REST API を使用して自動化

SnapCenter Software 4.6

NetApp
August 07, 2024

目次

REST API を使用して自動化 . . . 1
REST API の概要 . . . 1
SnapCenter REST API にネイティブでアクセスする方法 . . . 1
基盤としての REST Web サービス . . . 1
基本的な動作特性 . . . 2
API 要求を制御する入力変数 . . . 4
API 応答の解釈 . . . 7
サポートされている REST API . . . 10
Swagger API Web ページから REST API にアクセスする方法 . . . 17
REST API の使用を開始する . . . 17

REST API を使用して自動化

REST API の概要

REST API を使用して、 SnapCenter のいくつかの管理処理を実行できます。REST API は Swagger Web ページから利用できます。

REST API ドキュメントを表示する場合、および API 呼び出しを手動で問題する場合は、 \\ https://<SnapCenter_IP_address_or_name>:<SnapCenter_port>/swagger/_ で Swagger Web ページにアクセスします。

REST API をサポートするプラグインは次のとおりです。

- Microsoft SQL Server 用プラグイン
- Plug-in for SAP HANA Database の略
- カスタムプラグイン
- Plug-in for Oracle Database の略

SnapCenter REST API にネイティブでアクセスする方法

SnapCenter REST API には、 REST クライアントをサポートする任意のプログラミング言語を使用して直接アクセスできます。代表的な言語の選択肢は、 Python 、 PowerShell 、 Java です。

基盤としての REST Web サービス

Representational State Transfer （ REST ）は、分散 Web アプリケーションの作成に使用される形式です。Web サービス API の設計においては、サーバベースのリソースを公開してその状態を管理するための一連のテクノロジとベストプラクティスが確立されます。主流のプロトコルと標準を使用して、 SnapCenter を管理するための柔軟な基盤を提供しています。

リソースと状態の表示

リソースは、 Web ベースシステムの基本コンポーネントです。REST Web サービスアプリケーションを作成する場合、設計の早い段階で次の作業を行います。

システムまたはサーバベースのリソースの識別

すべてのシステムは、リソースを使用および管理します。リソースには、ファイル、ビジネストランザクション、プロセス、管理エンティティなどがあります。REST Web サービスに基づいてアプリケーションを設計する際に行う最初の作業の 1 つは、リソースを識別することです。

リソースの状態および関連する状態操作の定義

リソースの状態の数は有限で、リソースは必ずそのいずれかの状態にあります。状態、および状態の変化に影響する関連操作を明確に定義する必要があります。

URI エンドポイント

すべての REST リソースは、明確に定義されたアドレス指定方式を使用して定義および使用可能にする必要があります。リソースが置かれているエンドポイントは、 Uniform Resource Identifier （ URI ）で識別されます。

URI は、ネットワーク内の各リソースに一意の名前を作成するための一般的なフレームワークです。Uniform Resource Locator （ URL ）は、リソースを識別してアクセスするために Web サービスで使用される URI の一種です。リソースは通常、ファイルディレクトリに似た階層構造で公開されます。

HTTP メッセージ

Hypertext Transfer Protocol （ HTTP ）は、 Web サービスのクライアントとサーバがリソースに関する要求と応答のメッセージを交換する際に使用するプロトコルです。

Web サービスアプリケーションの設計の一環として、 HTTP メソッドはリソースおよび対応する状態管理アクションにマッピングされます。HTTP はステートレスです。したがって、関連する一連の要求と応答を 1 つのトランザクションの一部として関連付けるには、要求と応答のデータフローで伝送される HTTP ヘッダーに追加情報を含める必要があります。

JSON 形式

Web サービスのクライアントとサーバの間で情報を構造化して転送する方法は複数ありますが、最も広く使用されているのは JavaScript Object Notation （ JSON ）です。

JSON は、単純なデータ構造をプレーンテキストで表すための業界標準であり、リソースについての状態情報の転送に使用されます。SnapCenter REST API では、 JSON を使用して、各 HTTP 要求と応答の本文で伝送されるデータをフォーマットします。

