コマンドライン インターフェイスの概要

この章では、Cisco NX-OS ソフトウェアのコマンドライン インターフェイス（CLI）について説明します。

この章は、次の内容で構成されています。

CLI プロンプトの概要

デバイスに正常にアクセスすると、コンソールポートまたはリモート ワークステーションの端末ウィンドウに、次のような CLI プロンプトが表示されます。

```
User Access Verification
login: admin
Password:<password>
Cisco Nexus Operating System (NX-OS) Software
TAC support: http://www.cisco.com/tac
Copyright (c) 2002-2009, Cisco Systems, Inc. All rights reserved.
The copyrights to certain works contained in this software are
owned by other third parties and used and distributed under
license. Certain components of this software are licensed under
the GNU General Public License (GPL) version 2.0 or the GNU
Lesser General Public License (LGPL) Version 2.1. A copy of each
such license is available at
http://www.opensource.org/licenses/gpl-2.0.php and
http://www.opensource.org/licenses/lgpl-2.1.php
switch#
```

デバイスのデフォルトのホスト名は変更できます。

CLI プロンプトから、次の方法を実行できます。

- CLI コマンドによる機能の設定
- コマンド履歴へのアクセス
- コマンド解析機能の使用

（注） 通常の操作では、ユーザ名は大文字と小文字が区別されます。 ただし、コンソール ポートにデバイスを接続しているときには、ユーザ名の設定に関係なく、すべて大文字でログインユーザ名を入力できます。 正しいパスワードを入力すれば、デバイスにログインできます。

コマンド モード

ここでは、Cisco NX-OS の CLI でのコマンド モードについて説明します。

EXEC コマンド モード

最初にログインしたときは、Cisco NX-OS ソフトウェアは EXEC モードになります。 EXEC モードで使用可能なコマンドには、デバイスの状態および構成に関する情報を表示する **show** コマン

ド、**clear** コマンド、デバイス コンフィギュレーションに保存しない処理を実行するその他のコマンドなどがあります。

グローバル コンフィギュレーション コマンド モード

グローバル コンフィギュレーション モードでは、最も広範囲のコマンドを使用できます。 この用語は、デバイス全体に影響を与える特性や機能を表します。グローバルコンフィギュレーション モードでコマンドを入力すると、デバイスをグローバルに設定することができます。また、さらに特定のコンフィギュレーション モードを開始して、インターフェイスやプロトコルなどの特定の要素を設定することもできます。

手順の概要

1. **configure terminal**

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	**configure terminal**
	
	例：
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。
		（注） CLI プロンプトが変化し、グローバル コンフィギュレーション モードになっていることが示されます。

インターフェイス コンフィギュレーション コマンド モード

グローバル コンフィギュレーション モードから開始するコンフィギュレーション モードの一例が、インターフェイス コンフィギュレーション モードです。 デバイスでインターフェイスを設定するには、インターフェイスを指定して、インターフェイスコンフィギュレーションモードを開始する必要があります。

インターフェイス単位で多数の機能をイネーブルにする必要があります。インターフェイス コンフィギュレーション コマンドは、イーサネット インターフェイスや管理インターフェイス（mgmt 0）などのデバイス上のインターフェイスの動作を変更します。

インターフェイスの設定の詳細については、『*Cisco Nexus 5000 Series NX-OS Layer 2 Switching Configuration Guide*』を参照してください。

手順の概要

1. **configure terminal**
2. **interface** *type number*

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	**configure terminal**
	
	例：
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

ステップ 2	**interface** *type number* 例： switch(config)# interface ethernet 2/2 switch(config-if)#	設定するインターフェイスを指定します。 この CLI によって、指定したインターフェイスのインターフェイス コンフィギュレーション モードが開始されます。 （注） CLI プロンプトが変化し、インターフェイス コンフィギュレーション モードになっていることが示されます。

サブインターフェイス コンフィギュレーション コマンド モード

グローバル コンフィギュレーション モードから、サブインターフェイスと呼ばれる VLAN インターフェイスを設定するコンフィギュレーションサブモードにアクセスできます。サブインターフェイス コンフィギュレーション モードでは、1 つの物理インターフェイスに複数の仮想インターフェイスを設定できます。サブインターフェイスは、プロトコルに対しては個別の物理インターフェイスとして解釈されます。

また、サブインターフェイスにより、1 つのインターフェイスで、あるプロトコルに対する複数のカプセル化を使用できます。たとえば、サブインターフェイスに VLAN を関連付ける IEEE 802.1Q カプセル化を設定できます。

サブインターフェイスの設定の詳細については、を参照してください。サブインターフェイス コマンドの詳細については、『*Cisco Nexus 7000 Series NX-OS Interfaces Command Reference*』を参照してください。

手順の概要

1. **configure terminal**
2. **interface** *type number**.subint*

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	**configure terminal**
	
	例：
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

ステップ 2	**interface** *type number.subint*
	
	例：
	`switch(config)# interface ethernet 2/2.1`
	`switch(config-subif)#`
		設定する VLAN インターフェイスを指定します。
		この CLI によって、指定した VLAN インターフェイスのサブインターフェイス コンフィギュレーション モードが開始されます。
		（注） CLI プロンプトが変化し、グローバル コンフィギュレーション モードになっていることが示されます。

コマンド モードの保存および復元

Cisco NX-OS ソフトウェアを使用すると、現在のコマンド モードを保存し、機能を設定してから、以前のコマンド モードを復元することができます。 **push** コマンドでコマンド モードを保存し、**pop** コマンドでコマンド モードを復元します。

次に、コマンド モードを保存して復元する例を示します。

```
switch# configure terminal
switch(config)# event manager applet test
switch(config-applet)# push
switch(config-applet)# configure terminal
switch(config)# username testuser password newtest
switch(config)# pop
switch(config-applet)#
```

コンフィギュレーション コマンド モードの終了

コンフィギュレーション コマンド モードを終了するには、次の作業のいずれかを行います。

手順の概要

1. **exit**
2. **end**
3. （任意） Ctrl+Z

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	**exit**
	
	例：
	`switch(config-if)# exit`
	`switch(config)#`
		現在のコンフィギュレーション コマンド モードを終了して、元のコンフィギュレーション コマンド モードに戻ります。

ステップ 2	**end**
	
	例：
	`switch(config-if)# end`
	`switch#`
		現在のコンフィギュレーション コマンド モードを終了して、EXEC モードに戻ります。

ステップ 3	Ctrl+Z
	
	例：
	`switch(config-if)# ^z`
	`switch#`
		（任意）
		現在のコンフィギュレーション コマンド モードを終了して、EXEC モードに戻ります。
		注意 有効なコマンドを入力してから、コマンドラインの最後で Ctrl+Z を使用すると、CLI によってそのコマンドが実行コンフィギュレーション ファイルに追加されます。 ほとんどの場合、**exit** または **end** コマンドを使用してコンフィギュレーション モードを終了する必要があります。

