshell run python /flash/sample_script.py loop55

*** Configuring interface loop55 with 'configurep' function ***

Line 1 SUCCESS: interface loopback55
Line 2 SUCCESS: ip address 10.55.55.55 255.255.255.0
Line 3 SUCCESS: no shut
Line 4 SUCCESS: end

*** Configuring interface %s with 'configure' function ***

*** Printing show cmd with 'executep' function ***

Interface                IP-Address      OK? Method Status          Protocol
Vlan1                    unassigned     YES NVRAM  administratively down  down
GigabitEthernet0/0      192.0.2.1      YES NVRAM  up              up
GigabitEthernet1/0/1    unassigned     YES unset  down            down
GigabitEthernet1/0/2    unassigned     YES unset  down            down
GigabitEthernet1/0/3    unassigned     YES unset  down            down
:
:
:
Tel1/1/4                 unassigned     YES unset  down            down
Loopback55               10.55.55.55    YES TFTP  up              up
Loopback66               unassigned     YES manual up              up

*** Printing show cmd with 'execute' function ***

Building configuration...
Current configuration : 93 bytes
!
interface Loopback55
 ip address 10.55.55.55 255.255.255.0
 logging event link-status
end

*** Configuring interface %s with 'cli' function ***

*** Printing show cmd with 'clip' function ***

Building configuration...
Current configuration : 93 bytes
!
interface Loopback55
 ip address 10.55.55.55 255.255.255.0
 logging event link-status
```