基本的な動作特性

REST で共通のテクノロジとベストプラクティスは確立されますが、各 API の詳細は設計内容に応じて異なる場合があります。

要求と応答の API トランザクション

すべての REST API 呼び出しは、 SnapCenter サーバシステムへの HTTP 要求として実行され、クライアントへの関連する応答が生成されます。この要求と応答のペアで API トランザクションが構成されます。

API を使用する前に、要求の制御に使用できる入力変数と応答出力の内容を理解しておく必要があります。

CRUD 操作のサポート

SnapCenter REST API で使用できる各リソースへのアクセスは CRUD モデルに基づきます。

- 作成
- 読み取り
- 更新
- 削除

一部のリソースでは、一部の処理のみがサポートされます。

オブジェクト ID

各リソースインスタンスまたはオブジェクトには、作成時に一意の識別子が割り当てられます。ほとんどの場合、識別子は 128 ビットの UUID です。これらの識別子は、特定の SnapCenter サーバ内でグローバルに一意です。

新しいオブジェクトインスタンスを作成する API 呼び出しを実行すると、関連付けられた ID を含む URL が HTTP 応答の場所ヘッダーにある呼び出し元に返されます。リソースインスタンスを以降の呼び出しで参照する際には、この識別子を抽出して使用できます。

> オブジェクト識別子の内容と内部構造は、いつでも変更される可能性があります。識別子を使用するのは、該当する API 呼び出しで関連付けられているオブジェクトを参照するときに必要に応じてのみです。

オブジェクトのインスタンスとコレクション

リソースパスと HTTP メソッドに応じて、 API 呼び出しを特定のオブジェクトインスタンスまたはオブジェクトのコレクションに適用できます。

同期操作と非同期操作

SnapCenter は、クライアントから同期または非同期で受信した HTTP 要求を実行します。

同期処理

SnapCenter は要求をただちに実行し、成功した場合は HTTP ステータスコード 200 または 201 を返します。

GET メソッドを使用する要求は、いずれも常に同期的に実行されます。また、 POST を使用する要求は、完了までに 2 秒かからないと予想される場合に、同期的に実行されるように設計されています。

非同期処理

非同期要求が有効な場合、 SnapCenter は要求を処理するバックグラウンドタスクと、タスクのアンカーを設定するジョブオブジェクトを作成します。HTTP ステータスコード 202 がジョブオブジェクトとともに呼び出し元に返されます。成功または失敗を確認するには、ジョブの状態を取得する必要があります。

POST メソッドと DELETE メソッドを使用する要求は、完了までに 2 秒以上かかると予想される場合に非同期で実行するように設計されています。

セキュリティ

REST API のセキュリティは、主に SnapCenter で利用可能な既存のセキュリティ機能に基づいています。API で使用されるセキュリティは次のとおりです。

トランスポートレイヤのセキュリティ

SnapCenter サーバとクライアントの間でネットワークを介して送信されるすべてのトラフィックは、通常、SnapCenter 設定に基づいて TLS を使用して暗号化されます。

HTTP 認証

HTTP レベルでは、 API トランザクションにベーシック認証が使用されます。base64 文字列のユーザ名とパスワードを含む HTTP ヘッダーが各要求に追加されます。

API 要求を制御する入力変数

API 呼び出しの処理方法は、 HTTP 要求で設定されたパラメータと変数を使用して制御できます。

HTTP メソッド

次の表に、 SnapCenter REST API でサポートされる HTTP メソッドを示します。

REST エンドポイントのそれぞれですべての HTTP メソッドを使用できるわけではありません。

HTTP メソッド	説明
取得	リソースインスタンスまたはコレクションのオブジェクトプロパティを取得します。
投稿（ Post ）	指定した入力に基づいて新しいリソースインスタンスを作成します。
削除	既存のリソースインスタンスを削除します。
PUT	既存のリソースインスタンスを変更します。

要求ヘッダー

HTTP 要求には複数のヘッダーを含める必要があります。

コンテンツタイプ

要求の本文に JSON が含まれている場合は、このヘッダーを *application/json* に設定する必要があります。

同意します

このヘッダーは、 *application/json* に設定してください。

承認

base64 文字列としてエンコードされたユーザ名とパスワードを使用するベーシック認証を設定する必要があります。

本文を要求します

要求の本文の内容は、それぞれの呼び出しに応じて異なります。HTTP 要求の本文は、次のいずれかで構成されます。

- JSON オブジェクトと入力変数
- 空です

オブジェクトのフィルタリング

GET を使用する API 呼び出しを発行する際、返されるオブジェクトを任意の属性に基づいて制限またはフィルタできます。たとえば、一致する正確な値を指定できます。

