SEIKO

運用ガイド
Operation Guide

SmartCS REST API Operation Guide

コンソールサーバー

NS-2260

第 1 版
2025 年 5 月 16 日
U00151172801

セイコーソリューションズ株式会社

目次

1　章　はじめに

1.1　本ドキュメントについて

本ドキュメントは SmartCS NS-2260 の REST API 機能を使う場合に必要となる情報をまとめた運用ガイドとなります。SmartCS の設定や各 API リソースの仕様、使用方法などを記載していますので、REST API 機能を使う場合にご利用下さい。

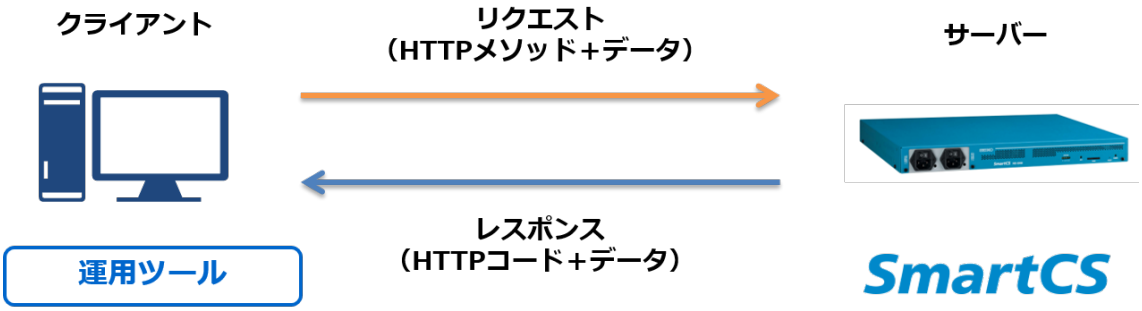
また、本ドキュメントは SmartCS の REST API 機能のみを取り扱ったドキュメントとなっております。SmartCS の各種設定や、CLI コマンドの詳細についてはそれぞれ「取扱説明書」、「コマンドリファレンス」、に詳細な記載がありますので、そちらも必要に応じて参照下さい。

1.2　機能概要

SmartCSのREST　API機能の概要について説明します。SmartCSはこれまでtelnetや SSH で装置にログイン後 CLI コマンドを実行して設定や情報取得、また SmartCS に接続している監視対象機器のオペレーションを行っていました。REST API 機能を使う事によって、様々なクライアントやツールから各機能（API リソース毎の処理）を実行する事ができます。

・SmartCS の情報や、設定内容の取得
・SmartCS の設定
・SmartCS のシリアルポートに接続している監視対象機器の TTY ログ情報の取得/検索
・SmartCS に接続している監視対象機器へのオペレーション実行

REST API は HTTP プロトコルを使って通信を行い、SmartCS の提供する API リソースに対して、各メソッドと送信データをクライアントからリクエストして送信して様々なオペレーションを指定します。SmartCS は、レスポンスとして HTTP コードと指定されたオペレーションに応じたデータをクライアントに返信します。



＜リクエスト時の、HTTP メソッドと CLI オペレーションのイメージ＞

HTTP メソッド	意味	CLI コマンド例
GET	API リソースの取得	show
POST	API リソースの作成等	create
PUT	API リソースの変更や更新	set, unset
DELETE	API リソースの削除	delete

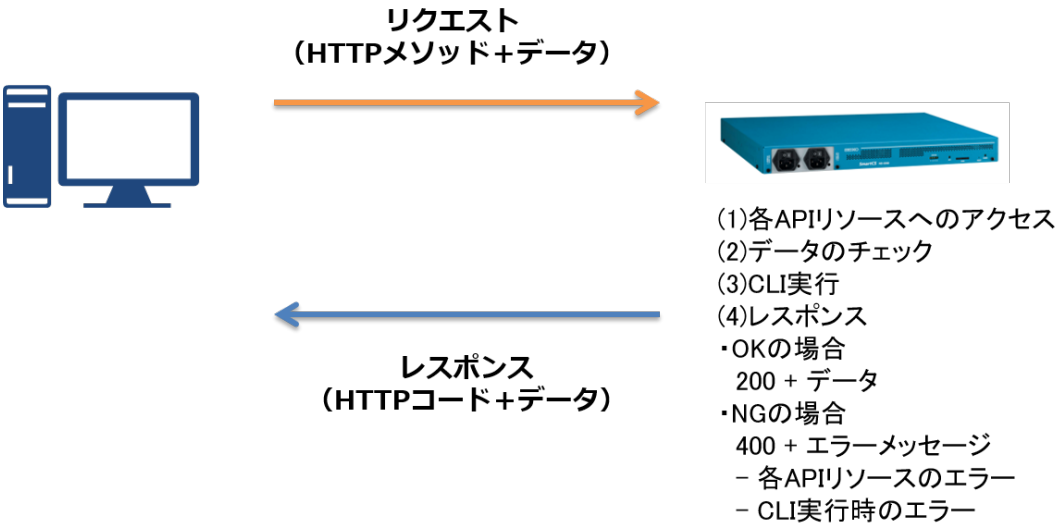
＜レスポンス時の HTTP コード例＞

HTTP コード	意味	内容
200	OK	オペレーションの成功
400	不正なリクエストなど	オペレーションの失敗

1.3 SmartCS の処理概要

SmartCS の REST API 機能の処理概要について説明します。REST API 機能ではクライアントから SmartCS が提供する各 API リソースにアクセスを行い、リクエスト内容に応じて機能毎のレスポンスを返します。

REST API のリクエストを受信した際の SmartCS の機能の処理イメージは、以下の図及び表の内容となります。



処理	内容
各 API リソースへのアクセス	クライアントからのリクエストに応じて各 API リソースにアクセスします。その際に、正しいメソッドが指定されているか、適切なユーザー権限が付与されているかなども確認します。
データチェック	リクエスト時のデータが各 API リソースの仕様に対して正しいかどうかをチェックします。
CLI コマンドの実行	リクエストデータをもとに、CLI コマンドを実行します。
レスポンス	CLI コマンドの実行結果に応じたレスポンスを返します。

リクエスト受信後に各 CLI 機能が動作して各機能を提供します。本ドキュメントでは各 CLI コマンドの詳細な仕様については説明しておらず、各 API リソースの仕様についてを説明しております。もしオペレーションがうまくいかない場合などは、CLI のエラーメッセージが出力される場合もある為、「コマンドリファレンス」についても参照下さい。

1.4 動作要件

SmartCS の REST API 機能は、バージョン 1.0 から対応しております。各バージョンの API リソース仕様については以下の表の通りとなります。

SmartCS のバージョン	ベース URL
V1.0	http://<IP アドレス>:<http ポート番号>/api/v1/

ベース URL 以降の各アドレスについては、「4 章 API リソースとメソッド」を参照してください。

2　章　準備

2.1　SmartCS の準備

2.1.1 REST API 機能の有効化・無効化

REST API 機能を使う為の SmartCS の設定について、下記に記載します。

(1) HTTP / HTTPS 機能の有効化

SmartCS の HTTP / HTTPS 機能を有効化します。

HTTP 機能を有効化する場合

```
(0)NS-2260# enable http
(0)NS-2260#
```

HTTPS 機能を有効化する場合

```
(0)NS-2260# enable https
(0)NS-2260#
```

HTTP/HTTPS のポート番号を設定する場合

```
(0)NS-2260# set http port 20080
(0)NS-2260# set https port 30443
(0)NS-2260#
```

設定が有効化されたかどうかを確認する場合は、show service で指定した設定が有効化(enable)されているか、ポート番号が変更されているか確認して下さい。また、show tcp コマンドで指定したサービスの TCP ポートがオープンしているかご確認下さい。

HTTP/HTTPS に REST API 機能を使ってアクセスできるセッション数は、最大で同時 8 セッションとなります。

(2) TTY マネージ機能の有効化

/ttymange API リソースにアクセスして、SmartCS に接続されている監視対象機器に対してオペレーションを行う場合は、TTY マネージ機能を有効化する必要があります。

TTY マネージ機能を有効化する場合

```
(0)NS-2260# enable ttymanage
(0)NS-2260#
```

2.1.2 ユーザーの作成

SmartCS の REST API 機能にアクセスする為のユーザーを作成する手順について説明します。REST API 機能の提供する各 API リソースにアクセスする為のユーザーは、拡張ユーザーグループ（extusr）に所属しているユーザーとなります。

拡張ユーザーグループに所属しているユーザーは、SSH からのみアクセスする事ができます。telnet、console 経由ではアクセスができません。

■SmartCS のユーザーグループと実行できる機能について

ユーザーグループ	内容
normal root	SmartCS に telnet, ssh, console から接続して装置の設定や情報取得を行う事ができるユーザーグループです。
portusr	SmartCS のシリアルポートに接続されている監視対象機器に telnet, ssh から接続してオペレーションを行う事ができるユーザーグループです。
extusr	設定により権限を付与することが出来るユーザーグループ。権限を付与していない場合でも normal グループユーザーと同様の権限があります。付与できる権限は root グループのユーザーと同じコマンド実行権限（管理者権限）と tty マネージオブジェクトのコマンド実行権限(tty マネージ機能権限)になります。 ■normal API リソースが提供する機能の中で、CLI の normal ユーザー権限で実行できる機能（show などの状態表示系コマンド）にアクセスできます。 ■root API リソースが提供する機能の中で、CLI の管理者権限で実行できる機能（create, set, unset,delete などの設定系コマンド）にアクセスできます。 ■ttymanage API リソースが提供する機能の中で、シリアルポートに接続されている監視対象機器の送受信ログの取得や検索、オペレーションを実現する機能にアクセスできます。
setup verup log	SmartCS に FTP, SFTP(SSH)で接続を行い、コンフィグファイル、バージョンアップファイル、ログファイルを送受信する事ができるユーザーグループです。

(1) 拡張ユーザーグループのユーザーを作成

拡張ユーザーグループに所属するユーザーを作成します。

```
(0)NS-2260# create user api group extusr password
New password: xxxxx
Retype new password: xxxxx
(0)NS-2260#
```

(2) 管理者権限を設定する場合

作成した拡張ユーザーグループのユーザーに、管理者権限を設定します。

```
(0)NS-2260# set user api permission root on
(0)NS-2260#
```

(3) tty マネージ機能権限を設定し、アクセス可能なポート番号を許可する場合

作成した拡張ユーザーグループのユーザーに、tty マネージ機能権限を設定して、アクセス可能なポート番号(例:1-4,16)を許可します。

```
(0)NS-2260# set user api permission ttymanage on
(0)NS-2260# set user api port 1-4,16
(0)NS-2260#
```

2.1.3 セキュリティー

REST API 機能を使う場合、必要に応じてセキュリティー強化の為 Firewall(ipfilter)機能を設定してご利用してください。

HTTP/HTTPS に設定されている TCP ポート番号を指定する事で、アクセス制限を行う事ができます。

2.2 ログ情報

2.2.1 REST API のアクセスログ

REST API 機能を使ってアクセスしたユーザーについては show log webapi コマンドで確認する事ができます。

最新の 5 件のアクセスログを確認する場合

```
(0)NS-2260#  show log webapi 5
2025 Mar 10 15:32:49 [10080] login success: restapi/172.31.8.41:42266
2025 Mar 10 15:32:49 [10080] logout: restapi/172.31.8.41:42266
2025 Mar 10 15:32:49 [10080] login success: restapi/172.31.8.41:42268
2025 Mar 10 15:32:49 [10080] logout: restapi/172.31.8.41:42268
2025 Mar 11 23:49:03 [10080] FAILED LOGIN FROM 172.31.8.41 FOR api,
Authentication failure.
(0)NS-2260#
```

アクセスしたユーザー名、IP アドレス、接続元のポート番号や、認証が成功したか、失敗したかについて確認する事ができます。

2.2.2 REST API のオペレーションログ

SmartCS の REST API 機能はリクエスト内容を最終的に CLI で実行します。その為、各 API リソースにアクセスした際に実行したコマンドについては、show log command コマンドで確認する事が出来ます。

/system/version API リソースに api ユーザーが REST API 機能でアクセスした場合

```
(0)NS-2260#  show log command 5
2025 Mar 11 23:55:08 somebody: show tty
2025 Mar 11 23:55:11 somebody: show user
2025 Mar 11 23:55:12 somebody: su
2025 Mar 11 23:55:20 api: show json version
2025 Mar 11 23:55:26 root: show log command 5
(0)NS-2260#
```

REST API 機能を使って、api ユーザーが、/system/version API リソースにアクセスした際に、show json version が実行されている事が確認できます。

3　章　REST API 機能

3.1　リクエスト

3.1.1 ベース URL

SmartCS の REST API 機能にアクセスする ベース URL は以下のアドレスとなります。

プロトコル	ベース URL
HTTP	http://<IP アドレス>:<http ポート番号>/api/v1/
HTTPS	https://<IP アドレス>:<https ポート番号>/api/v1/

