



概念

ONTAP Select

NetApp
February 19, 2025

目次

概念	1
基盤となるREST Webサービス	1
Deploy API へのアクセス方法	2
API のバージョン管理を導入	2
基本的な動作特性	3
要求と応答のAPIトランザクション	4
ジョブオブジェクトを使用した非同期処理	7

概念

基盤となるREST Webサービス

Representational State Transfer (REST) は、分散Webアプリケーションを作成するための形式です。WebサービスAPIの設計に適用すると、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ー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が使用されており、ONTAP Select クラスタの導入と管理のための柔軟な基盤が提供されます。

アーキテクチャと従来の制約

RESTは2000年にカリフォルニア大学アーバイン校で博士号を取得したロイ・フィールドینگによって正式に明言された "失策"。アーキテクチャスタイルは、一連の制約によって定義されます。これらの制約は、全体的に 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。制約は、ステートレス通信プロトコルを使用するクライアント / サーバアーキテクチャに基づいて、RESTful Web サービスアプリケーションを確立するものです。

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際の初期設計タスクには、次のものがあります。

- すべてのシステムが使用するシステムまたはサーバベースのリソースを識別し、リソースを維持します。リソースには、ファイル、ビジネスランザクション、プロセス、または管理エンティティがあります。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。
- リソースの状態の定義と関連する状態処理リソースは、常に有限数の状態のいずれかになります。状態、および状態の変化に影響を与えるために使用される関連操作は、明確に定義する必要があります。

クライアントとサーバの間でメッセージを交換しながら、一般的な CRUD (Create、Read、Update、Delete) モデルに従ってリソースにアクセスしてその状態を変更します。

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および利用可能にする必要があります。リソースが配置され、識別されるエンドポイントでは、Uniform Resource Identifier (URI) が使用されます。URIは、ネットワーク内の各リソースに一意的な名前を作成するための一般的なフレームワークを提供します。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 つのランザクションで関連付ける

には、要求 / 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の形式

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方法（Deploy REST API で使用）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。

Deploy API へのアクセス方法

REST Web サービスは柔軟性が高いため、ONTAP Select Deploy API にはいくつかの方法でアクセスできます。

Deploy ユーティリティの標準のユーザインターフェイス

ONTAP Select Deploy の Web ユーザインターフェイスから API にアクセスする場合は、主に API にアクセスします。ブラウザは、API を呼び出して、ユーザインターフェイスの設計に従ってデータを再フォーマットします。また、Deploy ユーティリティのコマンドラインインターフェイスから API にアクセスします。

ONTAP Select Deploy のオンラインドキュメントページです

ONTAP Select Deploy のオンラインドキュメントページでは、ブラウザを使用する際に別のアクセスポイントを使用できます。個々の API 呼び出しを直接実行する方法に加え、各呼び出しの入力パラメータやその他のオプションなど、API の詳細な概要も含まれています。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

さまざまなプログラミング言語やツールを使用して Deploy API にアクセスできます。広く利用されているのは、Python、Java、cURL などです。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 サービスの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と、API をより詳しく理解し、ONTAP Select の導入を自動化することができます。

API のバージョン管理を導入

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せん。API のバージョン番号は、Deploy のリリース番号とは関係ありません。Deploy のリリースに含まれている API のバージョンと、API の使用にどのような影響があるかを確認しておく必要があります。

Deploy 管理ユーティリティの最新リリースには、バージョン 3 の REST API が含まれています。Deploy ユーティリティの以前のリリースには、次のバージョンの API が含まれていました。

2.8 以降を導入します

ONTAP Select Deploy 2.8 以降のすべてのリリースには、REST API のバージョン 3 が含まれています。

2.7.2 以前のバージョンを導入します

ONTAP Select Deploy 2.7.2 およびそれ以前のすべてのリリースには、REST API のバージョン 2 が含ま

れています。



REST API のバージョン 2 と 3 には互換性がありません。バージョン 2 の API を含む以前のリリースから Deploy 2.8 以降にアップグレードする場合は、コマンドラインインターフェイスを使用して、API に直接アクセスする既存のコードおよびスクリプトを更新する必要があります。