< フィールド >=< クエリ値 >

完全一致に加えて、他の演算子を使用して、一連のオブジェクトを一定範囲の値で返すことができます。次の表に、 SnapCenter REST API でサポートされるフィルタ演算子を示します。

演算子	説明
=	等しい
<	より小さい
>	が次の値より大きい
⇐	が次の値以下です
>=	が次の値以上である必要があります
更新	または
！	と等しくない
*	すべてに一致するワイルドカード

また、クエリの一部として **null** キーワードまたはその negation*!null* を使用して、特定のフィールドが設定されているかどうかに基づいてオブジェクトのコレクションを返すこともできます。

通常、設定されていないフィールドはクエリの照合から除外されます。

特定のオブジェクトフィールドを要求しています

デフォルトでは、 GET を使用する API 呼び出しを発行すると、オブジェクトを一意に識別する属性のみが返されます。この最小のフィールドセットは、各オブジェクトのキーとして機能し、オブジェクトタイプによって異なります。次の方法で 'fields クエリー・パラメータを使用して ' 追加のオブジェクト・プロパティを選択できます

共通または標準のフィールド

fields=* を指定すると、最もよく使用されるオブジェクトフィールドが取得されます。これらのフィールドは、通常、ローカルサーバメモリに保持されるか、ほとんど処理を必要としません。これらのプロパティは、URL パスキー（ UUID ）を指定して GET を使用した場合にオブジェクトに対して返されるプロパティと同じです。

すべてのフィールド

fields=** を指定すると ' アクセスするために追加のサーバ処理が必要なフィールドも含め ' すべてのオブジェクトフィールドが取得されます

カスタムフィールドの選択

fields=<field_name> を使用すると、必要なフィールドを正確に指定できます。複数のフィールドを要求する場合は、スペースを入れずにカンマで区切る必要があります。

> ベストプラクティスとして、必要なフィールドを常に個別に指定することを推奨します。一連の共通フィールドまたはすべてのフィールドを取得するのは、必要な場合だけにしてください。共通として分類されるフィールドで、 *fields=** を使用して返されるフィールドは、ネットアップの内部パフォーマンス分析に基づいて決定されます。フィールドの分類は、今後のリリースで変更される可能性があります。

出力セット内のオブジェクトのソート

リソースコレクション内のレコードは、オブジェクトによって定義されたデフォルトの順序で返されます。フィールド名とソート方向を指定して 'ORDER BY クエリー・パラメータを使用すると ' 順序を次のように変更できます

`order_by=<` フィールド名 `>asc|desc`

たとえば、タイプフィールドを降順でソートし、 ID を昇順でソートできます。

```
order_by=type desc, id asc
```

- ソートフィールドを指定してソートの方向を指定しなかった場合、値は昇順でソートされます。
- 複数のパラメータを指定する場合は、各フィールドをカンマで区切る必要があります。

オブジェクトのコレクションを取得するときのページ付けです

GET を使用する API 呼び出しを発行して同じタイプのオブジェクトのコレクションにアクセスする場合、SnapCenter では 2 つの制約に基づいて可能なかぎり多くのオブジェクトを返します。これらの各制約は、要求に対する追加のクエリパラメータを使用して制御できます。特定の GET 要求に対する最初の制約に達した時点で要求が終了されるため、返されるレコードの数が制限されます。

> すべてのオブジェクトについての処理が完了する前に要求が終了した場合、次のレコードのバッチを取得するために必要なリンクが応答に含まれます。

オブジェクト数の制限

デフォルトでは、 SnapCenter は GET 要求に対して最大 10 、 000 個のオブジェクトを返します。この制限は、 _max_records_query パラメータを使用して変更できます。例：

