概念
ONTAP Select

NetApp
November 09, 2026

This PDF was generated from https://docs.netapp.com/ja-jp/ontap-select-9191/concept_api_rest.html on November 09, 2026. Always check docs.netapp.com for the latest.

目次

概念 1
ONTAP Select クラスターのデプロイと管理のための REST Web サービスについて 1
アーキテクチャと従来の制約 1
リソースと状態の表示 1
URIエンドポイント 1
HTTPメッセージ 1
JSONの形式 2
ONTAP Select Deploy APIへのアクセスについて 2
ユーティリティネイティブユーザーインターフェイスを導入する 2
ONTAP Select Deploy オンライン ドキュメント ページ 2
カスタムプログラム 2
ONTAP Select Deploy APIの基本的な運用特性 2
ハイパーバイザー ホストと ONTAP Select ノード 3
オブジェクトID 3
リクエスト識別子 3
同期呼び出しと非同期呼び出し 3
長時間実行されるジョブの完了を確認する 3
セキュリティ 4
ONTAP Select のリクエストとレスポンスの API トランザクション 4
API要求を制御する入力変数 4
API応答を解釈する 6
ONTAP Select の Job オブジェクトを使用した非同期処理 7
ジョブ オブジェクトを使用して記述された非同期要求 7
API リクエストに関連付けられた Job オブジェクトを照会する 7
非同期要求を発行するための一般的な手順 8

概念

ONTAP Select クラスターのデプロイと管理のための REST Web サービスについて

Representational State Transfer（REST）は、分散型Webアプリケーションを作成するためのスタイルです。WebサービスAPIの設計に適用すると、サーバーベースのリソースを公開し、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、ONTAP Selectクラスタの導入と管理のための柔軟な基盤を提供します。

アーキテクチャと従来の制約

RESTは、2000年にUC Irvineで Roy Fielding が博士論文 "学位論文"の中で正式に提唱しました。これは、一連の制約を通じてアーキテクチャスタイルを定義するもので、これらの制約が集合的にWebベースのアプリケーションと基盤となるプロトコルを改善します。これらの制約により、ステートレス通信プロトコルを使用するクライアント/サーバーアーキテクチャに基づくRESTful Webサービスアプリケーションが確立されます。

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ー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し、維持します。リソースとは、ファイル、ビジネス取引、プロセス、または管理エンティティなどを指します。REST Webサービスに基づいたアプリケーションを設計する際の最初のタスクの1つは、リソースを特定することです。
- リソースの状態と関連する状態操作の定義 リソースは常に有限個の状態のいずれかにあります。状態、および状態変化に影響を与えるために使用される関連操作は、明確に定義されなければなりません。

クライアントとサーバー間でメッセージが交換され、一般的なCRUD（作成、読み取り、更新、削除）モデルに従ってリソースの状態にアクセスしたり変更したりします。

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、提供される必要があります。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

ハイパーテキスト転送プロトコル（HTTP）は、Webサービスクライアントとサーバーがリソースに関するリクエストメッセージとレスポンスメッセージを交換するため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動詞（GETやPOSTなど）はリソースとそれに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、一連の関連するリクエストとレスポンスを1つのトランザクションに関連付けるには、リクエスト/レスポンスのデータフローとともに伝送されるHTTPヘッダーに追加情報を含める必要があります。

JSONの形式

クライアントとサーバー間で情報を構造化して転送する方法はいくつかありますが、最も一般的なオプション（Deploy REST APIで使用されるもの）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現するための業界標準であり、リソースを記述する状態情報の転送に使用されます。

ONTAP Select Deploy APIへのアクセスについて

REST Web サービスの本来の柔軟性により、ONTAP Select Deploy API には、いくつかの異なる方法でアクセスできます。

ONTAP Select Deploy に付属の REST API にはバージョン番号が割り当てられます。API のバージョン番号は、Deploy のリリース番号とは無関係です。ONTAP Select 9.19.1 Deploy 管理ユーティリティには、バージョン 3 の REST API が含まれています。

ユーティリティネイティブユーザーインターフェイスを導入する

APIにアクセスする主な方法は、ONTAP Select Deploy Webユーザーインターフェイスからアクセスすることです。ブラウザはAPIを呼び出し、ユーザーインターフェースのデザインに合わせてデータを再フォーマットします。DeployユーティリティのコマンドラインインターフェースからもAPIにアクセスできます。