IP アドレスは、v4/v6 の両方をサポートしています。

HTTP のポート番号のデフォルト値は 10080、HTTPS のポート番号のデフォルト値は 10443 となります。ポート番号はそれぞれ、1025～65000 の範囲で変更が可能です。

SmartCS の IP アドレス変更、HTTP/HTTPS のポート番号の変更方法については、取扱説明書 または コマンドリファレンスを参照してください。

3.1.2 HTTP メソッド

SmartCS の REST API 機能は、以下の HTTP メソッドをサポートしています。

メソッド	説明
GET	指定した API リソースから情報取得する要求を行います。 SmartCS の CLI コマンドの、show 系コマンドのオペレーションとなります。
POST	指定した API に新たなリソースを作成する要求を行います。また、特定のオペレーションの実行についての要求を行います。 SmartCS の CLI コマンドの、create コマンドなどのオペレーションとなります。
PUT	指定した API リソースについて変更/修正する要求を行います。 SmartCS の CLI コマンドの、set/unset コマンドなどのオペレーションとなります。
DELETE	指定した API リソースについて削除する要求を行います。 SmartCS の CLI コマンドの、delete コマンドなどのオペレーションとなります。

API リソース毎に指定できるメソッドは異なり、未サポートのメソッドを指定した場合はエラーとなります。詳細は「4 章 API リソースとメソッド」を参照してください。

3.1.3 パラメーター

各メソッドはリクエスト時にパラメーターを指定する事ができます。

リクエスト	内容
GET クエリー	URL のクエリーとしてオプション値をパラメーターとして指定する事ができます。 例: `http://<IP>:<PORT>/api/v1/users/{username}` {username} 部分が GET クエリーとして指定できるオプションデータとなります。指定できる内容は各 API リソースによって異なります。
リクエストボディー	リクエストボディーとして、JSON 形式のオプション値をパラメーターとして指定する事ができます。その際、HTTP ヘッダーのコンテンツタイプ（Content-Type）として、 application/json を指定してください。

詳細は「4 章 API リソースとメソッド」を参照してください。

3.2 認証

3.2.1 Basic 認証

SmartCS の REST API 機能はリクエスト時の認証として Basic 認証をサポートしています。Authorization ヘッダーに、ユーザー名（username）とパスワード（password）のペアを指定してください。SmartCS に登録されている拡張ユーザーが対象となります。

API リソース毎にアクセス可能なユーザー権限が異なります、詳細は「2.3 項 ユーザー権限と API リソース」を参照してください。

3.3　レスポンス

3.3.1 ステータスコード

SmartCS の REST API 機能は各 API リソースへのアクセス後、レスポンスとして以下のステータスコードを返します。

ステータスコード	意味	概要
200	OK	REST API のリクエストが正常に処理できた場合のステータスコードとなります。
400	Bad Request	REST API のリクエストがなんらかのエラーによって正常に処理できなかった場合のステータスコードとなります。エラー理由については、エラー時に返す JSON データの “message” を確認してください。 各エラーの詳細については「3.4 共通エラー」項を参照ください。

3.3.2 共通データ

各 API リソースへのアクセス後、ステータスコードとともに、JSON 形式のデータをレスポンスとして返します。JSON 形式のデータは共通データと各 API リソース独自のレスポンスデータに分かれており、共通データは “info” をキー としたオブジェクト型で格納されています。

＜共通データ＞

info	型	意味
result	数値	リクエストが正常に処理された場合は 0 エラーが発生した場合は 1 以上の値を返します
message	文字列	エラーメッセージが格納されます。 リクエストが正常に処理された場合、空文字列(“”)が格納されます。 エラーメッセージ例 ・認証やアクセス権限に関する各 API リソース 共通で発生するエラー 「3.4 共通エラー」項を参照下さい。 ・各 API リソースのパラメーターや、API リソースを介して 実行される CLI エラー 「4 章 API リソース」項の各 URL のエラーを 参照下さい。

例: リクエストが正常に処理された場合

```
{
    "info": {
        "result": 0,
        "message": ""
    },
    "systeminfo": {
        "Boot": {
            "System": {
                "Version": "1.0AH",
                "Build": "2025-03-10",
```

共通データ

各 API リソースの
レスポンスデータ

例：リクエストが正常に処理されなかった場合

```
{
    "info": {
        "result": 1,
        "message": "Error: Invalid request. "
    }
}
```

共通データ

3.4 共通エラー

各 API リソース共通で発生するエラー内容と、エラーが発生した場合の対処方法について説明します。

メッセージ	内容と対処
Error: Invalid request.(400)	指定した API リソースにアクセスする為の情報に不備がある場合に表示されます。
Error: Invalid request.(404)	指定した API リソースに存在しない URL を指定した場合に表示されます。
Error: Invalid request.(405)	指定した API リソースがサポートしていないメソッドを指定した場合に表示されます。
Error: Invalid request.(411)	リクエスト情報に必要なヘッダー情報（content-length）が含まれていない場合に表示されます。
Error: Invalid request.(417)	リクエスト情報に、未サポートのヘッダー情報（Expect）が含まれている場合に表示されます。
Error: Invalid request.(500)	リクエストを受信後、SmartCS の HTTP/HTTPS サーバーでエラーが発生した場合に表示されます。
Error: Invalid request.(501)	サポート外のメソッドを指定された場合に表示されます。

メッセージ	内容と対処
Error: Invalid request.(601)	リクエスト情報のヘッダーに必要なパラメーター（Authorization)が含まれていない場合に表示されます。
Error: Invalid request.(602)	リクエスト情報のヘッダーで指定したユーザーが存在しない、または指定した API リソースにアクセスする為の設定（権限）が与えられていない場合などに表示されます。指定しているユーザー情報や SmartCS のユーザー設定を確認してください。
Error: Invalid request.(901)	リクエストを受信後、SmartCS の Web サーバーでエラーが発生した場合に表示されます。

これらのエラーが発生した場合、各 API リソースの仕様を参考に、リクエスト情報（URL や指定しているオプションパラメーター）を確認してください。

4　章　API リソースとメソッド

SmartCS の REST API 機能が提供するリソース一覧となります。

本章では、各 API リソースの仕様(概要、リクエスト、レスポンス、エラー、実行例)について説明します。

分類	URL	メソッド	概要
SYSTEM	/system/version	GET	システム情報の取得
USERS	/users	GET	ユーザー情報（一覧）の取得
		POST	ユーザー作成
	/users/{username}	GET	ユーザー情報の取得
		PUT	ユーザー情報の編集
		DELETE	ユーザー削除
	/users/login	GET	ログインユーザー情報の取得
SERIAL	/serial/tty	GET	シリアル情報(一覧)の取得
	/serial/tty/{ttylist}	GET	シリアル情報の取得
		PUT	シリアル情報の編集
	/serial/hangup/tty/{ttylist}	POST	シリアルの hangup
TTYMANAGE	/ttymanage	POST	TTY マネージ機能を使ってシリアルポートに文字列の送受信スクリプトを実行
LOG/HISTORY	/log/history/console	GET	SmartCS のコンソールログ情報の取得
	/log/history/command	GET	SmartCS のコマンドログ情報の取得
	/log/history/ttysend	GET	SmartCS の ttysend ログ情報の取得
	/log/history/webapi	GET	SmartCS の webapi ログ情報の取得
LOG/SERIAL	/log/serial/tty/{ttyno}	GET	SmartCS の tty ログ情報の取得
	/log/serial/files/tty/{ttyno}	GET	SmartCS の tty ログ情報の取得 (DL)
	/log/serial/search/tty/{ttyno}	GET	SmartCS の tty ログ情報を検索

4.1 SYSTEM

4.1.1 /system/version (GET)

4.1.1.1 概要

システム情報を取得します。

4.1.1.2 リクエスト

項目	内容
アクセス可能なユーザー権限	normal root
オプションパラメーター	この API リソースは指定可能なオプションはありません。

4.1.1.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー名				内容
systeminfo	Boot	System	Version	起動しているシステムソフトウェアバージョン
			Build	起動しているシステムソフトウェア作成日
			Unit	起動しているシステム面
		Status		起動種別
		Config	Unit	起動しているスタートアップの保存面
			Startup	起動しているスタートアップの番号
		ROM	Version	BootROM バージョン
	SystemUpTime			システム起動時刻
	HW	Model		モデル名
		SerialNo		シリアル番号
		MAC	Local_Address	イーサネットアドレス
			Number	イーサネットアドレスの個数
		MainBoardCPU_Model		メインボード CPU
		MainBoardCPU_Clock		メインボード CPU 周波数
		MainMemory		メモリー容量
	System	Main		メイン面のシステムソフトウェアバージョン
		Backup		バックアップ面のシステムソフトウェアバージョン

4.1.1.4　エラー

このリソースは共通エラー以外のエラーを返しません。

4.1.1.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/system/version

{
  "info": {
    "result": 0,
    "message": ""
  },
  "systeminfo": {
    "Boot": {
      "System": {
        "Version": "1.0AH",
        "Build": "2025-03-10",
        "Unit": "main"
      },
      "Status": "Reboot",
      "Config": {
        "Unit": "external",
        "Startup": "startup1"
      },
      "ROM": {
        "Version": "1.0V"
      }
    },
    "SystemUpTime": "2025/03/26 15:29:22",
    "HW": {
      "Model": "NS-2260-48",
      "SerialNo": "4A20008S",
      "MAC": {
        "Local_Address": "XX:XX:XX:XX:XX:XX",
        "Number": "2"
      },
      "MainBoardCPU_Model": "A15",
      "MainBoardCPU_Clock": "1500MHz",
      "MainMemory": "1010028"
    },
    "System": {
      "Main": "1.0AH",
      "Backup": "1.0AG"
    }
  }
}
$
```

※実行例は整形して記載しています

4.2 USERS

4.2.1 /users (GET)

4.2.1.1 概要

ユーザー情報の一覧を取得します。

4.2.1.2 リクエスト

項目	内容
アクセス可能なユーザー権限	normal root
オプションパラメーター	この API リソースは指定可能なオプションはありません。

4.2.1.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー名			内容
users (配列)	name		ユーザー名
	group		ユーザーの所属しているグループ名
	encrypt		ハッシュ化されたパスワード
	uid		ユーザーグループ ID の情報
	port		シリアルポートの許可リスト
	permission	root	拡張ユーザーに設定されている管理者権限を表示 on ： 有効 off: 無効
		ttymanage	拡張ユーザーに設定されている tty マネージ機能権限を表示 on ： 有効 off: 無効
	sshkey (配列)	要素 1 [0]	メソッドを表示
		要素 2 [1]	公開鍵を表示

4.2.1.4　エラー

このリソースは共通エラー以外のエラーは返しません。

4.2.1.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/users

{
  "info": {
    "result": 0,
    "message": ""
  },
  "users": [
    {
      "name": "root",
      "group": "root",
      "encrypt": "",
      "uid": 0,
      "port": "",
      "permission": "",
      "sshkey": ""
    },
    {
      "name": "somebody",
      "group": "normal",
      "encrypt": "",
      "uid": 100,
      "port": "",
      "permission": "",
      "sshkey": ""
    },
      "name": "api",
      "group": "extusr",
      "encrypt": "SPS.H.EC3v2a1JRKDVqU.a9k1OIcA0",
      "uid": 401,
      "port": "1-16",
      "permission": {
        "root": "on",
        "ttymanage": "on"
    }
  ]
}
$
```

※本実行例は一部の情報について抜粋しています。

※実行例は整形して記載しています

4.2.2 /users (POST)

4.2.2.1 概要

ユーザーを作成します。

4.2.2.2 リクエスト

項目	内容
アクセス可能なユーザー権限	root
オプションパラメーター	リクエストボディーとして JSON フォーマットのデータを オブジェクト型で指定します。

キー名	バリューの型	内容
name (必須)	文字列	ユーザー名を指定します。 ■文字長 ：16 文字まで ■文字種 ：英数字 _ - （先頭文字は英字） ■default：無し
group (必須)	文字列	ユーザーグループ名を指定します。 ■指定可能な値： normal, extusr, portusr, setup, verup, log ■default：無し
password	文字列	パスワードを平文で指定します。 ■文字長 ：64 文字まで ■文字種 ：英数字 SPACE ! # % * + , - . / : = @ _ ~ ■default：無し ■備考 ：password, encrypt が両方設定されている場合は、password が優先されます。
encrypt	文字列	パスワードをハッシュ値で指定します。 ※SmartCS の show config running で出力される値

＜リクエストボディーのデータフォーマット＞

キー名	バリューの型	内容
port	文字列 or 数値	ポートユーザー、拡張ユーザーに許可するシリアルポートの番号を指定します。 ■設定値 ・1 ポートのみを指定する場合、数値での指定が可能です。 ・複数ポートを指定する場合、ttylist 形式を使い文字列で指定が可能です。 例：1,2,3,4,10,16 ポート “1-4,10,16” ・設定する値が ”” の場合、現在設定されている値を削除します。 ■default ：無し
uid	数値	作成するユーザーのユーザーID を指定します。 ■設定値 100～190：一般ユーザー 401～410：拡張ユーザー 501～599：ポートユーザー 198 ：セットアップユーザー 199 ：バージョンアップユーザー 200 ：ポートログ取得ユーザー ■default ：無し (未指定時は自動で割り当てます)

permission	オブジェクト	拡張ユーザーの場合、付与する権限を指定します。
		■設定値
		管理者権限を設定する場合
		“root”： “on”
		tty マネージ機能権限を設定する場合
		“ttymanage”： “on”
		■default
		{
		“root”： “off”,
		“ttymanage”： “off”
		}