「 mAX_records =20`

実際に返されるオブジェクトの数は、関連する時間の制約やシステム内のオブジェクトの総数に基づいて、有効な最大数よりも少なくなることがあります。

オブジェクトを読み出す時間を制限しています

デフォルトでは、 SnapCenter は GET 要求に許可された時間内にできるだけ多くのオブジェクトを返します。デフォルトのタイムアウトは 15 秒です。この制限は、 _return_timeout_query パラメータを使用して変更できます。例：

```
re turn _timeout =5
```

実際に返されるオブジェクトの数は、関連するオブジェクト数の制約やシステム内のオブジェクトの総数に基づいて、有効な最大数よりも少なくなることがあります。

結果セットの絞り込み

必要に応じて、これらの 2 つのパラメータを追加のクエリパラメータと組み合わせて、結果セットを絞り込むことができます。たとえば、次の例では、指定した時間のあとに生成された EMS イベントを最大 10 件まで返します。

```
time⇒ 2018-04-04T15:41:29.140265Z & max_records =10
```

複数の要求を問題で処理して、オブジェクトをページングできます。以降の API 呼び出しでは、前回の結果セットの最新イベントに基づいて新しい時間の値を使用する必要があります。

サイズのプロパティ

一部の API 呼び出しおよびクエリパラメータでは、入力値として数値が使用されます。バイト単位で整数を指定する代わりに、必要に応じて次の表に示すサフィックスを使用できます。

サフィックス	説明
KB	KB キロバイト（ 1024 バイト）またはキビバイト
MB	MB （ KB x 1024 バイト）またはメビバイト
GB	ギガバイト（ MB x 1024 バイト）またはギビバイト
容量	TB （ GB x 1024 バイト）またはテビバイト
PB	PB ペタバイト（ TB x 1024 バイト）またはペビバイト

API 応答の解釈

各 API 要求でクライアントへの応答が生成されます。応答を調べて成功したかどうかを

確認し、必要に応じて追加データを取得します。

HTTP ステータスコード

SnapCenter REST API で使用される HTTP ステータスコードを次に示します。

コード	説明
200	OK は、新しいオブジェクトを作成しない呼び出しが成功したことを示します。
201	オブジェクトが作成されました。応答の location ヘッダーにオブジェクトの一意の識別子が含まれます。
202.	承認バックグラウンドジョブで要求の実行が開始されましたが、まだ完了していません。
400	要求が正しくありません。要求の入力が認識されていないか、適切でありません。
401	権限のないユーザ認証に失敗しました。
403	認証（ RBAC ）エラーにより、アクセスが禁止されています。
404	要求で参照されているリソースが見つかりません。
405	メソッドが許可されていません要求内の HTTP メソッドはリソースに対してサポートされていません
409	競合先に別のオブジェクトを作成する必要があるか、要求されたオブジェクトがすでに存在するため、オブジェクトの作成に失敗しました。
500	内部エラーサーバーで一般的な内部エラーが発生しました。

応答ヘッダー

SnapCenter によって生成される HTTP 応答には、いくつかのヘッダーが含まれています。

場所

オブジェクトが作成されると、オブジェクトに割り当てられた一意の識別子を含む、新しいオブジェクトへの完全な URL が location ヘッダーに含まれます。

コンテンツタイプ

通常は 'application/json' です

応答の本文

API 要求の結果として返される応答の本文の内容は、オブジェクト、処理タイプ、および要求の成功または失敗によって異なります。応答は常に JSON 形式になります。

単一のオブジェクト

1 つのオブジェクトを要求に基づいて一連のフィールドとともに返すことができます。たとえば、 GET では、一意の識別子を使用してクラスタの選択したプロパティを取得できます。

複数のオブジェクト

リソースコレクションから複数のオブジェクトを返すことができます。いずれの場合も ' 一貫性のある形式が使用されており 'num_records にはオブジェクト・インスタンスの配列を含むレコードおよびレコードの数が示されますたとえば、特定のクラスタで定義されているノードを取得できます。

ジョブオブジェクト

API 呼び出しが非同期で処理されると、バックグラウンドタスクのアンカーを設定するジョブオブジェクトが返されます。たとえば、クラスタ構成の更新に使用される PATCH 要求は非同期で処理され、ジョブオブジェクトが返されます。