ONTAP Select Deploy オンライン ドキュメント ページ

ONTAP Select Deploy オンライン ドキュメント ページは、ブラウザを使用する際に代替のアクセス ポイントを提供します。このページでは、個々の API 呼び出しを直接実行する方法を提供するだけでなく、各呼び出しの入力パラメータやその他のオプションを含む、API の詳細な説明も掲載しています。API 呼び出しは、いくつかの異なる機能領域またはカテゴリに分類されています。

カスタムプログラム

Deploy APIには、さまざまなプログラミング言語やツールを使用してアクセスできます。人気のある選択肢としては、Python、Java、cURLなどが挙げられます。APIを使用するプログラム、スクリプト、またはツールは、REST Webサービスクライアントとして機能します。プログラミング言語を使用することで、APIをよりよく理解でき、ONTAP Selectデプロイメントを自動化する機会が得られます。

ONTAP Select Deploy APIの基本的な運用特性

RESTは共通の技術とベストプラクティスを確立するものの、各APIの詳細は設計上の選択によって異なる場合があります。APIを使用する前に、ONTAP Select Deploy APIの詳細と動作特性を把握しておく必要があります。

ハイパーバイザー ホストと **ONTAP Select** ノード

_ハイパーバイザー ホスト_は、ONTAP Select仮想マシンをホストするコア ハードウェア プラットフォームです。ONTAP Select仮想マシンがハイパーバイザー ホスト上に導入されてアクティブになると、その仮想マシンは_ONTAP Selectノード_とみなされます。Deploy REST APIのバージョン3では、ホスト オブジェクトとノード オブジェクトは別個のものです。これにより、1対多の関係が可能になり、1つ以上のONTAP Selectノードを同じハイパーバイザー ホスト上で実行できます。

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は、作成時に一意の識別子が割り当てられます。これらの識別子は、ONTAP Select Deployの特定のインスタンス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呼び出しを発行すると、関連付けられたID値がHTTPレスポンスの `location`ヘッダーで呼び出し元に返されます。識別子を抽出して、リソースインスタンスを参照する際の以降の呼び出しで使用できます。

> ⓘ オブジェクト識別子の内容と内部構造は変更になることがあります。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ために必要な場合だけにしてください。

リクエスト識別子

成功したAPIリクエストにはすべて、一意の識別子が割り当てられます。識別子は、関連するHTTPレスポンスの `request-id`ヘッダーで返されます。リクエスト識別子を使用すると、特定の単一のAPIリクエスト・レスポンス取引における一連の活動をまとめて参照できます。例えば、リクエストIDに基づいて、トランザクションに関するすべてのイベントメッセージを取得できます。

同期呼び出しと非同期呼び出し

サーバーがクライアントから受信したHTTPリクエストを処理する主な方法は2つあります。

- 同期型：サーバーはリクエストを即座に実行し、ステータスコード200、201、または204で応答します。
- 非同期：サーバーはリクエストを受け入れ、ステータスコード202で応答します。これは、サーバーがクライアントからのリクエストを受け入れ、リクエストを完了するためのバックグラウンドタスクを開始したことを示しています。最終的な成功または失敗はすぐには判明せず、追加のAPI呼び出しによって判定する必要があります。

長時間実行されるジョブの完了を確認する

一般的に、完了までに時間がかかる可能性のある操作は、サーバー側のバックグラウンドタスクを使用して非同期的に処理されます。Deploy REST APIでは、すべてのバックグラウンドタスクは、タスクを追跡し、現在の状態などの情報を提供するJobオブジェクトによってアンカーされます。バックグラウンドタスクが作成されると、一意の識別子を含むJobオブジェクトがHTTPレスポンスで返されます。

Jobオブジェクトを直接照会することで、関連するAPI呼び出しの成否を判断できます。詳細については、「Jobオブジェクトを使用した非同期処理」を参照してください。

Job オブジェクトを使用する以外にも、リクエストの成功または失敗を判断する方法はいくつかあります。

- イベントメッセージ 元のレスポンスで返されたリクエストIDを使用すると、特定のAPI呼び出しに関連付けられたすべてのイベントメッセージを取得できます。イベントメッセージには通常、成功または失敗を