キー名	バリューの型	内容
sshkey	配列	ssh の公開鍵を設定します。 ■設定値 [“<method>”, “<public-key>”] Method は 20 文字まで Public-key は 720 文字まで ■default ：無し

＜CLI コマンドの実行順番＞

実行順	リクエストの延長で実行する CLI コマンド
1	create user <username> group <group> [uid <userid>] [port <enable_port_list>] [{password | encrypt <string>}] ※password, encrypt の両方を指定された場合は password を設定
2	set user <username> permission { root | ttymanage {on | off }}
3	set user <username> sshkey <method> <public-key>

4.2.2.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

4.2.2.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid json data (400)	リクエストボディーの内容が JSON フォーマットでない場合に表示されます。
Error: Invalid request body. (400)	リクエストボディーにオプションパラメーターの指定がない場合や、仕様以外のパラメーターが指定されている場合に表示されます。
Error: The required parameter is missing.($key) (400)	リクエストボディーに必須パラメーターが無い場合に表示されます。
Error: Invalid argument value.($key) (200)	リクエストボディーで指定されたパラメーターが、正しく処理できない場合に表示されます。
Error: $key contains non-usable characters. (200)	

出力された場合は、リクエストボディーの JSON データの内容を確認してください。

4.2.2.5　実行例

リクエストボディーの JSON データ

```
$ cat users-post.json
{
    "name": "testuser",
    "group": "extusr",
    "password": "abcdefghijklmn51",
    "encrypt": "UuSOuT.h8r6nSBV0xaeR1bRhLf9Zx/",
    "uid": 403,
    "port": "1-4",
    "permission": {
        "root": "off",
        "ttymanage": "on"
    },
    "sshkey": [
        "ssh-rsa","AAAAB3NzaC1yc2EAAAADAQABA3FO"
    ]
}
$
```

※リクエストボディーの JSON データ例

```
$ curl -u api:api -X POST
  -H "Content-Type: application/json"
  http://<IP>:<PORT>/api/v1/users --data @./users-post.json

{
  "info": {
    "result": 0,
    "message": ""
  }
}
$
```

※実行例は整形して記載しています

4.2.3 /users/{username} (GET)

4.2.3.1 概要

指定したユーザー情報を取得します。

4.2.3.2 リクエスト

項目	内容
アクセス可能なユーザー権限	normal root
オプションパラメーター	GET クエリーでユーザー名を指定します。

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
{username}	情報を取得する作成済みのユーザー名を指定します。

4.2.3.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー名			内容
users （配列）	name		ユーザー名
	group		ユーザーの所属しているグループ名
	encrypt		ハッシュ化されたパスワード
	uid		ユーザーグループ ID の情報
	port		シリアルポートの許可リスト
	permission	root	拡張ユーザーに設定されている管理者権限を表示 on ： 有効 off: 無効
		ttymanage	拡張ユーザーに設定されている tty マネージ機能権限を表示 on ： 有効 off: 無効
	sshkey （配列）	要素 1 [0]	メソッドを表示
		要素 2 [1]	公開鍵を表示

※4.2.1 項のレスポンスデータと同じ内容となります。
　users 配列に格納されているユーザーは指定したユーザー名のみとなます。

4.2.3.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid request. (400)	GET クエリーで指定したユーザー名に未サポートの文字列が指定される場合に表示されます。 GET クエリーで指定したユーザー名を確認してください。
Error: user {username} does not exist.(103) (200)	指定したユーザーが存在しない場合に表示されます。 GET クエリーで指定したユーザー名を確認してください。

4.2.3.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/users/somebody

{
  "info": {
    "result": 0,
    "message": ""
  },
  "users": [
    {
      "name": "somebody",
      "group": "normal",
      "encrypt": "",
      "uid": 100,
      "port": "",
      "permission": "",
      "sshkey": ""
    }
  ]
}

$
```

※実行例は整形して記載しています

4.2.4 /users/{username}　(PUT)

4.2.4.1　概要

指定したユーザー情報を編集します。

4.2.4.2　リクエスト

項目	内容
アクセス可能なユーザー権限	root
オプションパラメーター	GET クエリーでユーザー名を指定します。 指定したユーザーの情報をリクエストボディーとして JSON フォーマットのデータをオブジェクト型で指定します。 ※設定変更するキー名とバリューのペアのみ （設定変更の対象データのみ） でもユーザー情報の編集を行う事ができます。

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
{username}	情報を編集する作成済みのユーザー名を指定します。

＜リクエストボディーのデータフォーマット＞

キー名	バリューの型	内容
password	文字列	パスワードを平文で指定します。 ■文字長　：64 文字まで ■文字種　：英数字 SPACE ！ # % * + , - . / : = @ _ ~ ■default：無し ■備考　　：password, encrypt が両方設定されている場合は、password が優先されます。
encrypt	文字列	パスワードをハッシュ値で指定します。 ※SmartCS の show config running で出力される値

キー名	バリューの型	内容
port	文字列 or 数値	ポートユーザー、拡張ユーザーに許可するシリアルポートの番号を指定します。 ■設定値 ・1 ポートのみを指定する場合、数値での指定が可能です。 ・複数ポートを指定する場合、ttylist 形式を使い文字列で指定が可能です。 例：1,2,3,4,10,16 ポート “1-4,10,16” ・設定する値が ”” の場合、現在設定されている値を削除します。

permission	オブジェクト	拡張ユーザーの場合、付与する権限を指定します。
		■設定値
		管理者権限を設定する場合
		“root”： “on”
		tty マネージ機能権限を設定する場合
		“ttymanage”： “on”
		■default
		{
		“root”： “off”,
		“ttymanage”： “off”
		}

sshkey	配列	ssh の公開鍵を設定します。 ■設定値 [“<method>”, “<public-key>”] Method は 20 文字まで Public-key は 720 文字まで ・設定する値が ”” の場合、現在設定されている値を削除します。 ■default ：無し

※name, uid, group 情報をリクエストボディーとして指定してもエラーにはなりません。
（ユーザー情報は GET クエリ―の値を使う為、参照もされません）

※すべてのキー/バリューペアを指定する必要はありません。
変更対象のデータのみで設定変更を行うことができます。

＜CLI コマンドの実行順番＞

実行順	リクエストの延長で実行する CLI コマンド
1	set user <username> {password | encrypt <string>}} ※password, encrypt の両方を指定された場合は password を設定
2	set user <username> port <enable port_list > 又は unset user <username> port
3	set user <username> permission {root | ttymanage { on | off }}
4	set user <username> sshkey <method> <public-key> 又は unset user <username> sshkey

4.2.4.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

4.2.4.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid json data (400)	リクエストボディーの内容が JSON フォーマットでない場合に表示されます。
Error: Invalid request. (400)	GET クエリーで指定したユーザー名に未サポートの文字列が指定される場合に表示されます。 GET クエリーで指定したユーザー名を確認してください。
Error: Invalid request body. (400)	リクエストボディーにオプションパラメーターの指定がない場合や、仕様以外のパラメーターが指定されている場合に表示されます。
Error: Invalid argument value.($key) (200)	リクエストボディーで指定されたパラメーターが、正しく処理できない場合に表示されます。
Error: $key contains non-usable characters. (200)	
Error: ※その他エラーメッセージ (200)	指定したユーザーが存在しない場合や、リクエストボディーリクエストボディーで指定されたパラメーターが、正しく処理できない場合に表示されます。

出力された場合は、GET クエリーやリクエストボディーの JSON データの内容を確認してください。

4.2.4.5　実行例

リクエストボディーの JSON データ

```
$ cat users-put.json
{
    "name": "testuser",
    "group": "extusr",
    "password": "abcdefghijklmn51",
    "encrypt": "UuSOuT.h8r6nSBV0xaeR1bRhLf9Zx/",
    "uid": 403,
    "port": "1-4,8,10",
    "permission": {
        "root": "off",
        "ttymanage": "on"
    },
    "sshkey": [
        "ssh-rsa","AAAAB3NzaC1yc2EAAAADAQABA3FO"
    ]
}
$
```

※リクエストボディーの JSON データ例

（設定変更するデータ行のみでもユーザー情報の編集を行う事ができます。）

```
$ curl -u api:api -X PUT -H "Content-Type: application/json"
  http://<IP>:<PORT>/api/v1/users/testuser --data @./users-put.json

{
  "info": {
    "result": 0,
    "message": ""
  }
}
$
```

※実行例は整形して記載しています

4.2.5 /users/{username} (DELETE)

4.2.5.1 概要

指定したユーザーを削除します。

4.2.5.2 リクエスト

項目	内容
アクセス可能なユーザー権限	root
オプションパラメーター	GET クエリー

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
{username}	削除する作成済みのユーザー名を指定します。

4.2.5.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

4.2.5.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid request. (400)	GET クエリーで指定したユーザー名に未サポートの文字列が指定される場合に表示されます。 GET クエリーで指定したユーザー名を確認してください。
Error: user {username} does not exist.(103) (200)	指定したユーザーが存在しない場合に表示されます。 GET クエリーで指定したユーザー名を確認してください。

4.2.5.5　実行例

```
$ curl -u api:api -X DELETE http://<IP>:<PORT>/api/v1/users/testuser1

{
  "info": {
    "result": 0,
    "message": ""
  },
}

$
```

※実行例は整形して記載しています

4.2.6 /users/login (GET)

4.2.6.1 概要

現在ログインしているユーザー情報を取得します。

4.2.6.2 リクエスト

項目	内容
アクセス可能なユーザー権限	normal root
オプションパラメーター	この API リソースは指定可能なオプションはありません。

4.2.6.3 レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー名		内容
user_login (配列)	User-Name	ログインしているユーザー名
	Device	接続に使用しているデバイス名またはデバイス番号
	Login-Time	ログインした時間
	Idle	最後に操作を行ってからの経過時間
	Remote-Host	接続しているホストの IP アドレス または 名前

4.2.6.4　エラー

このリソースは共通エラー以外のエラーは返しません。

4.2.6.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/users/login

{
  "info": {
    "result": 0,
    "message": ""
  },
  "user_login": [
    {
      "User-Name": "somebody",
      "Device": "console",
      "Login-Time": "May 27 00:05:18",
      "Idle": "00:00",
      "Remote-Host": ""
    },
    {
      "User-Name": "api",
      "Device": "0",
      "Login-Time": "May 26 22:06:18",
      "Idle": "00:20",
      "Remote-Host": "172.31.8.41"
    }
  ]
}

$
```

※実行例は整形して記載しています

4.3 SERIAL

4.3.1 /serial/tty (GET)

4.3.1.1 概要

各 TTY のシリアル情報の一覧を取得します。

4.3.1.2 リクエスト

項目	内容
アクセス可能なユーザー権限	normal root
オプションパラメーター	この API リソースは指定可能なオプションはありません。

4.3.1.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー			内容
ttylist (配列)	tty		シリアルポートの番号
	config	baud	シリアルポートの転送速度
		bitchar	シリアルポートのデータビット長
		parity	シリアルポートのパリティー
		stop	シリアルポートのストップビット長
		flow	シリアルポートのフロー制御
		detect_dsr	DSR 信号遷移検出機能の動作設定
		label	監視対象機器のラベル名
	status	DSR	信号線 DSR の現在の状態
		CTS	信号線 CTS の現在の状態
		DTR	信号線 DTR の現在の状態
		RTS	信号線 RTS の現在の状態
		CD	信号線 CD の現在の状態
	stats	TX_Octets	送信オクテット数
		RX_Octets	受信オクテット数
		Error_Parity	受信パリティーエラーの回数
		Error_Framing	受信フレーミングエラーの回数
		Error_Overrun	受信オーバーランの回数
		Break_Count	受信ブレークの回数

4.3.1.4　エラー

このリソースは共通エラー以外のエラーを返しません。

4.3.1.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/serial/tty

{
  "info": {
    "result": 0,
    "message": ""
  },
  "ttylist": [
    {
      "tty": 1,
      "config": {
        "baud": 115200,
        "bitchar": 8,
        "parity": "none",
        "stop": 1,
        "flow": "none",
        "detect_dsr": "off",
        "label": "NS-2260-48"
      },
      "status": {
        "DSR": "on",
        "CTS": "on",
        "DTR": "on",
        "RTS": "on",
        "CD": "on"
      },
      "stats": {
        "TX_Octets": 0,
        "RX_Octets": 0,
        "Error_Parity": 0,
        "Error_Framing": 0,
        "Error_Overrun": 0,
        "Break_Count": 0
      }
    },
    {
      "tty": 2,
      "config": {
        "baud": 9600,
    :
    省略
    :
    }
  ]
}