コマンド モードの概要

次の表は、主なコマンド モードに関する情報をまとめたものです。

表 *1*：コマンド モードの概要

モード	アクセス方法	プロンプト	終了方法
EXEC	ログインプロンプトから、ユーザ名とパスワードを入力します。	`switch#`	終了してログインプロンプトに戻るには、**exit** コマンドを使用します。
グローバル コンフィギュレーション	EXEC モードで、**configure terminal** コマンドを使用します。	`switch(config)#`	終了して EXEC モードに戻るには、**end** または **exit** コマンドを使用するか、Ctrl+Z を押します。
インターフェイス コンフィギュレーション	グローバル コンフィギュレーション モードで、インターフェイス コマンドを使用し、**interface** コマンドを使用してインターフェイスを指定します。	`switch(config-if)#`	終了してグローバル コンフィギュレーション モードに戻るには、**exit** コマンドを使用します。 終了して EXEC モードに戻るには、**exit** コマンドを使用するか、Ctrl+Z を押します。
サブインターフェイス コンフィギュレーション	グローバル コンフィギュレーション モードで、**interface** コマンドを使用してサブインターフェイスを指定します	`switch(config-subif)#`	終了してグローバル コンフィギュレーション モードに戻るには、**exit** コマンドを使用します。 終了して EXEC モードに戻るには、**end** コマンドを使用するか、Ctrl+Z を押します。

特殊文字

次の表に、Cisco NX-OS のテキスト文字列で特殊な意味を持つため、正規表現などの特殊なコンテキストでのみ使用する必要のある文字を示します。

表 *2*：特殊文字

文字	説明
%	パーセント
#	ポンド、ハッシュ、または番号
...	省略記号
|	縦線
< >	より小さい、またはより大きい
[ ]	角カッコ
{ }	波カッコ

キーストローク ショートカット

次の表に、EXEC モードとコンフィギュレーション モードの両方で使用できるコマンド キーの組み合わせを示します。

表 *3*：キーストローク ショートカット

キーストローク	説明
Ctrl+A	カーソルを行の先頭に移動します。
Ctrl+B	カーソルを 1 文字分だけ左に進めます。 複数行にわたってコマンドを入力するときは、←キーまたは Ctrl+B キーを繰り返し押してシステム プロンプトまでスクロールバックして、コマンド エントリの先頭まで移動できます。あるいは Ctrl+A キーを押してコマンド エントリの先頭に移動します。
Ctrl+C	コマンドを取り消して、コマンド プロンプトに戻ります。
Ctrl+D	カーソル位置にある文字を削除します。
Ctrl+E	カーソルを行の末尾に移動します。
Ctrl+F	カーソルを 1 文字分だけ右に進めます。

キーストローク	説明
Ctrl+G	コマンド ストリングを削除せずに、コマンド モードを終了して以前のコマンドモードに戻ります。
Ctrl+K	カーソル位置からコマンドラインの末尾までのすべての文字を削除します。
Ctrl+L	現在のコマンドラインを再表示します。
Ctrl+N	コマンド履歴の次のコマンドを表示します。
Ctrl+O	端末の画面をクリアします。
Ctrl+P	コマンド履歴の前のコマンドを表示します。
Ctrl+R	現在のコマンドラインを再表示します。
Ctrl+T	カーソルの場所にある文字を、カーソルの右にある文字と置き換えます。 カーソルが 1 文字右に移動します。
Ctrl+U	カーソル位置からコマンドラインの先頭までのすべての文字を削除します。
Ctrl+V	後に続くキー ストロークの特別な意味を削除します。 たとえば、正規表現に疑問符（?）を入力する前に Ctrl+V を押します。
Ctrl+W	カーソルの左にある単語を削除します。
Ctrl+X、H	入力したコマンドの履歴を一覧表示します。 このキーの組み合わせを使用するときは、Ctrl キーと X キーを同時に押して放してから、H を押します。
Ctrl+Y	バッファ内の最新のエントリを呼び出します（キーを同時に押します）。
Ctrl+Z	コンフィギュレーション セッションを終了して、EXEC モードに戻ります。 有効なコマンドを入力してから、コマンドラインの最後で Ctrl+Z を使用すると、コマンドの結果の設定がまず実行コンフィギュレーション ファイルに追加されます。

キーストローク	説明
↑キー	コマンド履歴の前のコマンドを表示します。
↓キー	コマンド履歴の次のコマンドを表示します。
→キー ←キー	コマンドストリング内でカーソルを前方または後方に移動させ、現在のコマンドを編集できるようにします。
?	使用可能なコマンドのリストを表示します。

キーストローク	説明

Tab	ワードの最初の文字を入力して Tab キーを押すと、ワードが補完されます。文字に一致するすべてのオプションが表示されます。
	タブを使用すると、次の項目名を完成させることができます。
	• コマンド名
	• ファイル システム内のスキーム名
	• ファイル システム内のサーバ名
	• ファイル システム内のファイル名
	例：
	`switch(config)# xm<Tab>`
	`switch(config)# xml<Tab>`
	`switch(config)# xml server`

	例：
	`switch(config)# c<Tab>`
	`callhome  class-map  clock  cts`
	`cdp  cli  control-plane`
	`switch(config)# cl<Tab>`
	`class-map  cli  clock`
	`switch(config)# cla<Tab>`
	`switch(config)# class-map`

	例：
	`switch# cd bootflash:<Tab>`
	`bootflash:  bootflash://sup-1/`
	`bootflash:///  bootflash://sup-2/`
	`bootflash://module-5/`
	`bootflash://sup-active/`
	`bootflash://module-6/  bootflash://sup-local/`

	例：
	`switch# cd bootflash://mo<Tab>`
	`bootflash://module-5/`
	`bootflash://module-6/cv`
	`switch# cd bootflash://module-`

コマンドの省略形

コマンドの最初の数文字を入力することで、コマンドおよびキーワードを省略できます。 省略形には、コマンドまたはキーワードを一意に識別でき得る文字数を含める必要があります。 コマンドの入力で問題が生じた場合は、システムプロンプトを確認し、疑問符（?）を入力して使用でき

るコマンドのリストを表示してください。コマンドモードが間違っているか、間違った構文を使用している可能性があります。

次の表に、コマンドの省略形の例を示します。

表 *4*：コマンド省略形の例

コマンド	省略形
configure terminal	conf t
copy running-config startup-config	copy run start
interface ethernet 1/2	int e 1/2
show running-config	sh run

部分的なコマンド名の補完

完全なコマンド名を思い出せない場合や、入力量を減らす場合は、コマンドの先頭の数文字を入力して、Tab キーを押します。コマンドライン パーサーは、入力された文字列がコマンド モードに対して一意である場合に、コマンドを補完します。キーボードに Tab キーがない場合は、代わりに Ctrl+I を押します。

コマンドは、コマンドが一意になるのに十分な文字が入力されていれば認識されます。たとえば、EXEC モードで「conf」と入力した場合、「conf」で始まるコマンドは **configure** コマンドしかないため、CLI はこのエントリを **configure** コマンドに関連付けることができます。

次の例では、Tab キーを押したとき、CLI は EXEC モードで **conf** の一意の文字列を認識します。

```
switch# conf<Tab>
switch# configure
```