示す情報が含まれており、エラー状態のデバッグにも役立ちます。

- リソースの状態またはステータス 一部のリソースは、状態またはステータス値を保持しており、これを照会することでリクエストの成功または失敗を間接的に判断できます。

セキュリティ

Deploy API は、以下のセキュリティ技術を使用しています：

- トランスポート層セキュリティ Deployサーバーとクライアント間のネットワーク上で送信されるすべてのトラフィックは、TLSによって暗号化されます。暗号化されていないチャネル上でHTTPプロトコルを使用することはサポートされていません。TLSバージョン1.2をサポートしています。
- HTTP認証　すべてのAPIトランザクションには基本認証が使用されます。ユーザー名とパスワードをbase64文字列で含むHTTPヘッダーが、すべてのリクエストに追加されます。

ONTAP Select のリクエストとレスポンスの API トランザクション

Deploy API呼び出しはすべて、Deploy仮想マシンへのHTTPリクエストとして実行され、その仮想マシンはクライアントに対して関連する応答を生成します。このリクエストとレスポンスのペアは、APIトランザクションとみなされます。Deploy APIを使用する前に、リクエストを制御するために使用できる入力変数と、レスポンス出力の内容について理解しておく必要があります。

API要求を制御する入力変数

HTTPリクエストに設定されたパラメータを通して、API呼び出しの処理方法を制御できます。

要求ヘッダー

HTTPリクエストには、以下のヘッダーを含める必要があります：

- content-type リクエスト本文に JSON が含まれる場合、このヘッダーは application/json に設定する必要があります。
- accept レスポンスボディに JSON が含まれる場合、このヘッダーは application/json に設定する必要があります。
- authorization 基本認証は、ユーザー名とパスワードをbase64文字列でエンコードして設定する必要があります。

要求の本文

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。

- 入力変数（新しいクラスタの名前など）を含むJSONオブジェクト
- 空の

オブジェクトをフィルタリング

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、一致する完全な値を指定できます。

```
<field>=<query value>
```

完全一致に加えて、値の範囲にわたるオブジェクトのセットを返すための他の演算子も利用可能です。ONTAP Selectは以下のフィルタリング演算子をサポートしています。

演算子	説明
=	等しい
<	小なり
>	より大きい
⇐	次の値以下
>=	次の値以上
|	または
!	等しくない
*	すべてに一致するワイルドカード

また、nullキーワードまたはその否定（!null）をクエリの一部として使用することで、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セットを返すこともできます。

オブジェクトフィールドの選択

デフォルトでは、GETメソッドを使用してAPI呼び出しを行うと、オブジェクトを一意に識別する属性のみが返されます。この最小限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、オブジェクトの種類によって異なります。フィールドクエリパラメータを使用して、以下の方法で追加のオブジェクトプロパティを選択できます：

- 安価なフィールドを指定 `fields=*`して、ローカルサーバーのメモリに保持されている、またはアクセスにほとんど処理を必要としないオブジェクトフィールドを取得します。
- 高価なフィールド `fields=**`を指定して、追加のサーバー処理が必要なフィールドを含むすべてのオブジェクトフィールドを取得します。
- カスタムフィールドの選択 `fields=FIELDNAME`を使用して、希望するフィールドを正確に指定してください。複数のフィールドを要求する場合は、値を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、必要な特定のフィールドを常に特定する必要があります。安価なフィールドまたは高価なフィールドのセットは、必要な場合にのみ取得してください。安価および高価の分類は、内部パフォーマンス分析に基づいてNetAppによって決定されます。特定のフィールドの分類はいつでも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をソートします

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。order_byクエリパラメータにフィールド名とソート方向を指定して、順序を変更できます。

```
order_by=<field name> asc|desc
```

例えば、typeフィールドを降順でソートし、次にidを昇順でソートすることができます。
`order_by=type desc, id asc`

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。

ページネーション

GETを使用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API呼び出しを発行すると、デフォルトで一致するすべてのオブジェクトが返されます。必要に応じて、リクエストでmax_recordsクエリパラメータを使用して、返されるレコード数を制限できます。例：
`max_records=20`

必要に応じて、このパラメータを他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。例えば、以下のコードは、指定された時刻以降に生成された最大10個のシステムイベントを返します：
`time⇒ 2019-04-04T15:41:29.140265Z&max_records=10`