$
```

※実行例は整形して記載しています

4.3.2 /serial/tty/{ttylist} (GET)

4.3.2.1 概要

指定した TTY のシリアル情報を取得します。

4.3.2.2 リクエスト

項目	内容
アクセス可能なユーザー権限	normal root
オプションパラメーター	GET クエリー

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
{ttylist}	TTY 番号を ttylist 形式で指定します。 例：tty 番号 1,2,3,4,10 を指定する場合 1-4,10

4.3.2.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー			内容
ttylist (配列)	tty		シリアルポートの番号
	config	baud	シリアルポートの転送速度
		bitchar	シリアルポートのデータビット長
		parity	シリアルポートのパリティー
		stop	シリアルポートのストップビット長
		flow	シリアルポートのフロー制御
		detect_dsr	DSR 信号遷移検出機能の動作設定
		label	監視対象機器のラベル名
	status	DSR	信号線 DSR の現在の状態
		CTS	信号線 CTS の現在の状態
		DTR	信号線 DTR の現在の状態
		RTS	信号線 RTS の現在の状態
		CD	信号線 CD の現在の状態
	stats	TX_Octets	送信オクテット数
		RX_Octets	受信オクテット数
		Error_Parity	受信パリティーエラーの回数
		Error_Framing	受信フレーミングエラーの回数
		Error_Overrun	受信オーバーランの回数
		Break_Count	受信ブレークの回数

4.3.2.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid request. (400)	GET クエリーで指定した TTY リスト(ttylist)に未サポートの文字列や仕様外の値が指定された場合に表示されます。 GET クエリーで指定した ttylist を確認してください。
Error: ″show json tty 1,16,17 <-- syntax error [tty number (n[-n][,n[-n]]... n=1-16 listmax=16)]″ ※その他CLIのエラーメッセージ (200)	指定した tty 番号が存在しない場合や、GET クエリーで指定されたパラメーターが、正しく処理できない場合に表示されます。

4.3.2.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/serial/tty/1,16

{
  "info": {
    "result": 0,
    "message": ""
  },
  "ttylist": [
    {
      "tty": 1,
      "config": {
        "baud": 9600,
        "bitchar": 8,
        "parity": "none",
        "stop": 1,
        "flow": "none",
        "detect_dsr": "off",
        "label": "NS-2260-48"
      },
      "status": {
        "DSR": "on",
        "CTS": "on",
        "DTR": "on",
        "RTS": "on",
        "CD": "on"
      },
      "stats": {
        "TX_Octets": 0,
        "RX_Octets": 0,
        "Error_Parity": 0,
        "Error_Framing": 0,
        "Error_Overrun": 0,
        "Break_Count": 0
      }
    },
    {
      "tty": 16,
      "config": {
        "baud": 115200,
    :
    省略
    :
    }
  ]
}

$
```

※実行例は整形して記載しています

4.3.3 /serial/tty/{ttylist} (PUT)

4.3.3.1 概要

指定した TTY のシリアル情報を編集します。

4.3.3.2 リクエスト

項目	内容
アクセス可能なユーザー権限	root
オプションパラメーター	GET クエリ― 指定した各 TTY の設定をリクエストボディーとして、JSON フォーマットのデータをオブジェクト型で指定します。 ※設定変更するキー名とバリューのペアのみ （設定変更の対象データのみ） でシリアル情報の編集を行う事ができます。

＜GET クエリ―によるパラメーターの指定＞

パラメーター	内容
{ttylist}	TTY 番号を ttylist 形式で指定します。 例：tty 番号 1,2,3,4,10 を指定する場合 1-4,10

＜リクエストボディーのデータフォーマット＞

キー名	バリューの型	内容
baud	数値	シリアルポートの転送速度を指定します。 ■指定可能な値： 2400, 4800, 9600, 19200, 38400, 57600, 115200 ■default：無し

キー名	バリューの型	内容
bitchar	数値	シリアルポートのデータビット長を指定します。 ■指定可能な値 7,8 ■default：無し
parity	文字列	シリアルポートのパリティーを指定します。 ■指定可能な値： even, odd, none ■default：無し
stop	数値	シリアルポートのストップビット長を指定します。 ■指定可能な値 1, 2 ■default：無し
flow	文字列	シリアルポートのフロー制御を指定します。 ■指定可能な値 xon, rs, none ■default：無し
detect_dsr	文字列	DSR 信号遷移検出機能の動作設定を指定します。 ■指定可能な値 on_edge, on_polling, off ■default：無し
label	文字列	監視対象機器のラベル名を指定します。 ■文字長 ：1-32 文字 ■文字種 ：英数字 SPACE @ _ - . ■default：無し ■備考 ・設定する値が ”” の場合、現在設定されている値を削除します。

＜CLI コマンドの実行順番＞

実行順	リクエストの延長で実行する CLI コマンド
1	set tty <ttylist> baud
2	set tty <ttylist> bitchar
3	set tty <ttylist> parity
4	set tty <ttylist> stop
5	set tty <ttylist> flow
6	set tty <ttylist> detect_dsr
7	set portd tty <ttylist> label <"string"> 又は unset portd tty < ttylist> label

4.3.3.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

4.3.3.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid json data (400)	リクエストボディーの内容が JSON フォーマットでない場合に表示されます。
Error: Invalid request. (400)	GET クエリーで指定したユーザー名に未サポートの文字列が指定される場合に表示されます。 GET クエリーで指定したユーザー名を確認してください。
Error: Invalid argument value.($key) (200)	リクエストボディーで指定されたパラメーターが、正しく処理できない場合に表示されます。
Error: $key contains non-usable characters. (200)	
Error: ※その他エラーメッセージ (200)	GET クエリーで指定した ttylist の値が範囲外(NS-2260-16 で、17 を指定）など、指定されたパラメーターが、正しく処理できない場合に表示されます。

出力された場合は、GET クエリー、リクエストボディーの JSON データ内容を確認してください。

4.3.3.5　実行例

リクエストボディーの JSON データ

```
$ cat tty-put.json
{
    "baud": 9600,
    "bitchar": 8,
    "parity": "none",
    "stop": 1,
    "flow": "none",
    "detect_dsr": "off",
    "label": "SWITCH-1"
}

$
```

※リクエストボディーの JSON データ例

（設定変更するデータ行のみでもシリアル情報の編集を行う事ができます。）

```
$ curl -u api:api -X PUT -H "Content-Type: application/json"
  http://<IP>:<PORT>/api/v1/serial/tty/16 --data @./tty-put.json

{
  "info": {
    "result": 0,
    "message": ""
  }
}

$
```

※実行例は整形して記載しています

4.3.4 /serial/hangup/tty/{ttylist} (POST)

4.3.4.1 概要

指定した TTY をハングアップします。

4.3.4.2 リクエスト

項目	内容
アクセス可能なユーザー権限	root
オプションパラメーター	GET クエリー
	POST ※リクエストボディーデータはなし

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
{ttylist}	TTY 番号を ttylist 形式で指定します。 例：tty 番号 1,2,3,4,10 を指定する場合 1-4,10

4.3.4.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

4.3.4.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid request. (400)	GET クエリーで指定したユーザー名に未サポートの文字列が指定される場合に表示されます。 GETクエリーで指定したユーザー名を確認してください。
Error: ※その他エラーメッセージ (200)	GETクエリーで指定したttylistの値が範囲外(NS-2260-16で、17を指定）など、指定されたパラメーターが、正しく処理できない場合に表示されます。

4.3.4.5　実行例

リクエストボディーの JSON データ

```
$ curl -u api:api -X POST -H "Content-Type: application/json" –d ""
  http://<IP>:<PORT>/api/v1/serial/hangup/tty/16

{
  "info": {
    "result": 0,
    "message": ""
  }
}

$
```

※実行例は整形して記載しています

4.4 TTYMANAGE

4.4.1 /ttymanage (POST)

4.4.1.1 概要

TTY マネージ機能を使ってシリアルポートに文字列の送受信を実行します。

4.4.1.2 リクエスト

項目	内容
アクセス可能なユーザー権限	ttymanage
オプションパラメーター	リクエストボディーとして JSON フォーマットのデータを オブジェクト型で指定します。

＜リクエストボディーのデータフォーマット＞

（１）基本機能

キー名	型	内容
tty (必須)	数値 (文字列)	tty 番号を指定します。 ■範囲　：1-48 ■備考 ・ttylist 形式は未サポート ・文字列指定“1”でも動作します。 ■default：無し
cmd_timeout	数値 (文字列)	sendchar 送信時のタイムアウト値(秒)を指定します。 ■範囲　：1-30 ■備考 ・文字列指定でも動作します。 ■default：10

キー名	型	内容
nl	文字列	sendchar の末尾に付与する改行コードを指定します。 ■選択肢 ：cr, lf, crlf ■備考 ：選択肢以外はエラーとなります。
recvchar	配列	sendchar 送信後に待ち受ける文字列を指定します。 ■登録数 ：最大 16 ■default：無し
recvchar_regex	配列	sendchar 送信後に待ち受ける文字列（正規表現）を指定します。 ■登録数 ：最大 8 ■default：無し
sendchar	配列	送信する文字列を指定します。 ※文字列及び各オプションについては全てダブルクオーテーションで囲って指定します。 ■登録数 ：最大 1024 ■文字長 ：1-128 文字 ■文字種 ：英数字 SPACE ！％＊＋，－．／：＝＠＿＾～ ■特殊な送信方法 ・__NL__ 改行送信 ・__CTL__:hex 制御文字(1 文字)送信 ・__HEX__:hexs 制御文字(複数)送信 ■送信タイミングオプション ・__WAIT__:sec ・__NOWAIT__ ・__NOWAIT__:sec ※時間指定オプションの範囲は 1-1800(秒)

＜リクエストボディーのデータフォーマット＞

（２）エラー時の動作

キー名	型	内容
error_detect_on_sendchar	文字列	sendchar 送信後にエラーが発生した場合、以降の sendchar を送信するか/しないかの設定を指定します。 ■選択肢 ・exec エラーが発生した場合でも、sendchar を送信します ・cancel エラーが発生した場合、sendchar を送信しません ■備考 ・エラー例 - TimeOut - Session limit over - Error Method - Connection closed
error_recvchar_regex	配列	sendchar 送信後、指定された文字列(正規表現)が含まれていたらエラーとなる文字列を指定します。 ■登録数 ：最大 8 ■default：無し

＜リクエストボディーのデータフォーマット＞

（３）デバッグ

キー名	型	内容
ttycmd_debug	文字列	デバッグ情報をレスポンスデータに含むかどうかの設定値を指定します。 ■選択肢 ：off, on, detail ■default：off ■備考 ：選択肢以外はエラーとなります。

4.4.1.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー	内容
request	リクエストボディーで指定したリクエストデータ
error	発生したエラー数
data (配列)	リクエストボディーで指定したシリアルの送受信文字列について 送信データ（execute_command） ・文字列 受信データ（response） ・配列 の組み合わせをオブジェクト形式で配列に格納したデータ
debug	デバッグデータ ※ttycmd_debug が on または detail の時のみ

4.4.1.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容
Error: Invalid json data (400)	リクエストボディーの内容が JSON フォーマットでない場合に表示されます。
Error: The required parameter is missing.($key) (400)	リクエストボディーに必須パラメーターが無い場合に表示されます。
Error: The XXX option must be an array. (400)	リクエストボディーに配列型で指定するパラメーターをサポート外の型で指定した場合に表示されます。（sendchar, recvchar, recvchar_regex, error_recvchar_regex）
Error: The cmd_timeout option must be an integer(range:1-30). (400)	リクエストボディーの cmd_timeout オプションの設定範囲が誤っている場合に表示されます。
Error: The nl option must be a string (crlf, cr, lf (default: cr)). (400)	リクエストボディーの nl オプションの設定値が誤っている場合に表示されます。
Error: XXX :sec option must be an integer(range:1-1800). (400)	リクエストボディーの sendchar オプションの送信タイミングの時間設定値（__WAIT__:sec、__NOWAIT__:sec）が誤っている場合に表示されます。
Error: The configurable lines of sendchar are 1-1024. (400)	リクエストボディーの sendchar オプションの設定数が範囲外の場合に表示されます。
Error: The configurable lines of recvchar are 1-16. (400)	リクエストボディーの recvchar オプションの設定数が範囲外の場合に表示されます。
Error: The configurable lines of recvchar_regex are 1-8. (400)	リクエストボディーの recvchar_regex オプションの設定数が範囲外の場合に表示されます。