```
end
```

サポートされる Python のバージョン

ゲストシェルは、Python バージョン 2.7 をプリインストールしています。ゲストシェルは、仮想化された Linux ベースの環境であり、Cisco デバイスの自動制御と管理のための Python アプリケーションを含む、カスタム Linux アプリケーションを実行するように設計されています。Montavista CGE7 がインストールされたプラットフォームは Python バージョン 2.7.11 をサポートし、CentOS 7 がインストールされたプラットフォームは Python バージョン 2.7.5 をサポートします。

次の表は、Python の各バージョンおよびサポート対象のプラットフォームに関する情報を示しています。

表 1: Python バージョンサポート

Python のバージョン	プラットフォーム
Python バージョン 2.7.5	Cisco Catalyst 3650 シリーズ スイッチおよび Cisco Catalyst 3850 シリーズ スイッチを除くすべてのサポート対象プラットフォーム。
Python バージョン 2.7.11	<ul style="list-style-type: none">• Cisco Catalyst 3650 シリーズ スイッチ• Cisco Catalyst 3850 シリーズ スイッチ

Python のバージョン	プラットフォーム
Python バージョン 3.6	<p>Cisco IOS XE Amsterdam 17.1.1 以降のリリースでサポートされています。</p> <p>Cisco IOS XE Amsterdam 17.1.1 および Cisco IOS XE Amsterdam 17.2.1 では、Python V2 がデフォルトです。ただし、Cisco IOS XE Amsterdam 17.3.1 以降のリリースでは、Python V3 がデフォルトです。</p> <p>(注) Cisco Catalyst 9200 シリーズスイッチは、Cisco IOS XE Amsterdam 17.1.1 および Cisco IOS XE Amsterdam 17.2.1 で Python Version 3.6 をサポートしていません。Cisco Catalyst 9200 シリーズスイッチは、Cisco IOS XE Amsterdam 17.3.1 以降のリリースで Python V3 をサポートしています。</p> <p>(注) Cisco Catalyst 3650 シリーズスイッチおよび Cisco Catalyst 3850 シリーズスイッチではサポートされていません。</p>

CentOS 7 がインストールされたプラットフォームは、オープンソースリポジトリからの Redhat Package Manager (RPM) のインストールをサポートします。

Cisco CLI Python モジュールの更新

Cisco CLI Python モジュールおよび EEM モジュールは、デバイスにインストール済みです。ただし、Yum または事前にパッケージ化されているバイナリのいずれかを使用して Python のバージョンを更新する場合は、シスコが提供する CLI モジュールも更新する必要があります。



(注) Python バージョン 2 がすでにあるデバイスで Python バージョン 3 への更新を行うと、デバイス上には両方のバージョンの Python が存在するようになります。Python を実行するには、次の IOS コマンドのいずれかを使用します。

- **guestshell run python2** コマンドは、Python バージョン 2 を有効化します。
- **guestshell run python3** コマンドは、Python バージョン 3 を有効化します。
- **guestshell run python** コマンドは、Python バージョン 2 を有効化します。

Python のバージョンを更新するには、次の方法のいずれかを使用します。

- スタンドアロン tarball のインストール
- CLI モジュールのための PIP のインストール

Python API に関するその他の参考資料

関連資料

関連項目	マニュアル タイトル
ゲスト シェル	ゲスト シェル
EEM Python モジュール	EEM の Python スクリプト

シスコのテクニカル サポート

説明	リンク
<p>シスコのサポート Web サイトでは、シスコの製品やテクノロジーに関するトラブルシューティングにお役立ていただけるように、マニュアルやツールをはじめとする豊富なオンラインリソースを提供しています。</p> <p>お使いの製品のセキュリティ情報や技術情報を入手するために、Cisco Notification Service (Field Notice からアクセス)、Cisco Technical Services Newsletter、Really Simple Syndication (RSS) フィードなどの各種サービスに加入できます。</p> <p>シスコのサポート Web サイトのツールにアクセスする際は、Cisco.com のユーザ ID およびパスワードが必要です。</p>	http://www.cisco.com/support

Python API の機能情報

次の表に、このモジュールで説明した機能に関するリリース情報を示します。この表は、ソフトウェア リリース トレインで各機能のサポートが導入されたときのソフトウェア リリースだけを示しています。その機能は、特に断りがない限り、それ以降の一連のソフトウェア リリースでもサポートされます。

プラットフォームのサポートおよびシスコソフトウェアイメージのサポートに関する情報を検索するには、Cisco Feature Navigator を使用します。Cisco Feature Navigator にアクセスするには、www.cisco.com/go/cfn に移動します。Cisco.com のアカウントは必要ありません。

表 2: CLI Python モジュールの機能情報

機能名	リリース	機能情報
CLI Python モジュール	Cisco IOS XE Everest 16.5.1a	<p>Python プログラマビリティでは、ユーザが CLI を使用して IOS と対話できるようにする Python モジュールが提供されます。</p> <p>Cisco IOS XE Everest 16.5.1a では、この機能は次のプラットフォームに実装されていました。</p> <ul style="list-style-type: none"> • Cisco Catalyst 3650 シリーズ スイッチ • Cisco Catalyst 3850 シリーズ スイッチ • Cisco Catalyst 9300 シリーズ スイッチ • Cisco Catalyst 9500 シリーズ スイッチ <p>Cisco IOS XE Everest 16.5.1b では、この機能は次のプラットフォームに実装されていました。</p> <ul style="list-style-type: none"> • Cisco 4000 シリーズ サービス統合型ルータ
	Cisco IOS XE Everest 16.6.2	この機能は、Cisco Catalyst 9400 シリーズ スイッチに実装されました。
	Cisco IOS XE Fuji 16.7.1	<p>この機能は、次のプラットフォームに実装されていました。</p> <ul style="list-style-type: none"> • Cisco ASR 1000 アグリゲーション サービス ルータ • Cisco CSR 1000v シリーズクラウド サービス ルータ
	Cisco IOS XE Fuji 16.8.1 Cisco IOS XE Fuji 16.8.1a	

機能名	リリース	機能情報
		<p>Cisco IOS XE Fuji 16.8.1 では、この機能は次のプラットフォームに実装されていました。</p> <ul style="list-style-type: none">• Cisco ASR 1004 ルータ• Cisco ASR 1006 ルータ• Cisco ASR 1006-X ルータ• Cisco ASR 1009-X ルータ• Cisco ASR 1013 ルータ• 最低 4 GB の RAM を搭載した Cisco 4000 シリーズ サービス統合型ルータ モデル <p>Cisco IOS XE Fuji 16.8.1a では、この機能は Cisco Catalyst 9500 ハイパフォーマンス シリーズ スイッチに実装されていました。</p>

翻訳について

このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。