エラーオブジェクト

エラーが発生した場合は、常にエラーオブジェクトが返されます。たとえば、クラスタに定義されていないフィールドを変更しようとするとエラーが表示されます。

空です

場合によっては、データが返されず、応答の本文に空の JSON オブジェクトが含まれることがあります。

エラー

エラーが発生した場合は、応答の本文でエラーオブジェクトが返されます。

の形式で入力し

エラーオブジェクトの形式は次のとおりです。

```
"error": {
"message": "<string>",
"code": <integer>[,
"target": "<string>"]
}
```

code の値で一般的なエラーの種類やカテゴリを特定し、 message で具体的なエラーの内容を確認できます。該当する場合、エラーに関連する特定のユーザ入力がターゲットフィールドに表示されます。

一般的なエラーコード

次の表に、一般的なエラーコードを示します。特定の API 呼び出しについては、追加のエラーコードが含まれる場合があります。

コード	説明
409	同じ識別子のオブジェクトがすでに存在します。
400	フィールドの値が無効であるか、値が指定されていないか、余分なフィールドが指定されています。
400	この処理はサポートされません。
405	指定した識別子のオブジェクトが見つかりません。
403	要求を実行する権限が拒否されました。
409	リソースが使用中です。

サポートされている REST API

他のプラグインでサポートされている REST API

SnapCenter REST API で使用できるリソースは、 SnapCenter API ドキュメントページに表示されるカテゴリ別に分類されています。以下に、各リソースの簡単な概要とベースリソースパスを示し、使用に際しての追加の考慮事項がある場合はその情報も示します。

認証

この API 呼び出しは、 SnapCenter サーバにログインする際に使用できます。この API は、以降の要求の認証に使用するユーザ認証トークンを返します。

ドメイン

次の API 呼び出しを使用して次の処理を実行できます。

- すべてのドメインを取得します
- 特定のドメインの詳細を取得します
- ドメインを登録または登録解除します
- ドメインを変更します

ジョブ

次の API 呼び出しを使用して次の処理を実行できます。

- すべてのジョブを取得します
- ジョブのステータスを取得します
- ジョブをキャンセルまたは停止します

設定

次の API 呼び出しを使用して次の処理を実行できます。

- クレデンシャルを登録、表示、変更、または削除します
- 通知を設定します

ホスト

次の API 呼び出しを使用して次の処理を実行できます。

- ホストの詳細を取得します
- インストールされているプラグインとそのリソースの詳細を取得します
- プラグインホストを追加、削除、または変更する
- プラグインをインストールまたはアップグレードする

リソース

次の API 呼び出しを使用して次の処理を実行できます。

- リソースの読み出し
- リソースを作成、変更、または削除する
- リソースを保護する
- リソースのバックアップ、リストア、クローニング

バックアップ

次の API 呼び出しを使用して次の処理を実行できます。

- バックアップの詳細を取得します
- バックアップの名前変更または削除

クローン

次の API 呼び出しを使用して次の処理を実行できます。

- クローンの詳細を取得します
- クローンを削除します。

それ

次の API 呼び出しを使用して次の処理を実行できます。

- クローンスプリット処理のステータスを取得します
- クローンスプリット処理を開始または停止します

リソースグループ

次の API 呼び出しを使用して次の処理を実行できます。

- リソースグループの詳細を取得する
- リソースグループを作成、変更、または削除する
- リソースグループをバックアップする

ポリシー

次の API 呼び出しを使用して次の処理を実行できます。

- ポリシーの詳細を取得します
- ポリシーを作成、変更、または削除する

ストレージ

次の API 呼び出しを使用して次の処理を実行できます。

- ストレージの詳細を取得します
- ストレージを作成、変更、または削除する
- ストレージ上のリソースを検出
- ストレージ上に共有を作成するか、削除します

共有

次の API 呼び出しを使用して次の処理を実行できます。

- 共有の詳細を取得します
- ストレージ上に共有を作成するか、削除します

プラグイン

これらの API 呼び出しは、ホスト上のすべてのプラグインを取得し、さまざまな処理を実行するために使用できます。

レポート

次の API 呼び出しを使用して次の処理を実行できます。

- バックアップ、リストア、クローニング、およびプラグインのレポートを生成する
- スケジュールを追加、実行、削除、または変更します

アラート

次の API 呼び出しを使用して次の処理を実行できます。

- すべてのアラートを取得します
- アラートを削除します

RBAC

次の API 呼び出しを使用して次の処理を実行できます。

- ユーザ、グループ、およびロールの詳細を取得します
- ユーザを追加します
- ロールを作成、変更、または削除します
- ロールおよびグループを割り当てまたは割り当て解除します