コマンド補完機能を使用すると、CLI により完全なコマンド名が表示されます。CLI は、Return または Enter キーが押されるまでコマンドを実行しません。これにより、完全なコマンドが省略形によって意図したものでない場合に、コマンドを修正できます。複数のコマンドを示す可能性のある一連の文字を入力した場合は、一致するコマンドのリストが表示されます。

たとえば、**co<Tab>** と入力すると、「co」で始まる、EXEC モードで使用できるすべてのコマンドが一覧表示されます。

```
switch# co<Tab>
configure   copy
switch# co
```

コマンド エントリを補完できるように、入力した文字が再びプロンプトに表示されることに注意してください。

コマンド階層での現在の場所の識別

一部の機能においては、複数のレベルにわたる設定サブモード階層があります。 このような場合には、現在の作業コンテキスト（PWC）についての情報を表示できます。

手順の概要

1. **where detail**

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	**where detail**
	
	例：
	`switch# configure terminal`
	`switch(config)# interface mgmt0`
	`switch(config-if)# where detail`
	`mode:  conf`
	`interface mgmt0`
	`username:  admin`
		PWC を表示します。

コマンドの no 形式の使用

ほぼすべてのコンフィギュレーション コマンドには、機能をディセーブルにしたり、デフォルト値に戻したり、設定を削除したりするために使用できる **no** 形式があります。 Cisco NX-OS のコマンド リファレンスの資料では、コマンドの **no** 形式が使用できる場合は常に **no** 形式の機能について説明しています。

次の例では、機能をディセーブルにする方法を示します。

```
switch# configure terminal
switch(config)# feature tacacs+
switch(config)# no feature tacacs+
```

次の例では、機能のデフォルト値に戻す方法を示します。

```
switch# configure terminal
switch(config)# banner motd #Welcome to the switch#
switch(config)# show banner motd
Welcome to the switch

switch(config)# no banner motd
switch(config)# show banner motd
User Access Verification
```

次の例では、機能の設定を削除する方法を示します。

```
switch# configure terminal
switch(config)# radius-server host 10.10.2.2
switch(config)# show radius-server
retransmission count:0
timeout value:1
deadtime value:1
total number of servers:1

following RADIUS servers are configured:
        10.10.1.1:
                available for authentication on port:1812
                available for accounting on port:1813
        10.10.2.2:
                available for authentication on port:1812
                available for accounting on port:1813

switch(config)# no radius-server host 10.10.2.2
switch(config)# show radius-server
retransmission count:0
timeout value:1
deadtime value:1
total number of servers:1

following RADIUS servers are configured:
        10.10.1.1:
                available for authentication on port:1812
                available for accounting on port:1813
```

次の例では、EXEC モードでコマンドの **no** 形式を使用する方法を示します。

```
switch# cli var name testinterface ethernet1/2
switch# show cli variables
SWITCHNAME="switch"
TIMESTAMP="2009-05-12-13.43.13"
testinterface="ethernet1/2"

switch# cli no var name testinterface
switch# show cli variables
SWITCHNAME="switch"
TIMESTAMP="2009-05-12-13.43.13"
```

CLI 変数の設定

ここでは、Cisco NX-OS の CLI で使用する CLI 変数について説明します。

CLI 変数について

Cisco NX-OS ソフトウェアでは、CLI コマンドで変数を定義して使用することができます。

CLI 変数は次の方法で参照できます。

- コマンドラインで直接入力する。
- **run-script** コマンドを使用して開始するスクリプトに渡す。 親シェルで定義した変数は、子の **run-script** コマンド プロセスで使用できます。

CLI 変数には、次の特性があります。

- 入れ子状態の参照を使用して、別の変数から変数を参照することはできません。
- スイッチのリロードまたは現在のセッションの間だけ存在できます。

Cisco NX-OS では、事前定義された TIMESTAMP という変数が 1 つあります。 この変数は、コマンドが実行される現在の時刻を YYYY-MM-DD-HH.MM.SS という形式で参照します。

（注） TIIMESTAMP 変数名は大文字と小文字を区別します。 文字はすべて大文字です。

CLI セッションのみの変数の設定

CLI セッション変数を、CLI セッションの期間のみ保持されるように定義できます。 これらの変数は、定期的に実行するスクリプトに役立ちます。名前をカッコで囲み、その前にドル記号（$）を付加することによって、その変数を参照できます（たとえば、$(*variable-name*)）。

手順の概要

1. **cli var name** *variable-name variable-text*
2. （任意） **show cli variables**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**cli var name** *variable-name variable-text* 例： `switch# cli var name testinterface ethernet 2/1`	CLI セッション変数を設定します。 *variable-name* 引数には、大文字と小文字を区別して、最大 31 文字の英数字で値を指定します。 *variable-text* 引数は 200 文字以下の長さの英数字で、大文字と小文字を区別し、スペースを含むことができます。
ステップ 2	**show cli variables** 例： `switch# show cli variables`	（任意） CLI 変数の設定を表示します。

永続的な CLI 変数の設定

CLI セッションやデバイスのリロードをまたいで保持される CLI 変数を設定できます。

手順の概要

1. **configure terminal**
2. **cli var name** *variable-name variable-text*
3. **exit**
4. （任意） **show cli variables**
5. （任意） **copy running-config startup-config**

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	**configure terminal**
	
	例：
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

ステップ 2	**cli var name** *variable-name variable-text* 例： `switch(config)# cli var name testinterface ethernet 2/1`	CLI 固定変数を設定します。 変数名は、大文字と小文字が区別される英数字文字列で、英字で始まる必要があります。 最大長は 31 文字です。

ステップ 3	**exit**
	
	例：
	`switch(config)# exit`
	`switch#`
		グローバル コンフィギュレーション モードを終了します。

ステップ 4	**show cli variables** 例： `switch# show cli variables`	（任意） CLI 変数の設定を表示します。
ステップ 5	**copy running-config startup-config** 例： `switch(config)# copy running-config startup-config`	（任意） 実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

コマンド エイリアス

ここでは、コマンド エイリアスに関する情報を提供します。

コマンド エイリアスについて

頻繁に使用するコマンドを、コマンドエイリアスに置き換えて定義することができます。コマンド エイリアスは、コマンド構文のすべてまたは一部を表すことができます。

コマンド エイリアスには、次の特性があります。

- コマンド エイリアスはすべてのユーザ セッションに対してグローバルです。
- コマンド エイリアスをスタートアップ コンフィギュレーションに保存すると、再起動後も維持されます。
- コマンド エイリアス変換は常にすべてのコンフィギュレーション モードまたはサブモードのすべてのキーワードの中で最優先されます。
- コマンド エイリアスの設定は他のユーザ セッションに対してただちに有効になります。
- Cisco NX-OS ソフトウェアでは、デフォルトのエイリアスとして **alias** が用意されています。これは、**show cli alias** コマンドに相当し、ユーザ定義のすべてのエイリアスを表示します。
- デフォルトのコマンド エイリアスである **alias** を削除または変更することはできません。
- エイリアスは最大で 1 の深さにネストできます。1 つのコマンド エイリアスは、有効なコマンドを参照する必要がある別のコマンド エイリアスを参照できますが、その他のコマンド エイリアスは参照できません。
- コマンドエイリアスは必ず、コマンドラインの最初のコマンドキーワードを置き換えます。
- あらゆるコマンド モードで、コマンドのコマンド エイリアスを定義できます。
- コマンドエイリアスでCLI変数を参照すると、エイリアスには、変数の参照ではなく現在の変数の値が表示されます。
- 検索およびフィルタリングを実行する **show** コマンドのコマンドエイリアスを使用できます。