メッセージ (ステータスコード)	内容
Error: The configurable lines of error_recvchar_regex are 1-8. (400)	リクエストボディーの error_recvchar_regex オプションの設定数が範囲外の場合に表示されます。
Error: The error_detect_on_sendchar option must be a string (cancel, exec (default: cancel)). (400)	リクエストボディーの error_detect_on_sendchar オプションの設定値がサポート外の場合に表示されます。
Error: The ttycmd_debug option must be a string (off, on, detail (default: off)). (400)	リクエストボディーの ttycmd_debug オプションの設定値がサポート外の場合に表示されます。
Error: sendchar contains non-usable characters. (400)	リクエストボディーの sendchar オプションの設定値にサポート外の文字種が含まれていた場合に表示されます。
Error: ※その他エラーメッセージ (200)	リクエストボディーの tty オプションの設定値が範囲外（NS-2260-16 で、17 を指定）など、指定されたパラメーターが、正しく処理できない場合に表示されます。

表示された場合、リクエストボディーの JSON データの内容を確認してください。

4.4.1.5　実行例

リクエストボディーの JSON データ

```
$ cat switch_version.json
{
    "tty": 2,
    "nl": "cr",
    "cmd_timeout": 30,
    "recvchar": [
        "Switch>",
        "Switch#",
        "Press RETURN to get started."
    ],
    "recvchar_regex": [
        "[Uu]sername:",
        "[Pp]assword:",
        "(^|\\r|\\n|!)[a-zA-Z0-9_().-]*(>|#) "
    ],
    "sendchar": [
        "__NL__",
        "ssol",
        "ssol",
        "terminal length 0",
        "show version",
        "exit"
    ]
}

$
```

※TTY 番号 2 に接続されている Switch 製品にログイン後、

”terminal length 0”、

“show version”

コマンドを実行してログアウトするシナリオ例となります。

※実行例は整形して記載しています

```
$ curl -u api:api -X POST -H "Content-Type: application/json"
  http://<IP>:<PORT>/api/v1/ttymanage --data @./switch_version.json

{
  "info": {
    "result": 0,
    "message": ""
  },
  "request": {
    "tty": 2,
    "nl": "cr",
    "cmd_timeout": 30,
    "sendchar": [
      "__NL__",
      "ssol",
      "ssol",
      "terminal length 0",
      "show version",
      "exit"
    ],
    "recvchar": [
      "Switch>",
      "Switch#",
      "Press RETURN to get started."
    ],
    "recvchar_regex": [
      "[Uu]sername:",
      "[Pp]assword:",
      "(^|\\r|\\n|!)[a-zA-Z0-9_().-]*(>|#) "
    ],
    "error_recvchar_regex": [],
    "error_detect_on_sendchar": "cancel",
    "ttycmd_debug": "off"
  },
  "data": [
    {
      "execute_command": "__NL__",
      "response": [
        "Username:"
      ]
  〜〜
  省略
  〜〜
        "Press RETURN to get started.",
        ""
      ]
    }
  ],
  "error": 0
}
$
```

※レスポンスデータの一部(data)については内容を省略しています。

data 部分に、実際のシリアル通信のオペレーションが格納されます。

4.5 LOG/HISTORY

4.5.1 /log/history/command (GET)

4.5.1.1 概要

SmartCS のコマンドログ情報を取得します。

4.5.1.2 リクエスト

項目	内容
アクセス可能なユーザー権限	root
オプションパラメーター	GET クエリー

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
lines	表示するコマンドログの行数を指定します。 ■指定範囲 ・1-1000：指定した行数表示します。 例：10 と指定した場合、最新の 10 行を表示します。 ・all ：最新のログを最大 8192 行表示します。 ・未指定：パラメーターが未指定の場合、最新の 50 行を表示します。

4.5.1.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー	内容
log (配列)	SmartCS のコマンドログデータを表示

4.5.1.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid value(lines). (200)	GET クエリーで指定した表示行数に未サポートの文字列や値が指定される場合に表示されます。 GET クエリーで指定した内容を確認してください。

4.5.1.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/log/history/command

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
    "2025 May 26 22:06:21 root: show version",
      （省略）
    "2025 May 26 22:06:21 root: show user"
  ]
}

$
```

※オプションなし

※実行例は整形して記載しています

```
$ curl -u api:api -X GET
  http://<IP>:<PORT>/api/v1/log/history/command?lines=100

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
      （省略）
  ]
}

$
```

※最新の 100 行を表示

```
$ curl -u api:api -X GET
  http://<IP>:<PORT>/api/v1/log/history/command?lines=all

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
      （省略）
  ]
}

$
```

※全てのログ（最大 8192 行）を表示

4.5.2 /log/history/console (GET)

4.5.2.1 概要

SmartCS のコンソールログ情報を取得します。

4.5.2.2 リクエスト

項目	内容
アクセス可能なユーザー権限	root
オプションパラメーター	GET クエリー

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
lines	表示するコマンドログの行数を指定します。 ■指定範囲 ・1-1000：指定した行数表示します。 例：10 と指定した場合、最新の 10 行を表示します。 ・all ：最新のログを最大 8192 行表示します。 ・未指定：パラメーターが未指定の場合、最新の 50 行を表示します。

4.5.2.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー	内容
log (配列)	SmartCS のコンソールログデータを表示

4.5.2.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid value(lines). (200)	GET クエリーで指定した表示行数に未サポートの文字列や値が指定される場合に表示されます。 GET クエリーで指定した内容を確認してください。

4.5.2.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/log/history/console

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
    "2025 May 27 00:05:18 login: login success: somebody/console",
      （省略）
    "2025 May 27 00:15:18 login: logout: somebody/console"
  ]
}

$
```

※オプションなし

※実行例は整形して記載しています

```
$ curl -u api:api -X GET
  http://<IP>:<PORT>/api/v1/log/history/console?lines=100

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
      （省略）
  ]
}

$
```

※最新の 100 件を表示

```
$ curl -u api:api -X GET
  http://<IP>:<PORT>/api/v1/log/history/console?lines=all

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
      （省略）
  ]
}

$
```

※全てのログ（最大 8192 行）を表示

4.5.3 /log/history/ttysend/tty/{ttyno} (GET)

4.5.3.1 概要

TTY マネージ機能の各コマンドが TTY に送出したデータログを取得します。

4.5.3.2 リクエスト

項目	内容
アクセス可能なユーザー権限	root
オプションパラメーター	GET クエリー

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
{ttyno}	TTY 番号を一つ指定します。 ■指定範囲：1-48
lines	表示するコマンドログの行数を指定します。 ■指定範囲 ・1-1000：指定した行数表示します。 例：10 と指定した場合、最新の 10 行を表示します。 ・all ：最新のログを最大 8192 行表示します。 ・未指定：パラメーターが未指定の場合、最新の 50 行を表示します。

4.5.3.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー	内容
log (配列)	TTY マネージ機能の各コマンドが TTY に送出したデータログを表示

4.5.3.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid value(ttyno). (400)	GET クエリーで指定した TTY 番号に未サポートの文字列や値が指定される場合に表示されます。 GET クエリーで指定した内容を確認してください。
Error: Invalid value(lines). (200)	GET クエリーで指定した表示行数に未サポートの文字列や値が指定される場合に表示されます。 GET クエリーで指定した内容を確認してください。
Error: ※その他エラーメッセージ (200)	GET クエリーで指定した TTY 番号の値が範囲外（NS-2260-16 で、17 を指定）など、指定されたパラメーターが、正しく処理できない場合に表示されます。

4.5.3.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/log/history/ttysend/tty/4

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
    "2025 May 12 19:01:01 restapi: enable<CR>",
      （省略）
    "2025 May 12 19:02:19 restapi: show running-config<CR>
  ]
}

$
```

※オプションなし

※実行例は整形して記載しています

```
$ curl -u api:api -X GET
  http://<IP>:<PORT>/api/v1/log/history/ttysend/tty/4?lines=25

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
      （省略）
  ]
}

$
```

※最新の 25 行を表示

```
$ curl -u api:api -X GET
  http://<IP>:<PORT>/api/v1/log/history/ttysend/tty/4?lines=all

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
      （省略）
  ]
}

$
```

※全てのログ（最大 8192 行）を表示

4.5.4 /log/history/webapi (GET)

4.5.4.1 概要

SmartCS の webapi ログ情報を取得します。

4.5.4.2 リクエスト

項目	内容
アクセス可能なユーザー権限	root
オプションパラメーター	GET クエリー

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
lines	表示する REST API のログの行数を指定します。 ■指定範囲 ・1-1000：指定した行数表示します。 例：10 と指定した場合、最新の 10 行を表示します。 ・all ：最新のログを最大 8192 行表示します。 ・未指定：パラメーターが未指定の場合、最新の 50 行を表示します。

4.5.4.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー	内容
log (配列)	SmartCS の webapi のログデータを表示

4.5.4.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid value(lines). (200)	GET クエリーで指定した表示件数に未サポートの文字列や値が指定される場合に表示されます。 GET クエリーで指定した内容を確認してください。

4.5.4.5　実行例

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/log/history/webapi

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
     （省略）
    "2025 May 27 11:47:15 [10080] login success: api/172.31.8.41:41188",
    "2025 May 27 11:47:16 [10080] logout: api/172.31.8.41:41188"
  ]
}

$
```

※オプションなし

※実行例は整形して記載しています

```
$ curl -u api:api -X GET
  http://<IP>:<PORT>/api/v1/log/history/webapi?lines=100

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
     （省略）
  ]
}

$
```

※最新の 100 行を表示

```
$ curl -u api:api -X GET
  http://<IP>:<PORT>/api/v1/log/history/webapi?lines=all

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
     （省略）
  ]
}

$
```

※全てのログ（最大 8192 行）を表示

4.6 LOG/SERIAL

4.6.1 /log/serial/tty/{ttyno} (GET)

4.6.1.1 概要

SmartCS の TTY ログ情報を取得します。

4.6.1.2 リクエスト

項目	内容
アクセス可能なユーザー権限	ttymanage
オプションパラメーター	GET クエリー

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
{ttyno}	TTY 番号を一つ指定します。 ■指定範囲：1-48
lines	表示する TTY ログの行数を指定します。 ■指定範囲 ・1-1000：指定した行数表示します。 例：10 と指定した場合、最新の 10 行を表示します。 ・all ：最新のログを最大 8192 行表示します。 ・未指定：パラメーターが未指定の場合、最新の 50 行を表示します。

4.6.1.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー	内容
log (配列)	SmartCS が保存している指定した TTY 番号のログ情報を表示

4.6.1.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid value(ttyno). (400)	GET クエリーで指定した TTY 番号に未サポートの文字列や値が指定される場合に表示されます。 GET クエリーで指定した内容を確認してください。
Error: Invalid value(lines). (200)	GET クエリーで指定した表示件数に未サポートの文字列や値が指定される場合に表示されます。 GET クエリーで指定した内容を確認してください。
Error: ※その他エラーメッセージ (200)	GET クエリーで指定した TTY 番号の値が範囲外（NS-2260-16 で、17 を指定）など、指定されたパラメーターが、正しく処理できない場合に表示されます。

4.6.1.5　実行例

```
$  curl -u api:api -X GET http://<IP>:<PORT>/api/v1/log/serial/tty/2

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
     （省略）
  ]
}

$
```

※オプションなし

※実行例は整形して記載しています

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/log/serial/tty/2?lines=100

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
     （省略）
  ]
}

$
```

※最新の 100 行を表示

```
$ curl -u api:api -X GET http://<IP>:<PORT>/api/v1/log/serial/tty/2?lines=all

{
  "info": {
    "result": 0,
    "message": ""
  },
  "log": [
     （省略）
  ]
}

$
```

※最大 8192 行分のログファイルを表示

4.6.2 /log/serial/files/tty/{ttyno} (GET)

4.6.2.1 概要

SmartCS の TTY ログデータを取得(ダウンロード)します。

4.6.2.2 リクエスト

項目	内容
アクセス可能なユーザー権限	ttymanage
オプションパラメーター	GET クエリー

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
{ttyno}	TTY 番号を一つ指定します。 ■指定範囲：1-48
lines	表示する TTY ログの行数を指定します。 ■指定範囲 ・1-1000：指定した行数表示します。 例：10 と指定した場合、最新の 10 行を表示します。 ・all ：本装置に記録されているログを全て表示します。 ・未指定：パラメーターが未指定の場合、最新の 50 行を表示します。

4.6.2.3　レスポンス

項目	内容
フォーマット	テキストデータ（text/plain）

＜レスポンスデータ＞

内容
SmartCS が保存している指定した TTY 番号のログ情報をテキストデータでダウンロードします。 ■ファイル名仕様（Content-Disposition の attachment:filename） ・ラベル名_ホスト名_ttyNN_yymmddhhmm.log － NN は TTY 番号（TTY 番号が一桁の場合、0 が付与されます。TTY2 の場合、_tty02_） － ラベル名に SPACE を使っている場合、SPACE は_(アンダーバー)に変換されます。 － ラベル名が設定されていない場合は「ホスト名_ttyNN_yymmddhhmm.log」となります。

4.6.2.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid value(ttyno). (400)	GET クエリーで指定した TTY 番号に未サポートの文字列や値が指定される場合に表示されます。 GET クエリーで指定した内容を確認してください。
Error: Invalid value(lines). (200)	GET クエリーで指定した表示件数に未サポートの文字列や値が指定される場合に表示されます。 GET クエリーで指定した内容を確認してください。
Error: ※その他エラーメッセージ (200)	GET クエリーで指定した TTY 番号の値が範囲外（NS-2260-16 で、17 を指定）など、指定されたパラメーターが、正しく処理できない場合に表示されます。

4.6.2.5　実行例