設定

次の API 呼び出しを使用して次の処理を実行できます。

- 構成設定を表示します
- 設定を変更します

CertificateSettings

次の API 呼び出しを使用して次の処理を実行できます。

- 証明書のステータスを表示します
- 証明書の設定を変更します

リポジトリ

次の API 呼び出しを使用して次の処理を実行できます。

- NSM リポジトリをバックアップしてリストアします
- NSM リポジトリを保護し、保護を解除します
- フェイルオーバー
- NSM リポジトリをリビルドします

Oracle データベースでサポートされる REST API

Oracle データベースでは、 REST API を使用して次の処理を実行します。

クレデンシャル

- SnapCenter サーバにクレデンシャルを登録します
- SnapCenter サーバに登録されたクレデンシャルを取得します
- 名前でクレデンシャルを取得します
- クレデンシャルの変更
- 資格情報を削除します

ホスト

- ホストを追加します
- SnapCenter から 1 つ以上のホストを削除します
- 名前でホストを取得します

プラグイン

- 既存のホストへのプラグインのインストール
- プラグインをアップグレードする

ポリシー

- 新しいポリシーを作成する
- 名前を指定してポリシーを取得します
- ポリシーを変更する
- ポリシーを削除します

リソースグループ

- リソースグループを保護する
- 保護を変更します
- 保護を解除します
- 新しいリソースグループを作成する
- 名前を指定してリソースグループを取得します
- リソースグループの変更
- リソースグループを削除する

リソース

- データベースを検出します
- データベースを設定する

バックアップ

- リソースのバックアップ処理を開始する
- バックアップ処理を開始する
- 名前を指定してバックアップを取得します
- バックアップを取得します
- 名前によるバックアップの削除

リストア

- バックアップをリストアします

クローンの更新

- 指定したバックアップから Oracle データベースのクローン仕様を作成します
- クローン固有のファイルを表示、変更、および削除します
- クローンの更新

RBAC

- ユーザまたはグループにリソースを割り当てます
- ユーザまたはグループへのリソースの割り当てを解除します

構成設定

- 構成設定を取得および変更します

SnapCenter サーバのディザスタリカバリでサポートされる REST API

SnapCenter ディザスタリカバリ（ DR ）機能では、 REST API を使用して SnapCenter サーバをバックアップします。REST API を使用すると、 REST API Swagger ページで次の処理を実行できます。Swagger ページへのアクセス方法については、を参照してください "swagger API Web ページを使用して REST API にアクセスする方法"。

- 必要なもの *
- SnapCenter 管理者ユーザとしてログインする必要があります。
- DR リストア API を実行するには、 SnapCenter サーバが稼働している必要があります。
- このタスクについて *

SnapCenter Server DR はすべてのプラグインをサポートします。

ステップ	説明	REST API	HTTP メソッド
1.	既存の SnapCenter サーバ DR バックアップを取得します DR バックアップを格納するバックアップの名前とターゲットパスを指定する必要があります。	`/4.5/disasterrecovery/sa/backup?TargetPath={path}`	取得