コマンド エイリアスの定義

一般に使用されるコマンドのコマンド エイリアスを定義できます。

手順の概要

1. **configure terminal**
2. **cli alias name** *alias-name alias-text*
3. **exit**
4. （任意） **alias**
5. （任意） **copy running-config startup-config**

手順の詳細

	コマンドまたはアクション	目的

ステップ 1	**configure terminal**
	
	例：
	`switch# configure terminal`
	`switch(config)#`
		グローバル コンフィギュレーション モードを開始します。

ステップ 2	**cli alias name** *alias-name alias-text* 例： `switch(config)# cli alias name ethint interface ethernet`	コマンド エイリアスを設定します。 エイリアス名は英数字で表します。大文字と小文字は区別されません。先頭は英字にする必要があります。 30 文字以内で指定します。

ステップ 3	**exit**
	
	例：
	`switch(config)# exit`
	`switch#`
		グローバル コンフィギュレーション モードを終了します。

ステップ 4	**alias** 例： `switch# alias`	（任意） コマンド エイリアスの設定を表示します。
ステップ 5	**copy running-config startup-config** 例： `switch# copy running-config startup-config`	（任意） 実行コンフィギュレーションを、スタートアップ コンフィギュレーションにコピーします。

ユーザ セッションのコマンド エイリアスの設定

Cisco NX-OS デバイス上の他のどのユーザからも使用できない、現在のユーザ セッションのコマンド エイリアスを作成できます。 また、現在のユーザ アカウントによる将来の使用のためにコマンド エイリアスを保存することもできます。

手順の概要

1. **terminal alias** [**persist**] *alias-name command -string*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**terminal alias** [**persist**] *alias-name command -string* 例： `switch# terminal alias shintbr show interface brief`	現在のユーザ セッションのコマンド エイリアスを設定します。 このユーザ アカウントによる使用の将来のためにエイリアスを保存するには、**persist** キーワードを使用します。 （注） **persist** キーワードを省略しないでください。

コマンド スクリプト

ここでは、複数の作業を実行するコマンドのスクリプトを作成する方法について説明します。

コマンド スクリプトの実行

コマンドのリストをファイルに記述して、CLI から実行することができます。 コマンド スクリプトでは CLI 変数を使用できます。

（注） CLI プロンプトでは、スクリプト ファイルは作成できません。 スクリプト ファイルはリモート デバイスで作成して、Cisco NX-OS デバイスの bootflash:、slot0:、または volatile: ディレクトリにコピーします。

手順の概要

1. **run-script** [**bootflash:** | **slot0:** | **volatile:**]*filename*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**run-script** [**bootflash:**	**slot0:**	**volatile:**]*filename* 例： `switch# run-script testfile`	デフォルトディレクトリでファイルに記述されたコマンドを実行します。

端末への情報のエコー

端末に情報をエコーできます。これは、コマンド スクリプトから使用すると特に有効です。 エコーされたテキストで CLI 変数を参照したり、フォーマット オプションを使用したりすることができます。

次の表に、テキストに挿入できるフォーマット オプションを示します。

表 *5*： *echo* コマンドのフォーマット オプション

フォーマット オプション	説明
\b	バック スペースが挿入されます。
\c	テキスト ストリングの最後にある改行文字が削除されます。
\f	フォーム フィード文字が挿入されます。
\n	改行文字が挿入されます。
\r	テキスト行の最初に戻ります。
\t	水平タブ文字が挿入されます。
\v	垂直タブ文字が挿入されます。
\\	バックスラッシュ文字が表示されます。
nnn	対応する ASCII 8 進文字が表示されます。

手順の概要

1. **echo** [**backslash-interpret**] [*text*]

手順の詳細

	コマンドまたはアクション	目的

ステップ **1**	**echo** [**backslash-interpret**] [*text*]
	
	例：
	`switch# echo This is a test.`
	`This is a test.`
		backslash-interpret キーワードは、テキスト文字列にフォーマット オプションが含まれていることを示します。 *text* 引数は、大文字と小文字が区別される英数字で、空白を含むことができます。 最大長は 200 文字です。 デフォルトは空白行です。

コマンド処理の遅延

コマンド処理を一定時間遅らせることができます。これは、コマンド スクリプト内で特に有効です。

手順の概要

1. **sleep** *seconds*

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**sleep** *seconds* 例： `switch# sleep 30`	数秒の遅延を発生させます。 範囲は 0 ～ 2147483647 です。

状況依存ヘルプ

Cisco NX-OS ソフトウェアには、CLI に状況依存ヘルプ機能が用意されています。 任意の箇所でコマンドに疑問符（?）を指定すると、使用できる入力オプションが一覧表示されます。

CLI では、入力エラーを隔離するためにキャレット（^）記号を使用します。 ^ 記号は、コマンド ストリング内のコマンド、キーワード、または引数が誤って入力されている位置に表示されます。

この表では、状況依存ヘルプの出力例について説明します。

表 *6*：状況依存ヘルプの例

出力例	説明

`switch# clock ?`
`  set  HH:MM:SS Current Time`
`switch# clock`
	EXEC モードで **clock** コマンドのコマンド構文を表示します。
	このスイッチの出力では、**clock** コマンドを使用するためには **set** キーワードが必要であることが示されています。

`switch# clock set ?`
`  WORD  HH:MM:SS Current Time`
`switch# clock set`
	時間を設定するためのコマンド構文を表示します。
	このヘルプの出力では、クロックの設定に現在の時刻が必要であることと、時刻の形式が示されています。

`switch# clock set 13:32:00<CR>`
`% Incomplete command`
`switch#`
	現在の時刻を追加します。
	CLI によって、コマンドが不完全であることが示されます。

`switch# <Ctrl-P>` `switch# clock set 13:32:00`	入力した前のコマンドを表示します。

`switch# clock set 13:32:00 ?`
`  <1-31>  Day of the month`
`switch# clock set 13:32:00`
	clock set コマンドの他の引数が表示されます。

`switch# clock set 13:32:00 18 ?`
`  April  Month of the year`
`  August  Month of the year`
`  December  Month of the year`
`  February  Month of the year`
`  January  Month of the year`
`  July  Month of the year`
`  June  Month of the year`
`  March  Month of the year`
`  May  Month of the year`
`  November  Month of the year`
`  October  Month of the year`
`  September  Month of the year`
`switch# clock set 13:32:00 18`
	clock set コマンドの他の引数が表示されます。

`switch# clock set 13:32:00 18 April 08<CR>` `% Invalid input detected at '^' marker.`	クロック設定に日付を追加します。 CLI によって、08 の箇所にキャレット記号（^）でエラーがあることが示されます。