イベント（または任意のオブジェクトタイプ）をページ送りするために、複数のリクエストを発行できます。以降のAPI呼び出しでは、前回の結果セットに含まれる最新のイベントに基づいて、新しい時間値を使用する必要があります。

API応答を解釈する

各APIリクエストは、クライアントへの応答を生成します。応答を検証することで、それが成功したかどうかを判断し、必要に応じて追加データを取得できます。

HTTPステータス コード

Deploy REST API で使用される HTTP ステータスコードを以下に説明します。

Code	説明	説明
200	OK	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	Created	オブジェクトが正常に作成されました。ロケーション応答ヘッダーには、オブジェクトの一意の識子が含まれています。
202	Accepted	リクエストを実行するために長時間実行されるバックグラウンドジョブが開始されましたが、処理はまだ完了していません。
400	Bad request	要求の入力が認識されないか不適切です。
403	Forbidden	認証エラーのためアクセスが拒否されました。
404	Not found	要求で参照されているリソースが存在しません。
405	Method not allowed	リクエストで使用されているHTTP動詞は、このリソースではサポートされていません。
409	競合	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	サーバで一般的な内部エラーが発生しました。
501	未実装	URIは既知だが、リクエストを実行する能力がない。

応答ヘッダー

Deployサーバーによって生成されるHTTPレスポンスには、以下のヘッダーが含まれます：

- request-id 成功したすべてのAPIリクエストには、一意のリクエスト識別子が割り当てられます。
- オブジェクトが作成されると、location ヘッダーには、一意のオブジェクト識別子を含む、新しいオブジェクトへの完全な URL が含まれます。

応答の本文

APIリクエストに関連付けられたレスポンスの内容は、オブジェクト、処理タイプ、およびリクエストの成功または失敗によって異なります。レスポンスボディはJSON形式で出力されます。

- 単一オブジェクト リクエストに基づいて、一連のフィールドを持つ単一のオブジェクトが返されます。例えば、GETを使用して、一意の識別子を用い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、一貫した形式が使用され、 `num_records`でレコード数が示され、recordsにオブジェクトインスタンスの配列が含まれます。たとえば、特定のクラスタで定義されているすべてのノードを取得できます。
- JobオブジェクトAPI呼び出しが非同期で処理される場合、バックグラウンドタスクのアンカーとなるJobオブジェクトが返されます。例えば、クラスタの導入に使用されるPOSTリクエストは非同期で処理され、Job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、必ずエラーオブジェクトが返されます。たとえば、すでに存在する名前でクラスタを作成しようとすると、エラーが発生します。
- 空の場合：場合によってはデータが返されず、レスポンスボディが空になります。例えば、DELETEを使用して既存のホストを削除した場合、レスポンスボディは空になります。

ONTAP Select の Job オブジェクトを使用した非同期処理

Deploy API の呼び出しの中には、特にリソースを作成または変更する呼び出しは、他の呼び出しよりも完了に時間がかかる場合があります。ONTAP Select Deploy は、これらの長時間実行されるリクエストを非同期で処理します。

ジョブ オブジェクトを使用して記述された非同期要求

非同期的に実行されるAPI呼び出しを行うと、HTTP応答コード202が返されます。この応答コードは、要求が正常に検証され受け入れられたものの、まだ完了していないことを示します。要求はバックグラウンド タスクとして処理され、クライアントへの最初のHTTP応答後も引き続き実行されます。応答には、要求に対応するジョブ オブジェクトと、その一意の識別子が含まれます。

どの API 呼び出しが非同期で動作するかを判断するには、ONTAP Select Deploy オンライン ドキュメント ページを参照してください。

API リクエストに関連付けられた Job オブジェクトを照会する

HTTP応答で返されるジョブ オブジェクトには、いくつかのプロパティが含まれています。状態プロパティを照会して、要求が正常に完了したかどうかを確認できます。ジョブ オブジェクトは次のいずれかの状態になります。

- キュー登録済み
- 実行中
- 成功
- 失敗

ジョブ オブジェクトをポーリングしてタスクの最終状態（成功または失敗）を検出するには、2つの方法があります。

- 標準のポーリング要求 現在のジョブの状態はすぐに返されます
- ロングポーリングリクエストのジョブ状態は、以下のいずれかの条件が満たされた場合にのみ返されます：
  - ポーリング要求で指定された日時値よりも最近、状態が変更されました
  - タイムアウト値が経過しました（1〜120秒）