ステップ	説明	REST API	HTTP メソッド
2.	新しいサーバ DR バックアップを作成します。指定したサーバ DR バックアップから SnapCenter サーバをリストアします。	「 /4.5/disasterrecovery/sa/backup 」	投稿（ Post ）
3.	指定したサーバ DR バックアップから SnapCenter サーバをリストアします。 • 前提条件 * • 代替サーバのホスト名はプライマリサーバと同じにする必要がありますが、 IP アドレスは異なっていてもかまいません。 • サーバのバージョンはプライマリサーバと同じである必要があります。 • ホスト名はプライマリサーバと同じにする必要があります。 • 次のコマンドを使用して、 DR バックアップファイルが新しい SnapCenter サーバにコピーされていることを確認します。 xcopy <Ssource_Path>\< インストール先サーバー _ip>\< フォルダパス >/O/X/E/H/K {ex:xcopy C:\DRBackup\\10.225.81.114\c$\DRBackup/O/X/H/X/K}` プラグインがサーバのホスト名を解決できない場合は、各プラグイン・ホストにログインし、新しい IP の /etc/hosts エントリを「 <New IP> SC_Server_Name 」の形式で追加します たとえば、 10.225.81.35 SCServer1 と入力します サーバの /etc/hosts エントリはリストアされません。DR バックアップフォルダから手動でリストアできます。	「 /4.5/disasterrecovery/sa/restore 」	投稿（ Post ）
4.	バックアップ名に基づいて Server DR バックアップを削除します。	`/4.5/disasterrecovery/sa/backup`	削除

ステップ	説明	REST API	HTTP メソッド
5.	ストレージ DR を有効または無効にします	「 /4.5/disasterrecovery/sstorage 」	投稿（ Post ）

詳細については、を参照してください "ディザスタリカバリ API" ビデオ：

Swagger API Web ページから REST API にアクセスする方法

REST API は Swagger Web ページから利用できます。Swagger Web ページにアクセスして SnapCenter サーバ REST API を表示したり、 API を手動で問題呼び出したりできます。REST API を使用して、 SnapCenter サーバの管理やデータ保護処理を行うことができます。

REST API を実行する SnapCenter サーバの管理 IP アドレスまたはドメイン名を確認しておく必要があります。

REST API クライアントを実行するための特別な権限は必要ありません。すべてのユーザが Swagger Web ページにアクセスできます。REST API を使用してアクセスするオブジェクトに対する各権限は、 REST API へのログイン時にトークンを生成するユーザに基づいています。

- 手順 *

1. ブラウザから、 URL を入力して Swagger Web ページにアクセスします。形式は \\ https://<SnapCenter_IP_address_or_name>:<SnapCenter_port>/swagger/_ です。

   REST API URL に、 + 、 . 、 % 、 & の文字が含まれていないことを確認してください。

2. Swagger Explore * フィールドで、 Swagger API のドキュメントが自動的に表示されない場合は、「 \\ https://<SnapCenter_IP_address_or_name>:<SnapCenter_port>/Content/swagger/SnapCenter.yaml_ 」と入力します
3. [* Explore] をクリックします。

   API のリソースタイプまたはカテゴリのリストが表示されます。

4. API リソースタイプをクリックすると、そのリソースタイプの API が表示されます。

SnapCenter REST API の実行時に予期しない動作が発生した場合は、ログファイルを使用して原因を特定し、問題を解決することができます。SnapCenter ユーザー・インターフェイスからログ・ファイルをダウンロードするには、 * Monitor * > * Logs * > * Download * をクリックします。

REST API の使用を開始する

SnapCenter REST API はすぐに使用を開始できます。API にアクセスすると、ライブセットアップでより複雑なワークフロープロセスを使用する前にいくつかの情報を確認できます。

Hello world

システムで簡単なコマンドを実行して、 SnapCenter REST API の使用を開始し、利用可能かどうかを確認できます。

- 必要なもの *
- Curl ユーティリティがシステムで使用できることを確認します。
- SnapCenter サーバの IP アドレスまたはホスト名
- SnapCenter REST API にアクセスする権限を持つアカウントのユーザ名とパスワード。

> クレデンシャルに特殊文字が含まれている場合は、使用するシェルに基づいて Curl で許容される形式で指定する必要があります。たとえば ' 各特殊文字の前にバックスラッシュを挿入するか ' または 'username:password' 文字列全体を一重引用符で囲むことができます

- ステップ *

コマンドラインインターフェイスで、次のコマンドを実行してプラグイン情報を取得します。

curl -X get -u username ： password-k "<a href="https://<ip_address>/api/hosts?fields=IncludePluginInfo"`" class="bare">https://<ip_address>/api/hosts?fields=IncludePluginInfo"`

例

curl -X get -u admin ： password-k’<a href="https://10.225.87.97/api/hosts?fields=IncludePluginInfo"`" class="bare">https://10.225.87.97/api/hosts?fields=IncludePluginInfo"`