`switch# clock set 13:32:00 18 April ?`
`  <2000-2030>  Enter the year (no abbreviation)`
`switch# clock set 13:32:00 18 April`
	年を指定する正しい引数が表示されます。

`switch# clock set 13:32:00 18 April 2008<CR>` `switch#`	**clock set** コマンドの正しい構文を入力します。

正規表現の概要

Cisco NX-OS ソフトウェアは、**show** コマンドなどの、CLI 出力での検索およびフィルタリングのための正規表現をサポートしています。 正規表現では大文字と小文字が区別され、また複雑な一致要件を設定することができます。

特殊文字

その他のキーボード文字（!や~など）を、単一文字パターンとして使用することもできますが、特定のキーボード文字は、正規表現で使用されると特殊な意味を持ちます。

次の表に、特殊な意味を持つキーボード文字を示します。

表 *7*：特殊な意味を持つ特殊文字

文字	特殊な意味
.	任意の 1 文字（スペースを含む）と一致します。
*	0個以上のパターンのシーケンスと一致します。
+	1個以上のパターンのシーケンスと一致します。
?	パターンの 0 または 1 回の出現と一致します。
^	文字列の最初と一致します。
$	文字列の最後と一致します。
_（アンダースコア）	カンマ（,）、左波カッコ（{）、右波カッコ（}）、左カッコ（(）、右カッコ（)）、文字列の先頭、文字列の最後、またはスペースと一致します。 （注） アンダースコアは、BPG 関連のコマンドの正規表現としてのみ扱われます。

これらの特殊文字を単一文字パターンとして使用するには、各文字の前にバックスラッシュ（\）を置くことによって、特殊な意味を削除してください。 次の例には、それぞれドル記号（$）、アンダースコア（_）、およびプラス記号（+）に一致する単一文字パターンが含まれています。

```
\$ \_ \+
```

複数文字のパターン

文字、数字、または特別な意味を持たないキーボード文字を組み合わせて、複数文字のパターンを作成することもできます。 たとえば、a4% は複数文字の正規表現です。

複数文字のパターンでは、順序が大切です。 a4% という正規表現は、a、4、パーセント記号（%）の順に並んでいる文字と一致しています。 文字列の中に a4% という文字がその順序で含まれていないと、パターン マッチングは失敗します。 複数文字正規表現 a.（文字 a の後にピリオド）は、ピリオド文字の特別な意味を使用して、文字 a の後に任意の単一文字が続くストリングと一致します。 この例では、ab、a!、または a2 という文字列がすべてこの正規表現と一致します。

特殊文字の特別な意味を無効にするには、その前にバックスラッシュを挿入します。 たとえば、表現 a\. がコマンド構文で使用されている場合、ストリング a. だけが一致します。

位置指定

特殊文字を使用すると、文字列の最初または最後の位置に対して正規表現パターンを指定することができます。

次の表に、位置指定に使用できる特殊文字を示します。

表 *8*：位置指定に使用する特殊文字

文字	説明
^	文字列の最初と一致します。
$	文字列の最後と一致します。

たとえば、正規表現 **^con** は、「con」で始まる任意の文字列と一致し、**sole$** は「sole」で終わる任意の文字列と一致します。

（注） ^記号は、角カッコで囲まれた範囲内で使用すると、論理関数の「not」を表すためにも使用できます。 たとえば、正規表現 [^abcd] は、a、b、c、または d 以外の任意の単一文字に一致する範囲を示します。

show コマンドの出力の検索およびフィルタリング

多くの場合、**show** コマンドの出力は、長くて煩雑になります。 Cisco NX-OS ソフトウェアでは、情報を簡単に見つけることができるように、出力を検索およびフィルタリングするための手段が提供されています。 検索およびフィルタリングのオプションは、**show** コマンドの末尾にパイプ記

号（|）を付け、その後に指定します。CLI の状況依存ヘルプ機能を使用してオプションを表示できます。

```
switch# show running-config | ?
  cut      Print selected parts of lines.
  diff     Show difference between current and previous invocation (creates temp files:
           remove them with 'diff-clean' command and don't use it on commands with big
           outputs, like 'show tech'!)
  egrep    Egrep - print lines matching a pattern
  grep     Grep - print lines matching a pattern
  head     Display first lines
  human    Output in human format
  last     Display last lines
  less     Filter for paging
  no-more  Turn-off pagination for command output
  perl     Use perl script to filter output
  section  Show lines that include the pattern as well as the subsequent lines that are
           more indented than matching line
  sed      Stream Editor
  sort     Stream Sorter
  sscp     Stream SCP (secure copy)
  tr       Translate, squeeze, and/or delete characters
  uniq     Discard all but one of successive identical lines
  vsh      The shell that understands cli command
  wc       Count words, lines, characters
  xml      Output in xml format (according to .xsd definitions)
  begin    Begin with the line that matches
  count    Count number of lines
  end      End with the line that matches
  exclude  Exclude lines that match
  include  Include lines that match
```

キーワードのフィルタリングおよび検索

Cisco NX-OS CLI には、**show** コマンドと併用してコマンド出力の検索やフィルタリングを実行できる、一連のキーワードが用意されています。

次の表に、CLI の出力をフィルタリングおよび検索するためのキーワードを示します。

表 *9*：キーワードのフィルタリングおよび検索

キーワードの構文	説明
begin *string* 例： **show version | begin Hardware**	検索文字列に一致するテキストが含まれる行から表示を開始します。検索文字列は、大文字と小文字が区別されます。
count 例： **show running-config | count**	コマンドの出力行数を表示します。

キーワードの構文	説明
cut [**-d** *character*] {**-b** | **-c** | **-f** | **-s**} 例： `show file testoutput | cut -b 1-10`	出力行の一部分だけを表示します。一定のバイト数（**-b**）、文字数（**-vcut** [**-d** *character*] {**-b** | **-c** | **-f** | **-s**}）、またはフィールド数（**-f**）を表示できます。 また、**-d** キーワードを使用して、デフォルトのタグ文字以外のフィールドデリミタを定義することもできます。 **-s** キーワードは、行の表示にデリミタが含まれないようにします。
end *string* 例： `show running-config | end interface`	検索文字列が最後に現れる位置まですべての行を表示します。
exclude *string* 例： `show interface brief | exclude down`	検索文字列が含まれないすべての行を表示します。 検索文字列は、大文字と小文字が区別されます。
head [**lines** *lines*] 例： `show logging logfile | head lines 50`	出力の先頭部分を、指定した行数だけ表示します。 デフォルトの行数は 10 行です。
human 例： `show version | human`	**terminal output xml** コマンドを使用して出力形式が XML に設定されている場合に、出力を通常形式で表示します。
include *string* 例： `show interface brief | include up`	検索文字列が含まれるすべての行を表示します。 検索文字列は、大文字と小文字が区別されます。
last [*lines*] 例： `show logging logfile | last 50`	出力の末尾部分を、指定した行数だけ表示します。 デフォルトの行数は 10 行です。
no-more 例： `show interface brief | no-more`	画面の下端で`--More--`プロンプトを出して停止せず、すべての出力を表示します。
sscp *SSH-connection-name filename* 例： `show version | sscp MyConnection show_version_output`	ストリーミング セキュア コピー（sscp）を使用して、出力を名前付き SSH 接続にリダイレクトします。 名前付きの SSH 接続は、**ssh name** コマンドを使用して作成できます。