```
$ curl -u api:api –LOJ -X GET http://<IP>:<PORT>/api/v1/log/serial/files/tty/2
$ ls
SWITCH-1_NS-2260_tty02_2205271913.log

$
```

※オプションなし

※実行例は整形して記載しています

```
$ curl -u api:api –LOJ -X GET
  http://<IP>:<PORT>/api/v1/log/serial/files/tty/2?lines=100
$ ls
SWITCH-1_NS-2260_tty02_2205271913.log

$
```

※最新の 100 行分のログファイルをダウンロード

4.6.3 /log/serial/search/tty/{ttyno} (GET)

4.6.3.1 概要

SmartCS の TTY ログ情報を検索します。

4.6.3.2 リクエスト

項目	内容
アクセス可能なユーザー権限	ttymanage
オプションパラメーター	GET クエリー

＜GET クエリーによるパラメーターの指定＞

パラメーター	内容
{ttyno} (必須)	TTY 番号を一つ指定します。 ■指定範囲：1-48
string (必須)	検索文字列を指定します。 ■文字長 ：1-32 ■文字種 ：英数字 SPACE ! # % + , - . / : = @ _ ■備考 ：スペースを含んだ文字列を検索する場合、 スペースは %20 と指定します。
lines	検索文字列に合致した場合の出力行数を指定します。 ■指定範囲：0-64 ・0 を指定した場合、検索文字列を含む行のみが検索結果として出力されます。 ・1 を指定した場合、検索文字列＋前後 1 行の計 3 行が検索結果として出力されます。 64 を指定した場合は、129 行となります。 ■備考 ：lines 指定がない場合、検索文字列は出力されません。

4.6.3.3　レスポンス

項目	内容
フォーマット	JSON（オブジェクト型）
共通データ	「3.3.2 共通データ」参照

＜レスポンスデータ＞

キー	内容
tty	ログ情報を検索する TTY 番号
label	ログ情報を検索する TTY 番号に設定されているラベル名
string	検索文字列
count	ログ情報に含まれていた検索文字列数
data (配列)	ログ情報に含まれていた検索文字列を含むログデータを格納します。 検索結果の表示は 512 件までとなります。512 件を超える場合はエラーとなります。

4.6.3.4　エラー

このリソースは共通エラー以外に以下のエラーが発生します。

メッセージ (ステータスコード)	内容と対処
Error: Invalid value(ttyno). (400)	GET クエリーで指定した TTY 番号に未サポートの文字列や値が指定される場合に表示されます。
Error: The required parameter is missing.(string) (400)	GET クエリーの必須パラメーターである検索文字列が指定されていません。
Error: The configuable length of search string is 1-32. (400)	GET クエリーの検索文字列の文字長が範囲外となっています。
Error: string contains non-usable characters. (400)	GET クエリーの検索文字列に未サポートの文字種が含まれています。
Error: Invalid value(lines: 0-64). (400)	GET クエリーで指定した検索文字列数が範囲外の値となります。
Error: ※その他エラーメッセージ (200)	GET クエリーで指定した TTY 番号の値が範囲外（NS-2260-16 で、17 を指定）など、指定されたパラメーターが、正しく処理できない場合に表示されます。

表示された場合、GET クエリーで指定した内容を確認してください。

4.6.3.5　実行例

```
$ curl -u api:api -X GET
  "http://<IP>:<PORT>/api/v1/log/serial/search/tty/2?string=version"

{
  "info": {
    "result": 0,
    "message": ""
  },
  "tty": "2",
  "label": "SWITCH",
  "string": "version",
  "count": "4",
  "data": []
}

$
```

※“verison”を含む文字列を検索した場合

※実行例は整形して記載しています

```
$ curl -u api:api -X GET
  "http://<IP>:<PORT>/api/v1/log/serial/search/tty/2?string=version&lines=1"

{
  "info": {
    "result": 0,
    "message": ""
  },
  "tty": "2",
  "label": "SWITCH",
  "string": "version",
  "count": "4",
  "data": [
    [
      "SWITCH#",
      "SWITCH#show version",
      "XXX Software  Build-Date (4/16) "
    ],
    [
      "!",
      "version 1.0",
      "xxxxxxxxxxxxxxxxx"
    ],
    [
      （省略）
    ]
  ]
}

$
```

※“verison”を含む文字列の前後 1 行を検索した場合

5　章　/ttymanage　の解説

5.1　使用上の注意

/ttymanage の API リソースを指定して TTY マネージ機能を使う上での注意を記載します。TTY マネージ機能を使う事で、SmartCS のシリアルポートに接続されている機器のコンソールに対して指定された文字列の送受信を行い、様々なオペレーションを実行する事が可能となります。

ただし、以下の点には注意してご利用下さい。

(1) コンソールの初期状態

SmartCS に接続されている機器のコンソール状態について、TTY マネージ機能は管理、制御を行いません。最後に実行したコマンドによって、SmartCS に接続されている機器のコンソールは

・ログインプロンプト状態
・一般ユーザーシェル状態
・管理者ユーザーシェル状態
・設定投入用シェル状態

と様々な状態になっている可能性があります。SmartCS に接続されている機器のコンソール状態を考慮してリクエストボディーの JSON データを作成してください。

例：必ず最後にログインプロンプト状態に戻す等

(2) コンソール接続の排他制御

SmartCS のシリアルポートに接続されている機器に対してオペレーションする方法は2種類あります。

① telnet/ssh コマンドでポートサーバーにポートユーザーでアクセス後
直接通信を行う方法

②TTY マネージ機能を使い、/ttymanage の API リソースに対して
リクエストボディーとして JSON で定義するシリアル送受信シナリオを送信して、
通信を行う方法

この２種類のオペレーション方法がある為万が一両方のオペレーションを同時に実行してしまい、意図しないコマンドを送信してしまう、など事故を防ぐため、排他制御が可能となっています。

シリアルポートのオペレーションの排他機能を有効にする場合

```
(0)NS-2260# set portd service exclusive on
(0)NS-2260#
```

※デフォルトは排他制御が有効になっています

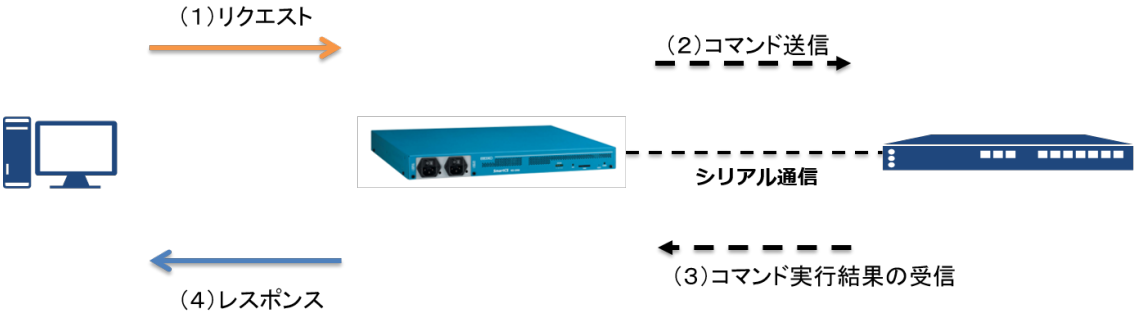
シリアルポートのオペレーションの排他機能を無効にする場合

```
(0)NS-2260# set portd service exclusive off
(0)NS-2260#
```

※シリアル送受信のシナリオを作成する際の検証時などにご利用ください

5.2　制限事項

REST API 機能全般に当てはまる内容となりますが、特に/ttymanage の API リソースを利用した TTY マネージ機能はクライアントからリクエストを受けて、シリアル通信を行った後にレスポンスを返すという機能の都合上、クライアントアプリケーションのタイムアウト時間にケアが必要となります。



※（２）（３）は１リクエストの処理内で複数行うケースが多いと思われます。

シリアル通信のオペレーションシナリオによって、REST API のクライアント側で設定するタイムアウト値は長めに設定してください。また、クライアントアプリケーションのタイムアウト時間が変更できない場合や、タイムアウト時間の設定値が短い場合（シリアル通信のオペレーションシナリオを終えられない時）は、リクエストを複数に分けて行うなどしてください。

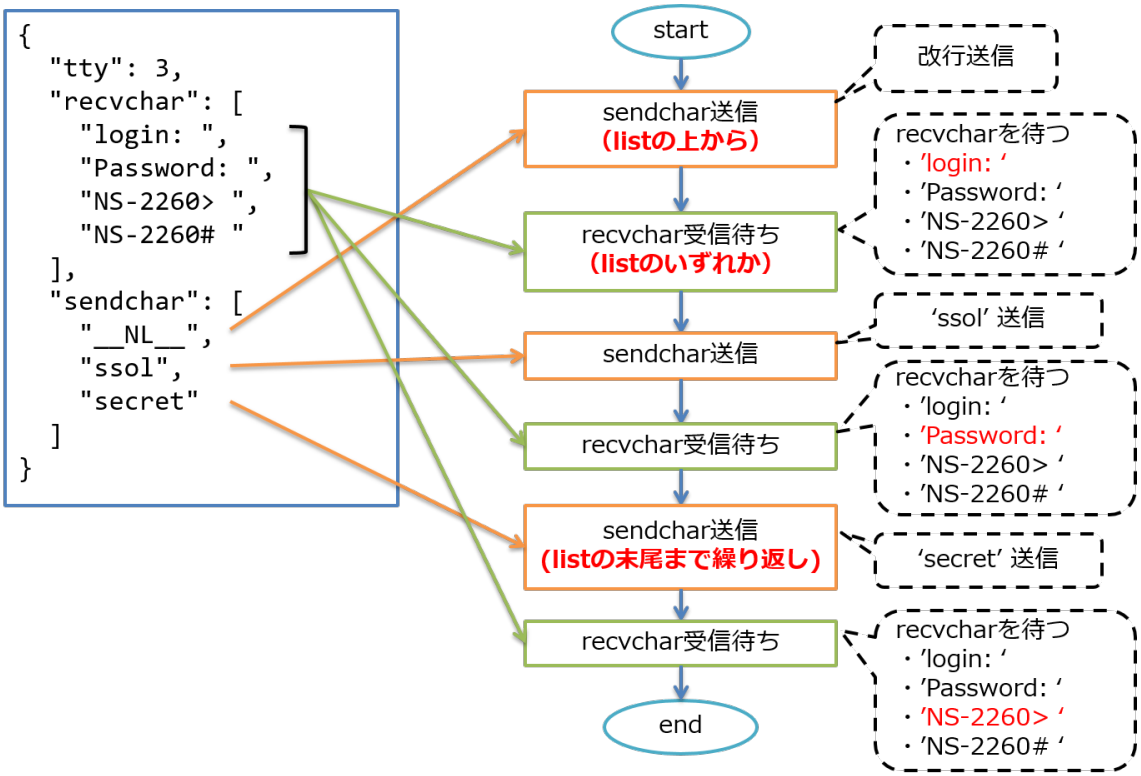
5.3　各オプションの動作

各オプションの動作についてそれぞれ解説します。

5.3.1 sendchar と recvchar の動作

sendchar は、指定された文字列を上から順番に送信します。recvchar は、文字列を送信後、一致する文字列が入出力内容に含まれるかどうかを待ちます。一致する文字列を受信した場合、次の文字列を送信します。

※下記の図は、recvchar オプションを指定した場合の例となります。

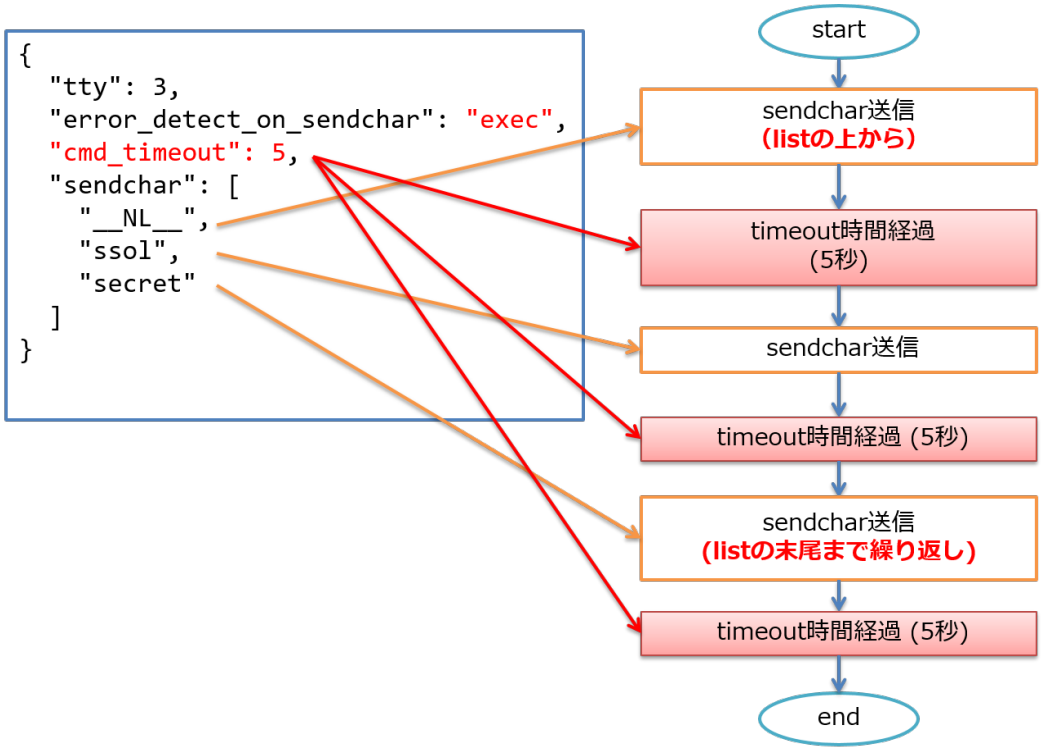


5.3.2 recvchar を設定しない場合の動作

recvchar（recvchar_regex）を指定しなかった場合、sendchar は cmd_timeout 時間の経過を待って次の文字列を送信します。

※/ttymanage のリクエストボディーとして渡す JSON オプションの必須オプションは、tty と sendchar オプションのみとなります。

※下記の図は、recvchar オプションを指定しない場合の例となります。



5.3.3 sendchar の特殊な設定

sendchar は指定した文字列を送信する以外にもオプションを指定する事で特殊な送信方法を設定する事ができます。

1. 改行文字列だけを送信する。

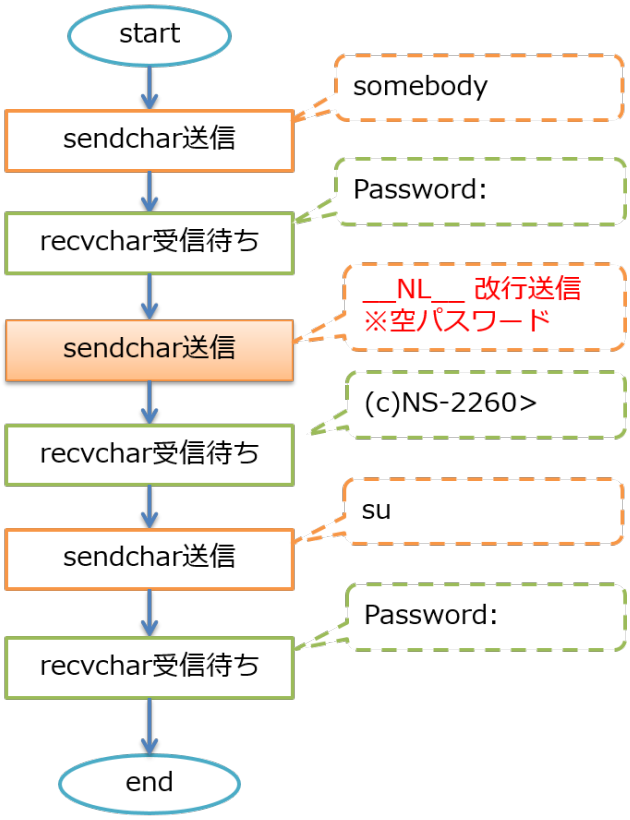
__NL__オプション

改行文字だけを送信する場合、送信文字列として”__NL__”を設定します。送信される改行コードは、nl オプションで設定した値となります。

コンソール経由でログイン処理のオペレーションを行う時などのパスワード入力時に、空パスワードを設定する場合などに使う事ができます。

＜__NL__オプションの使用例＞