標準ポーリングとロングポーリングは、ジョブオブジェクトを照会するために同じAPI呼び出しを使用します。ただし、ロングポーリングリクエストには2つのクエリパラメータが含まれます（`poll_timeout`および`last_modified`）。



Deploy 仮想マシンのワークロードを軽減するには、常にロング ポーリングを使用する必要があります。

非同期要求を発行するための一般的な手順

以下は、非同期API呼び出しを完了する手順の概要です。

1. 非同期API呼び出しを実行します。
2. 要求が正常に受け取られたことを示すHTTP応答202を受信します。
3. 応答の本文からジョブ オブジェクトの識別子を抽出します。
4. ループ内で、各サイクルで以下の処理を実行します：
   a. ロングポーリングリクエストを使用してジョブの現在の状態を取得します
   b. ジョブが非終了状態（キューに入っている、実行中）の場合は、ループを再度実行します。
5. ジョブが終了状態（成功、失敗）に達したら停止します。

著作権に関する情報

Copyright © 2026 NetApp, Inc. All Rights Reserved. Printed in the U.S.このドキュメントは著作権によって保護されています。著作権所有者の書面による事前承諾がある場合を除き、画像媒体、電子媒体、および写真複写、記録媒体、テープ媒体、電子検索システムへの組み込みを含む機械媒体など、いかなる形式および方法による複製も禁止します。

ネットアップの著作物から派生したソフトウェアは、次に示す使用許諾条項および免責条項の対象となります。

このソフトウェアは、ネットアップによって「現状のまま」提供されています。ネットアップは明示的な保証、または商品性および特定目的に対する適合性の暗示的保証を含み、かつこれに限定されないいかなる暗示的な保証も行いません。ネットアップは、代替品または代替サービスの調達、使用不能、データ損失、利益損失、業務中断を含み、かつこれに限定されない、このソフトウェアの使用により生じたすべての直接的損害、間接的損害、偶発的損害、特別損害、懲罰的損害、必然的損害の発生に対して、損失の発生の可能性が通知されていたとしても、その発生理由、根拠とする責任論、契約の有無、厳格責任、不法行為（過失またはそうでない場合を含む）にかかわらず、一切の責任を負いません。

ネットアップは、ここに記載されているすべての製品に対する変更を随時、予告なく行う権利を保有します。ネットアップによる明示的な書面による合意がある場合を除き、ここに記載されている製品の使用により生じる責任および義務に対して、ネットアップは責任を負いません。この製品の使用または購入は、ネットアップの特許権、商標権、または他の知的所有権に基づくライセンスの供与とはみなされません。

このマニュアルに記載されている製品は、1つ以上の米国特許、その他の国の特許、および出願中の特許によって保護されている場合があります。

権利の制限について：政府による使用、複製、開示は、DFARS 252.227-7013（2014年2月）およびFAR 5252.227-19（2007年12月）のRights in Technical Data -Noncommercial Items（技術データ - 非商用品目に関する諸権利）条項の(b)(3)項、に規定された制限が適用されます。

本書に含まれるデータは商用製品および / または商用サービス（FAR 2.101の定義に基づく）に関係し、データの所有権はNetApp, Inc.にあります。本契約に基づき提供されるすべてのネットアップの技術データおよびコンピュータ ソフトウェアは、商用目的であり、私費のみで開発されたものです。米国政府は本データに対し、非独占的かつ移転およびサブライセンス不可で、全世界を対象とする取り消し不能の制限付き使用権を有し、本データの提供の根拠となった米国政府契約に関連し、当該契約の裏付けとする場合にのみ本データを使用できます。前述の場合を除き、NetApp, Inc.の書面による許可を事前に得ることなく、本データを使用、開示、転載、改変するほか、上演または展示することはできません。国防総省にかかる米国政府のデータ使用権については、DFARS 252.227-7015(b)項（2014年2月）で定められた権利のみが認められます。

商標に関する情報

NetApp、NetAppのロゴ、http://www.netapp.com/TMに記載されているマークは、NetApp, Inc.の商標です。その他の会社名と製品名は、それを所有する各社の商標である場合があります。