キーワードの構文	説明
wc [**bytes** | **lines** | **words**] 例： `show file testoutput | wc bytes`	文字数、行数、またはワード数を表示します。デフォルトでは、行数、ワード数、文字数を表示します。
xml 例： `show version | xml`	出力を XML 形式で表示します。

diff ユーティリティ

show コマンドからの出力と、そのコマンドを以前に実行したときの出力を比較できます。

注意 **show tech-support** コマンドなど、出力が非常に長い **show** コマンドには、diff ユーティリティを使用しないでください。

diff ユーティリティの構文は次のとおりです。

diff [**--left-column**] [**-B**] [**-I**] [**-W** *columns*] [**-b**] [**-c** *lines*] [**-I**] [**-q**] [**-s**] [**-y**] [**again**] [**echo**]

次の表で、diff ユーティリティのキーワードについて説明します。

表 ***10*** : ***diff*** ユーティリティのキーワード

キーワード	説明
--left-column	横並びの形式で、共通の 2 つの行のうち左側の列だけが出力されます。
-B	空白行が挿入または削除されただけの違いを無視します。
-I	正規表現に一致する行が挿入または削除されただけの違いを無視します。
-W *columns*	横並びの形式で出力する列の幅を指定します。範囲は 0 ～ 4294967295 です。
-b	空白の数の違いを無視します。デフォルトでは空白の違いを表示します。
-c *lines*	表示されるコンテキストの行数を設定します。デフォルトの行数は 3 です。 範囲は 0 ～ 4294967295 です。

キーワード	説明
-I	大文字と小文字の違いを無視します。デフォルトでは大文字と小文字の違いを報告します。
-q	ファイルが異なることが示されますが、その違いの詳細は表示されません。デフォルトでは違いを表示します。
-s	2 つの出力が同じであるかどうかを示します。デフォルトでは、出力が同じ場合は何も表示されません。
-y	横並びの形式を使用して、出力の違いを表示します。デフォルトでは、先に古い方の出力行が表示され、その後に現在の出力行が表示されます。
again	新しい出力ファイルを作成しません。古い出力ファイルを使用して、単に表示オプションを変更し、さらにフィルタを追加します。
echo	現在のコマンドの出力をエコーします。 このキーワードは、前のコマンドの出力がない場合にのみ有効です。

Cisco NX-OS ソフトウェアは、現在および以前のすべてのユーザ セッションに対する **show** コマンドの最新の出力について、一時ファイルを作成します。 これらの一時ファイルを削除するには、**diff-clean** コマンドを使用します。

diff-clean [**all-sessions** | **all-users**]

デフォルトでは、**diff-clean** コマンドによって現在のユーザのアクティブ セッションに対する一時ファイルが削除されます。 **all-sessions** キーワードを指定すると、現在のユーザの過去および現在の全セッションに対する一時ファイルが削除されます。 **all-users** キーワードを指定すると、すべてのユーザの過去および現在の全セッションに対する一時ファイルが削除されます。

grep および egrep ユーティリティ

Global Regular Expression Print（grep）および Extended grep（egrep）コマンドライン ユーティリティを使用すると、**show** コマンドの出力をフィルタリングすることができます。

grep および egrep の構文は次のとおりです。

{**grep** | **egrep**} [**count**] [**ignore-case**] [**invert-match**] [**line-exp**] [**line-number**] [**next** *lines*] [**prev** *lines*] [**word-exp**] *expression*}]

次の表に、**grep** および **egrep** のパラメータを示します。

表 *11* : ***grep*** および ***egrep*** のパラメータ

パラメータ	説明
count	一致する行の合計数だけを表示します。
ignore-case	一致する行の大文字と小文字の違いを無視することを指定します。
invert-match	式と一致しない行を表示します。
line-exp	行が完全に一致する行だけを表示します。
line-number	一致する各行の先頭に行番号を表示することを指定します。
next *lines*	一致する行の後に表示する行数を指定します。デフォルトは 0 です。 指定できる範囲は 1 ～ 999 です。
prev *lines*	一致する行の前に表示する行数を指定します。デフォルトは 0 です。 指定できる範囲は 1 ～ 999 です。
word-exp	単語が完全に一致する行だけを表示します。
expression	出力を検索するための正規表現を指定します。

less ユーティリティ

less ユーティリティを使用すると、**show** コマンドの出力内容を一度に 1 ページずつ表示することができます。 less コマンドは : プロンプトに対して入力できます。 使用できるすべての less コマンドを表示するには、: プロンプトに対して h と入力してください。

sed ユーティリティ

ストリーム エディタ（sed）ユーティリティを使用して、次のように **show** コマンドの出力をフィルタリングしたり、操作したりすることができます。

sed *command*

command 引数には、sed ユーティリティのコマンドを指定します。

sort ユーティリティ

sort ユーティリティを使用して、**show** コマンド出力をフィルタリングできます。

sort ユーティリティの構文は次のとおりです。

sort [**-M**] [**-b**] [**-d**] [**-f**] [**-g**] [**-i**] [**-k** *field-number*[**.***char-position*][*ordering*]] [**-n**] [**-r**] [**-t** *delimiter*] [**-u**]
次の表に、sort ユーティリティのパラメータを示します。

表 *12* : ***sort*** ユーティリティのパラメータ

パラメータ	説明
-M	月でソートします。
-b	先頭の空白（スペース文字）を無視します。デフォルトのソートには、先頭の空白が含まれます。
-d	空白と英数字のみを比較することによってソートします。デフォルトのソートには、すべての文字が含まれます。
-f	小文字を大文字にします。
-g	一般的な数値を比較することによってソートします。
-i	印刷可能文字のみを使用してソートします。デフォルトのソートには、印刷不可能な文字が含まれます。
-k *field-number*[**.***char-position*][*ordering*]	キー値に従ってソートします。 デフォルトのキー値はありません。
-n	数値文字列の値に従ってソートします。
-r	ソート結果の順序を逆にします。デフォルトのソート出力は昇順です。
-t *delimiter*	指定されたデリミタを使用してソートします。デフォルトのデリミタはスペース文字です。
-u	ソート結果から重複した行を削除します。 ソート出力は重複した行を表示します。

--More-- プロンプトからの検索およびフィルタリング

show コマンドの出力で、--More-- プロンプトから出力を検索およびフィルタリングできます。

次の表に、--More-- プロンプトのコマンドを示します。

表 *13*： *--More--* プロンプトのコマンド

コマンド	説明
[*lines*]<スペース>	出力行を指定された行数または現在の画面サイズだけ表示します。
[*lines*]**z**	出力行を指定された行数または現在の画面サイズだけ表示します。 *lines* 引数を使用すると、その値が新しいデフォルトの画面サイズになります。
[*lines*]<リターン>	指定した行数または現在のデフォルトの行数だけ出力行を表示します。 初期デフォルトは1行です。 オプションの *lines* 引数を使用すると、その値が、このコマンドで表示する新しいデフォルトの行数になります。
[*lines*]**d** または [*lines*]Ctrl+Shift+D	指定した行数または現在のデフォルトの行数だけ出力行をスクロールします。 初期デフォルトは 11 行です。 オプションの *lines* 引数を使用すると、その値が、このコマンドで表示する新しいデフォルトの行数になります。
q または **Q** または Ctrl+C	--More-- プロンプトを終了します。
[*lines*]**s**	出力内の指定された行数または現在のデフォルトの行数だけ前方にスキップし、1 画面分の行を表示します。 デフォルトは 1 行です。
[*lines*]**f**	出力内の指定された画面数または現在のデフォルトの画面数だけ前方にスキップし、1 画面分の行を表示します。 デフォルトは1画面です。
=	現在の行番号を表示します。