```
{
  "tty": 1,
  "recvchar": [
    "login: ",
    "Password: ",
    "NS-2260> ",
    "NS-2260# "
  ],
  "sendchar": [
    "somebody",
    "__NL__",
    "su",
    "__NL__",
    "show version"
  ]
}
```



2. 送信文字列毎に、タイムアウト時間を設定する。

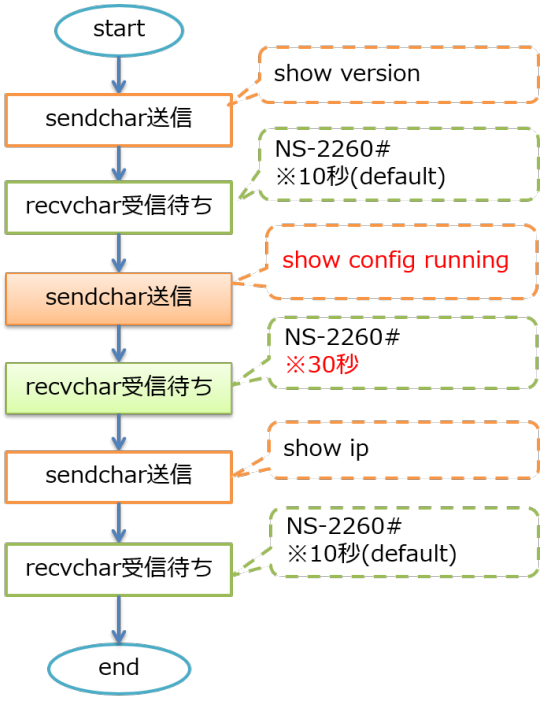
__WAIT__:sec オプション

sendchar で指定した文字列は、cmd_timeout オプションで設定された時間（デフォルト 10 秒）、recvchar で設定された文字列を待ちます。
文字列の送信によって SmartCS に接続されている機器のコンソールで実行されるコマンドが、結果を出力するまでに時間が掛かる場合（ランニングコンフィグの取得コマンドやサポート情報の取得コマンドの実行）、特定の送信文字列のみタイムアウト時間を変更する事ができます。

以下の例の場合、recvchar のタイムアウト値はデフォルトの 10 秒ですが、「show config running」文字列の送信時のみ、タイムアウト時間を 30 秒に設定し動作します。

＜__WAIT__:sec オプションの使用例＞

```
{
  "tty": 1,
  "recvchar": [
    "login: ",
    "Password: ",
    "NS-2260> ",
    "NS-2260# "
  ],
  "sendchar": [
    "show version",
    "show config running __WAIT__:30",
    "show ip"
  ]
}
```



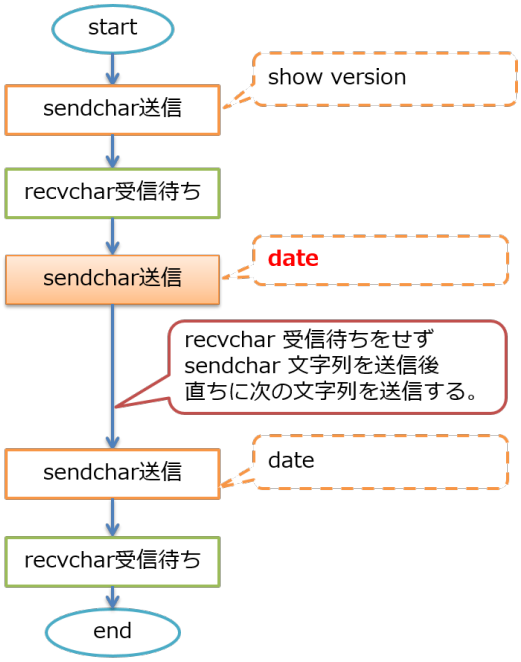
3. 送信文字列毎に、recvchar を待たない設定をする。

__NOWAIT__ オプション

cmd_timeout オプションで設定された時間を待たずにすぐ次の文字列を送信するオプションとなります。
※約1秒後に送信します。
コンソール接続先に対して、recvchar を待たずに連続して文字列を送信したい場合などに使う事ができます。

＜__NOWAIT__オプションの使用例＞

```
{
  "tty": 1,
  "cmd_timeout": 5,
  "recvchar": [
    "login: ",
    "Password: ",
    ">",
    "# "
  ],
  "sendchar": [
    "show version",
    "date __NOWAIT__",
    "date"
  ]
}
```



4. 送信文字列毎に、recvchar を待たずに時間だけで待つように設定する。

__NOWAIT__:sec オプション

recvchar の設定がある場合、文字列送信後の入出力結果に recvchar が含まれているかの確認を行い、含まれている場合に次の文字列の送信を行います。ただし、SmartCS に接続されている機器のコンソールに対して行いたいオペレーションによっては、これらの基本動作によって意図通りに動作させる事ができない場合があります。

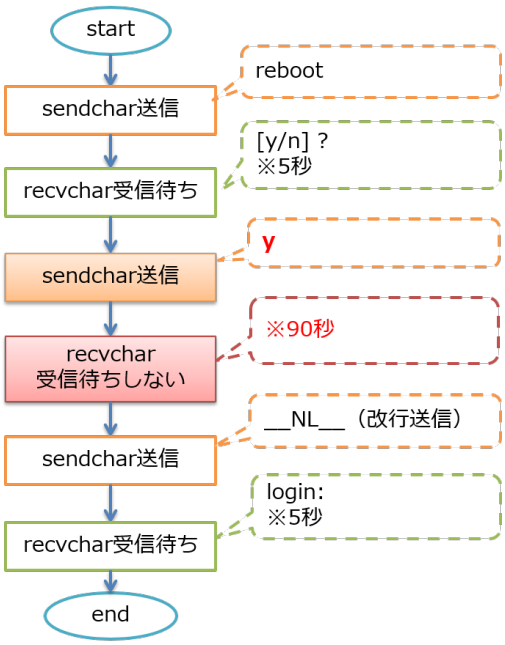
（例）

- recvchar に ”#”、”>” といった文字列を設定し 本来であれば SmartCS に接続されている機器のプロンプトを待ちたいが、実行したコマンドの出力に “>” が含まれており、次の文字列が送信されてしまう場合。
- リブート や バージョンアップコマンドを実行した際、コンソールに様々な文字や記号が出力されてしまう為、意図せず recvchar にマッチしてしまい、リブート中などに次の文字列が送信されてしまう場合。

上記のようなシチュエーションでも出来る限り意図通りコンソールオペレーションが行えるように、送信文字列毎に recvchar を待たない設定をする事ができます。

＜__NOWAIT__:sec オプションの使用例＞

```
{
  "tty": 1,
  "cmd_timeout": 5,
  "recvchar": [
    "login: ",
    "Password: ",
    ">",
    "#",
    "[y/n] ?"
  ],
  "sendchar": [
    "show version",
    "reboot",
    "y __NOWAIT__:90",
    "__NL__",
    "show version"
  ]
}
```



5. 制御文字を送信する

__CTL__ オプション

制御文字を送信する場合、sendchar に「__CTL__:hex」を指定します。送信可能な制御文字は以下の範囲となります。

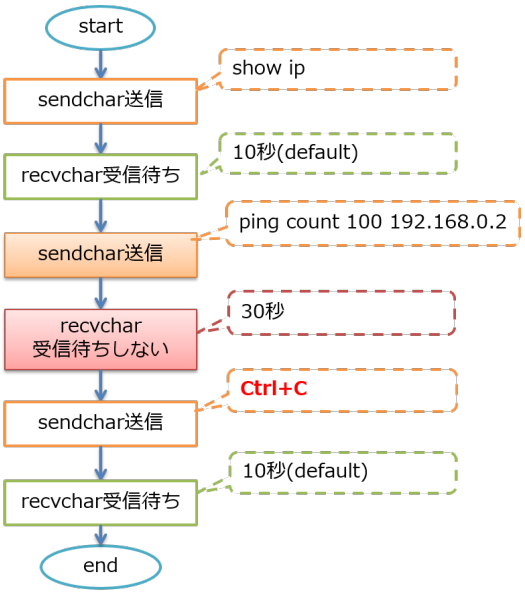
00：[Ctrl-@]	08：[Ctrl-H]	10：[Ctrl-P]	18：[Ctrl-X]
01：[Ctrl-A]	09：[Ctrl-I]	11：[Ctrl-Q]	19：[Ctrl-Y]
02：[Ctrl-B]	0a：[Ctrl-J]	12：[Ctrl-R]	1a：[Ctrl-Z]
03：[Ctrl-C]	0b：[Ctrl-K]	13：[Ctrl-S]	1b：[Ctrl-[ ] / ESC
04：[Ctrl-D]	0c：[Ctrl-L]	14：[Ctrl-T]	1c：[Ctrl-¥]
05：[Ctrl-E]	0d：[Ctrl-M]	15：[Ctrl-U]	1d：[Ctrl-]]
06：[Ctrl-F]	0e：[Ctrl-N]	16：[Ctrl-V]	1e：[Ctrl-^]
07: [Ctrl-G]	0f：[Ctrl-O]	17：[Ctrl-W]	1f：[Ctrl-_]
			7f：[Delete] / Ctrl-?

※一番左の値が__CTL__:hex の hex 部分となります。

コンソール経由で制御文字を送信する場合に使うことができます。
例：ping の実行を停止する。特定の NW 機器のコマンド実行後に送信する。等

＜__CTL__:hex オプションの使用例＞