コマンド	説明
[*count*]/*expression*	正規表現に一致する行にスキップし、1 画面分の出力行を表示します。 式が複数回出現する行を検索するには、オプションの *count* 引数を使用します。 このコマンドでは、他のコマンドで使用できる現在の正規表現が設定されます。
[*count*]**n**	現在の正規表現に一致する次の行にスキップし、1 画面分の出力行を表示します。 一致を通り越してスキップするには、オプションの *count* 引数を使用します。
{**!** | **:!**[*shell-cmd*]}	*shell-cmd* 引数で指定されたコマンドをサブシェルで実行します。
.	前のコマンドを繰り返します。

コマンド履歴の使用

Cisco NX-OS ソフトウェアの CLI を使用すると、現在のユーザ セッションのコマンド履歴にアクセスできます。 変更を加えて、または変更なしでコマンドを呼び出したり、再発行したりできます。 また、コマンド履歴をクリアすることもできます。

コマンドの呼び出し

コマンド履歴の中のコマンドを呼び出し、任意に変更を加えて、再入力することができます。

次に、コマンドを呼び出して再入力する例を示します。

```
switch(config)# show cli history
0  11:04:07   configure terminal
1  11:04:28   show interface ethernet 2/24
2  11:04:39     interface ethernet 2/24
3  11:05:13       no shutdown
4  11:05:19     exit
5  11:05:25   show cli history
switch(config)# !1
switch(config)# show interface ethernet 2/24
```

Ctrl+P と Ctrl+N のキーストローク ショートカットを使用してコマンドを呼び出すこともできます。

CLI の履歴呼び出しの制御

Ctrl+P と Ctrl+N のキーストローク ショートカットを使用して CLI の履歴から呼び出すコマンドを制御できます。 デフォルトでは、Cisco NX-OS ソフトウェアは、現在のコマンド モードおよびそれ以上のコマンド モードのすべてのコマンドを呼び出します。 たとえば、グローバル コンフィギュレーション モードで作業している場合は、コマンド呼び出しのキーストローク ショートカットによって、EXEC モードとグローバル コンフィギュレーション モードの両方のコマンドが呼び出されます。 **terminal history no-exec-in-config** コマンドを使用すると、コンフィギュレーション モードにいるときに EXEC モード コマンドの呼び出しを回避できます。

手順の概要

1. [**no**] **terminal history no-exec-in-config**

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	[**no**] **terminal history no-exec-in-config** 例： `switch# terminal history no-exec-in-config`	コンフィギュレーション モードで呼び出しのキーストローク ショートカットが使用されたときに EXEC コマンドを削除するように CLI の履歴を設定します。 デフォルトでは、EXEC コマンドが呼び出されます。 コマンドの **no** 形式を使用して、デフォルトに戻すことができます。

CLI の編集モードの設定

Ctrl+P と Ctrl+N のキーストローク ショートカットを使用して CLI の履歴からコマンドを呼び出し、再発行する前にそれらのコマンドを編集できます。 デフォルトの編集モードは emacs です。 編集モードを vi に変更できます。

手順の概要

1. [**no**] **terminal edit-mode vi** [**persist**]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	[**no**] **terminal edit-mode vi** [**persist**] 例： `switch# terminal edit-mode vi`	そのユーザ セッションについて、CLI の編集モードを vi に変更します。 **persist** キーワードを使用すると、現在のユーザ名の設定がセッションをまたいで保持されます。

	コマンドまたはアクション	目的
		emacs の使用に戻すには、**no** を使用します。

コマンド履歴の表示

show cli history コマンドを使用して、コマンド履歴を表示できます。

show cli history コマンドの構文は次のとおりです。

show cli history [*lines*] [**unformatted**]

show cli history [*lines*] [**config-only** | **exec-only** | **this-mode-only**] [**unformatted**]

デフォルトでは、表示される行数は12行で、コマンドの番号とタイムスタンプが出力されます。

次に、コマンド履歴をデフォルトの行数だけ表示する例を示します。

```
switch# show cli history
```

次に、コマンド履歴を 20 行表示する例を示します。

```
switch# show cli history 20
```

次に、コマンド履歴の中のコンフィギュレーション コマンドだけを表示する例を示します。

```
switch(config)# show cli history config-only
```

次に、コマンド履歴の中の EXEC コマンドだけを表示する例を示します。

```
switch(config)# show cli history exec-only
```

次に、現在のコマンド モードに関するコマンド履歴のコマンドだけを表示する例を示します。

```
switch(config-if)# show cli history this-mode-only
```

次に、コマンド番号とタイムスタンプを表示せず、コマンド履歴のコマンドだけを表示する例を示します。

```
switch(config)# show cli history unformatted
```

CLI の確認プロンプトのイネーブルまたはディセーブル

Cisco NX-OS ソフトウェアでは、多くの機能において、処理を続行する前に確認を求めるプロンプトが CLI に表示されます。 これらのプロンプトをイネーブルまたはディセーブルにすることができます。 デフォルトはイネーブルです。

手順の概要

1. [**no**] **terminal dont-ask** [**persist**]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	[**no**] **terminal dont-ask** [**persist**] 例： `switch# terminal dont-ask`	CLI の確認プロンプトをディセーブルにします。 **persist** キーワードを使用すると、現在のユーザ名の設定がセッションをまたいで保持されます。 デフォルトはイネーブルです。 CLI 確認プロンプトをイネーブルにするには、コマンドの **no** 形式を使用します。

CLI の表示色の設定

次のように、表示される CLI の色を変更できます

- 前のコマンドの処理が成功した場合は、プロンプトを緑色で表示する。
- 前のコマンドの処理が失敗した場合は、プロンプトを赤色でする。
- ユーザの入力は青色で表示する。
- コマンドの出力はデフォルトの色で表示する。

デフォルトの色は、ターミナル エミュレータ ソフトウェアによって送信される色です。

手順の概要

1. **terminal color** [**evening**] [**persist**]

手順の詳細

	コマンドまたはアクション	目的
ステップ 1	**terminal color** [**evening**] [**persist**] 例： `switch# terminal color`	端末セッションの CLI の表示色を設定します。 **evening** キーワードはサポートされません。 **persist** キーワードを使用すると、現在のユーザ名の設定がセッションをまたいで保持されます。デフォルト設定は永続的ではありません。

モジュールへのコマンドの送信

slot コマンドを使用して、スーパーバイザ モジュール セッションからモジュールにコマンドを直接送信できます。

slot の構文は次のとおりです。

slot *slot-number* [**quoted**] *command-string*

デフォルトでは、*command-string* 引数のキーワードと引数はスペースで区切られます。 モジュールに複数のコマンドを送信するには、スペース文字、セミコロン（;）、スペース文字でコマンドを区切ります。

quoted キーワードは、コマンド ストリングの先頭と末尾に二重引用符（"）が使用されることを示します。 スーパーバイザ モジュール セッションでだけサポートされている diff などのフィルタリング ユーティリティにモジュール コマンド出力をリダイレクトする場合は、このキーワードを使用します。

次に、モジュール情報を表示したり、フィルタリングしたりする例を示します。

```
switch# slot 2 show version | grep lc
```

次に、スーパーバイザ モジュール セッションに関するモジュール情報をフィルタリングする例を示します。

```
switch# slot 2 quoted "show version" | diff
switch# slot 4 quoted "show version" | diff -c
*** /volatile/vsh_diff_1_root_8430_slot__quoted_show_version.old       Wed Apr 29 20:10:41
 2009
--- -   Wed Apr 29 20:10:41 2009
***************
*** 1,5 ****
! RAM 1036860 kB
! lc2
  Software
    BIOS:      version 1.10.6
    system:    version 4.2(1) [build 4.2(0.202)]
--- 1,5 ----
! RAM 516692 kB
! lc4
  Software
    BIOS:      version 1.10.6
    system:    version 4.2(1) [build 4.2(0.202)]
***************
*** 12,16 ****
  Hardware
      bootflash: 0 blocks (block size 512b)

!    uptime is 0 days 1 hours 45 minute(s) 34 second(s)

--- 12,16 ----
  Hardware
      bootflash: 0 blocks (block size 512b)

!    uptime is 0 days 1 hours 45 minute(s) 42 second(s)
```

BIOS ローダー プロンプト

スーパーバイザモジュールに電源が投入されると、特殊なBIOSイメージが自動的にロードされ、システムを起動するための有効なキックスタートイメージを見つけようとします。有効なキックスタート イメージが見つからない場合は、次の BIOS ローダー プロンプトが表示されます。

```
loader>
```

Cisco NX-OS ソフトウェアを `loader>` プロンプトからロードする方法については、『*Cisco Nexus 5000 Troubleshooting Guide*』を参照してください。

CLI の使用例

ここでは、CLI の使用例について説明します。

コマンド エイリアスの定義

次に、コマンド エイリアスを定義する方法の例を示します。

```
cli alias name ethint interface ethernet
cli alias name shintbr show interface brief
cli alias name shintupbr shintbr | include up | include ethernet
```

次に、コマンド エイリアスを使用する方法の例を示します。

```
switch# configure terminal
switch(config)# ethint 2/3
switch(config-if)#
```

CLI セッション変数の使用

構文 **$(***variable-name***)** を使用して変数を参照できます。
次の例では、ユーザ定義の CLI セッション変数を参照する方法を示します。

```
switch# show interface $(testinterface)
Ethernet2/1 is down (Administratively down)
  Hardware is 10/100/1000 Ethernet, address is 0000.0000.0000 (bia 0019.076c.4dac)
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA
  auto-duplex, auto-speed
  Beacon is turned off
  Auto-Negotiation is turned on
  Input flow-control is off, output flow-control is off
  Auto-mdix is turned on
  Switchport monitor is off
  Last clearing of "show interface" counters never
  5 minute input rate 0 bytes/sec, 0 packets/sec
  5 minute output rate 0 bytes/sec, 0 packets/sec
  L3 in Switched:
```

```
    ucast: 0 pkts, 0 bytes - mcast: 0 pkts, 0 bytes
  L3 out Switched:
    ucast: 0 pkts, 0 bytes - mcast: 0 pkts, 0 bytes
  Rx
    0 input packets 0 unicast packets 0 multicast packets
    0 broadcast packets 0 jumbo packets 0 storm suppression packets
    0 bytes
  Tx
    0 output packets 0 multicast packets
    0 broadcast packets 0 jumbo packets
    0 bytes
    0 input error 0 short frame 0 watchdog
    0 no buffer 0 runt 0 CRC 0 ecc
    0 overrun  0 underrun 0 ignored 0 bad etype drop
    0 bad proto drop 0 if down drop 0 input with dribble
    0 input discard
    0 output error 0 collision 0 deferred
    0 late collision 0 lost carrier 0 no carrier
    0 babble
    0 Rx pause 0 Tx pause 0 reset
```

システム定義のタイムスタンプ変数の使用

次の例では、**show** コマンドの出力をファイルにリダイレクトするときに $(TIMESTAMP) を使用します。

```
switch# show running-config > rcfg.$(TIMESTAMP)
Preparing to copy....done
switch# dir
       12667     May 01 12:27:59 2008  rcfg.2008-05-01-12.27.59

Usage for bootflash://sup-local
8192 bytes used
20963328 bytes free
20971520 bytes total
```

コマンド スクリプトの実行

次に、スクリプト ファイルで指定する CLI コマンドの例を示します。

```
switch# show file testfile
configure terminal
interface ethernet 2/1
no shutdown
end
show interface ethernet 2/1
```

次に **run-script** コマンドの実行の出力例を示します。

```
switch# run-script testfile
`configure terminal`
`interface ethernet 2/1`
`no shutdown`
`end`
`show interface ethernet 2/1 `
Ethernet2/1 is down (Link not connected)
  Hardware is 10/100/1000 Ethernet, address is 0019.076c.4dac (bia 0019.076c.4dac)
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA
  Port mode is trunk
```

```
auto-duplex, auto-speed
Beacon is turned off
Auto-Negotiation is turned on
Input flow-control is off, output flow-control is off
Auto-mdix is turned on
Switchport monitor is off
Last clearing of "show interface" counters 1d26.2uh
5 minute input rate 0 bytes/sec, 0 packets/sec
5 minute output rate 0 bytes/sec, 0 packets/sec
Rx
  0 input packets 0 unicast packets 0 multicast packets
  0 broadcast packets 0 jumbo packets 0 storm suppression packets
  0 bytes
Tx
  0 output packets 0 multicast packets
  0 broadcast packets 0 jumbo packets
  0 bytes
  0 input error 0 short frame 0 watchdog
  0 no buffer 0 runt 0 CRC 0 ecc
  0 overrun  0 underrun 0 ignored 0 bad etype drop
  0 bad proto drop 0 if down drop 0 input with dribble
  0 input discard
  0 output error 0 collision 0 deferred
  0 late collision 0 lost carrier 0 no carrier
  0 babble
  0 Rx pause 0 Tx pause 0 reset
```

CLI に関する追加情報

ここでは、CLI に関するその他の情報について説明します。

CLI の関連資料

関連項目	参照先
Cisco NX-OS のライセンス	『*Cisco NX-OS Licensing Guide*』
コマンド リファレンス	『*Cisco Nexus 7000 Series NX-OS Fundamentals Command Reference*』『*Cisco Nexus 5000 Series NX-OS Command Reference*』