```
{
  "tty": 1,
  "cmd_timeout": 5,
  "recvchar": [
    "login: ",
    "Password: ",
    ">",
    "#"
  ],
  "sendchar": [
    "show ip",
    "ping count 100 192.168.0.2 __NOWAIT__:30",
    "__CTL__:03"
  ]
}
```



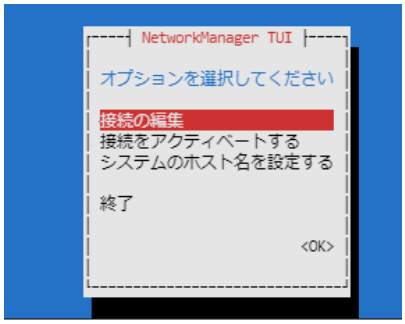
6. 制御文字を複数まとめて送信する

__HEX__ オプション

制御文字を複数まとめて送信する場合、sendchar に「__HEX__:hex」を指定します。送信可能な制御文字 00 ～ 7F の範囲となります。__HEX__オプションは、recvchar で指定した受信文字列を待ちません。

コンソール経由で制御文字をまとめて送信するシチュエーションとしては以下のような内容を想定しています。

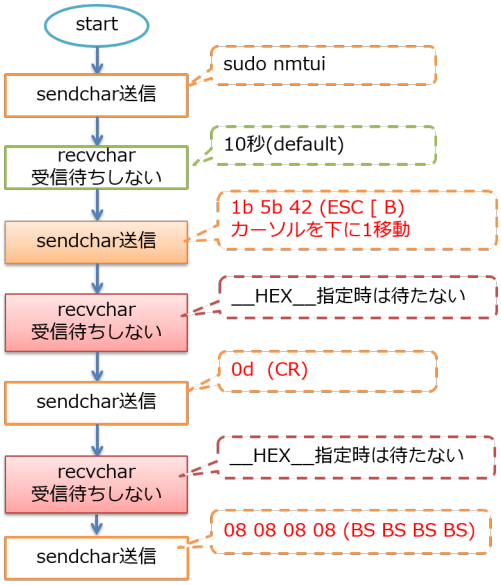
端末エミュレータ上で、Linux の nmtui コマンドなどを利用する場合に、Curses の操作(↑ ↓のカーソル移動など)を行う場合



sendchar で送信する文字列がサポート外の場合に、直接文字コードを指定して送信する場合

＜__HEX__:hex オプションの使用例＞

```
{
  "tty": 16,
  "recvchar": [
    "login: ",
    "Password: ",
    "~]$ ",
    "]# "
  ],
  "sendchar": [
    "sudo nmtui __NOWAIT__",
    "__HEX__: 1b 5b 42",
    "__HEX__: 0d",
    "__HEX__: 08 08 08 08",
    "__HEX__: 54 65 73 74",
    "__HEX__: 1b 5b 42",
    "__HEX__: 0d",
    "exit"
  ]
}
```



5.3.4 error_detect_on_sendchar の動作

sendchar で指定した文字列を送信する場合、以下の理由で文字列の送信がエラーとなる場合があります。

＜文字列の送信がエラーとなる要因＞

エラー要因		要因
recvchar をタイムアウト時間までに受信出来なかった		Error:: Timeout.
対象の tty に接続できない	接続許可設定がない為、接続できない。	Error:: Not allowed.
	排他制御により接続できない。	Error:: Session limit over.
	tty 管理用デーモンに接続ができない	Error:: Connection closed.
	error_recvchar_regex で設定した文字列を検出した。	Error:: Matched “xxx”.
error_detect_on_sendchar 設定が”cancel”の時に、次の送信文字列を送信しない		Error:: After error.

これらのエラーが発生した場合に次の文字列を送信してしまうと、本来想定していたオペレーションと異なる動作となってしまう恐れがあります、その為

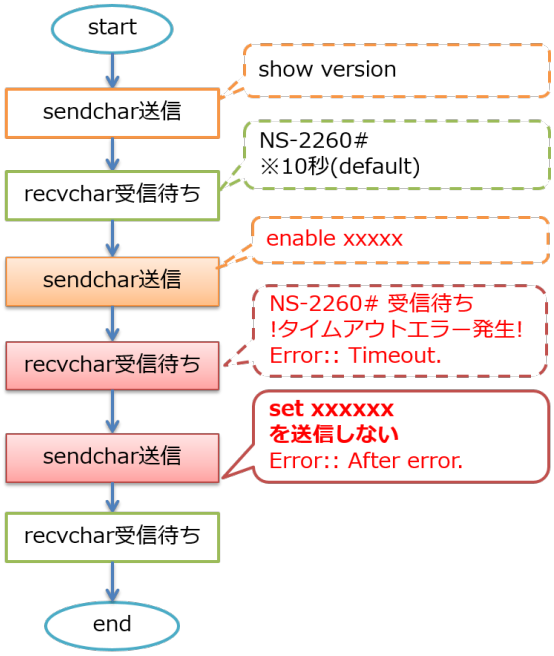
・エラー発生後も文字列をそのまま送信するか

・エラー発生後には文字列を送信しないか

についての動作を設定するオプションとして、error_detect_on_sendchar を用意しています。

1. error_detect_on_sendchar:cancel 設定時の動作

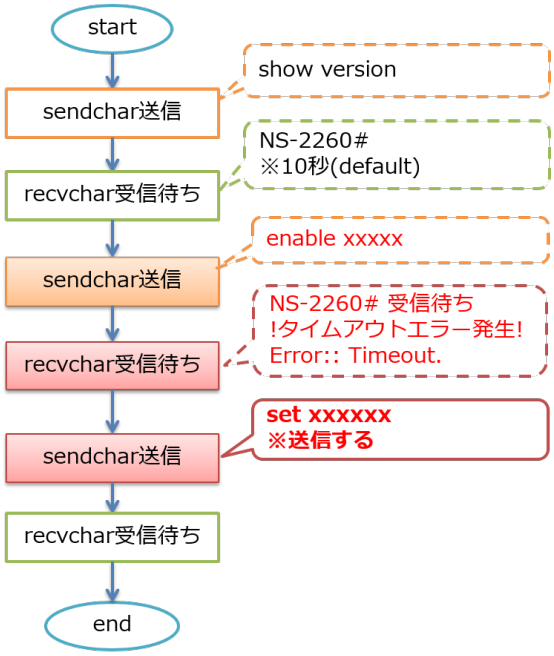
```
{
  "tty": 1,
  "error_detect_on_sendchar": "cancel",
  "recvchar": [
    "login: ",
    "Password: ",
    "NS-2260>",
    "NS-2260#"
  ],
  "sendchar": [
    "show version",
    "enable xxxxx",
    "set xxxxx"
  ]
}
```



※デフォルト値は、error_detect_on_sendchar:cancel です。

2. error_detect_on_sendchar:exec 設定時の動作

```
{
  "tty": 1,
  "error_detect_on_sendchar": "exec",
  "recvchar": [
    "login: ",
    "Password: ",
    "NS-2260>",
    "NS-2260#"
  ],
  "sendchar": [
    "show version",
    "enable xxxxx",
    "set xxxxx"
  ]
}
```



5.4　sendchar の送信オプション一覧

sendchar で送信できる方法の組み合わせは以下の通りとなります。

設定方法	備考
show version	文字列を送信
show version __WAIT__:sec	文字列を送信後、設定した時間 recvchar を待つ
show version __NOWAIT__	文字列を送信後、ただちに次の文字列を送信する。
show version __NOWAIT__:sec	文字列を送信後、設定した時間だけで待つ。
__NL__	改行を送信
__NL__ __WAIT__:sec	改行を送信後、設定した時間 recvchar を待つ
__NL__ __NOWAIT__	改行を送信後、ただちに次の文字列を送信する。
__NL__ __NOWAIT__:sec	改行を送信後、設定した時間だけで待つ。
__CTL__:03	制御文字を送信する。
__CTL__:03 __WAIT__:sec	制御文字を送信後、設定した時間 recvchar を待つ。
__CTL__:03 __NOWAIT__:sec	制御文字を送信後、設定した時間待って次の文字列を送信する。
__CTL__:03 __NOWAIT__	制御文字を送信後、ただちに次の文字列を送信する。

設定方法	備考
__HEX__: 54 65 73 74	16 進数で制御文字や制御コード、ASCII の文字列など様々なデータを送信します。
__HEX__: 54 65 73 74 __WAIT__:sec	16 進数で指定したデータを送信後、設定した時間 recvchar を待つ
__HEX__: 54 65 73 74 __NOWAIT__	16 進数で指定したデータを送信後、ただちに次の文字列を送信する。
__HEX__: 54 65 73 74 __NOWAIT__:sec	16 進数で指定したデータを送信後、設定した時間だけで待つ。

※"show version"部分は何らかの送信文字列を指定した場合の例となります。
※"__CTL__"で指定している 03 は、ctrl+C を指定した場合の例となります。
※"__HEX__"で指定している(54 65 73 74)は、ASCII の"Test"を指定した場合の例となります。
※"__HEX__"指定時は、recvchar で指定している文字列を待ちません。

5.5 正規表現を設定する

/ttymanage の API リソースを指定して、JSON 形式データでシリアルオペレーションの送受信シナリオを作成する場合、以下のオプション内において正規表現で設定を記載する事が可能です。

・recvchar_regex

・error_recvchar_regex

これらのオプションで設定できる正規表現について以下に記載します。

(1) 単一文字とマッチする表現

.	任意の1文字にマッチします。
[…]	(…は任意の文字)指定された任意の1文字にマッチします。
[^…]	(…は任意の文字)指定されていない任意の 1 文字にマッチします。
\k	(k が非英数字文字) 文字としてマッチします。
\d	0 から 9 の数字 1 文字にマッチします。
\D	\d 以外の 1 文字にマッチします。
\s	いずれかの空白文字にマッチします。
\S	\s 以外の 1 文字にマッチします。
\w	英数字と"_"(アンダーバー)の 1 文字にマッチします。
\W	\w 以外の 1 文字にマッチします。
\r	CR(0x0d)にマッチします。
\n	LF(0x0a)にマッチします。

(2) 付加することで反復したマッチを表す表現

*	0 回以上の反復マッチとなります。
+	1 回以上の反復マッチとなります。
?	0 回か 1 回のマッチとなります。
{*m*}	(*m* は 0 以上の整数)ちょうど *m* 回の反復マッチとなります。
{*m*,}	(*m* は 0 以上の整数) *m* 回以上の反復マッチとなります。
{*m*,*n*}	(*m*,*n* はそれぞれ 0 以上の整数) *m* 回から *n* 回までの反復マッチとなります。

(3) その他の表現

(re)	(re はあらゆる正規表現)re にマッチします。
|	この記号によって隔てられているいずれかの表現とマッチします。
[0-9]	0 から 9 の数字 1 文字にマッチします。
[a-z]	a から z の英字 1 文字にマッチします。
[A-Z]	A から Z の英字1文字にマッチします。

(4) 組合せの表現

(^|\n|\r)	行の先頭とマッチします。

(5) 正規表現の記述例

英字(大文字/小文字)、数字、記号(_(アンダーバー)、.(ドット)、-(ハイフン))から成る、複数種類のプロンプトを待ち受けたい場合。

＜マッチする文字列＞

例 : SmartCS_01> 、 SmartCS_01(config)# 、 SmartCS_01(config-if)# 、 SmartCS_01(config-line)#

```
recvchar_regex：
  - "(^|\\n|\\r)[a-zA-Z0-9_.-]*(\\(config)*(-if|-line)*\\)*(>|#)"
```

6　章　付録 A.　ユーザー権限毎のアクセス可能な API リソース

拡張ユーザーに付与する権限毎のアクセス可能な API リソースの一覧について、以下に記載します。

○:アクセス可能

×:アクセス不可

API リソース	メソッド	拡張ユーザーに付与する権限		
		normal	root	ttymanage
/system/version	GET	○	○	×
/users	GET	○	○	×
	POST	×	○	×
/users/{username}	GET	○	○	×
	PUT	×	○	×
	DELETE	×	○	×
/users/login	GET	○	○	×
/serial/tty	GET	○	○	×
/serial/tty/{ttylist}	GET	○	○	×
	PUT	×	○	×
/serial/hangup/tty/{ttylist}	POST	×	○	×
/ttymanage	POST	×	×	○
/log/history/command	GET	×	○	×
/log/history/console	GET	×	○	×
/log/history/ttysend/tty/{ttyno}	GET	×	○	×
/log/history/webapi	GET	×	○	×
/log/serial/tty/{ttyno}	GET	×	×	○
/log/serial/files/tty/{ttyno}	GET	×	×	○
/log/serial/search/tty/{ttyno}	GET	×	×	○

SEIKO
セイコーソリューションズ株式会社
〒261-8507 千葉県千葉市美浜区中瀬 1-8
support@seiko-sol.co.jp