基本的な動作特性

RESTでは共通のテクノロジーとベストプラクティスが確立されますが、各APIの詳細は設計の選択内容によって異なります。API を使用する前に、ONTAP Select Deploy API の詳細と運用上の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** ノードです

`a_hypervisor host_` 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 仮想マシンを導入してハイパーバイザーホストでアクティブにすると、その仮想マシンは `_ONTAP Select node_name` 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々になります。これにより、1 つ以上の ONTAP Select ノードを同じハイパーバイザーホストで実行できる 1 対多の関係が実現します。

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。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を呼び出したあと、関連付けられているIDの値がHTTP応答のヘッダーで呼び出し元に返され `location` ます。識別子を抽出して以降の呼び出しでリソースインスタンスを参照する際に使用できます。



オブジェクトIDの内容と内部構造はいつでも変更できます。識別子は、関連するオブジェクトを参照する場合にのみ、該当するAPI呼び出しで使用してください。

要求 ID

成功したすべての API 要求には、一意の識別子が割り当てられます。識別子は、関連付けられたHTTP応答のヘッダーで返され `request-id` ます。要求 ID を使用すると、単一の API 要求と応答のトランザクションのアクティビティをまとめて参照できます。たとえば、要求 ID に基づいて、トランザクションの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

サーバがクライアントから受信した HTTP 要求を実行する主な方法は 2 つあります。

- 同期サーバは要求をただちに実行し、ステータスコード 200、201、または 204 で応答します。
- 非同期サーバは要求を受け入れ、ステータスコード 202 で応答します。これは、サーバがクライアント要求を受け入れ、要求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ます。最終的な成功または失敗はすぐには確認できないため、追加の API 呼び出しで確認する必要があります。

長時間実行されているジョブの完了を確認する

通常、完了までに時間がかかる操作は、サーバでバックグラウンドタスクを使用して非同期に処理されます。Deploy REST API では、タスクを追跡し、現在の状態などの情報を提供するジョブオブジェクトによってすべてのバックグラウンドタスクがアンカーで設定されます。一意の識別子を含むジョブオブジェクトは、バックグラウンドタスクが作成されたあとの HTTP 応答で返されます。

ジョブオブジェクトを直接照会することで、関連する API 呼び出しの成功または失敗を確認できます。ジョブ object_for 追加情報 を使用した非同期処理を参照してください。

ジョブオブジェクトを使用する以外に、要求の成功または失敗を判断する方法がいくつかあります。これには次のものがあります。

- イベントメッセージ特定の API 呼び出しに関連するすべてのイベントメッセージを取得するには、元の応答で返された要求 ID を使用します。通常、イベントメッセージには成功または失敗の兆候が含まれており、エラー状態のデバッグ時にも役立ちます。
- リソースの状態または状態いくつかのリソースには状態またはステータスの値が保持されており、リクエストの成功または失敗を間接的に判断するためにクエリできます。

セキュリティ

Deploy API では、次のセキュリティテクノロジーを使用します。

- Transport Layer Security : Deploy 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、TLS を介して暗号化されます。暗号化されていないチャンネル上で 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 がサポートされています。
- HTTP 認証 : すべての 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 文字列のユーザー名とパスワードを含む HTTP ヘッダーがすべての要求に追加されます。

要求と応答の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 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、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。

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を使用して制御できます。

要求ヘッダー

HTTP 要求には、次のようなヘッダーを含める必要があります。

- content-type 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。
- 応答の本文に JSON が含まれる場合は受け入れます。このヘッダーを application/json に設定する必要があります。

- 認証基本認証は、base64 文字列でエンコードされたユーザ名とパスワードを使用して設定する必要があります。

リクエストの本文

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。

- JSON オブジェクトと入力変数（新しいクラスタの名前など）
- 空

オブジェクトのフィルタ

GETを使用するAPI呼び出しを実行するときに、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リングできます。たとえば、次のように完全に一致する値を指定できます。

<field>=<query value>

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。ONTAP Select では、次のフィルタ演算子がサポートされています。

演算子	製品説明
=	等しい
<	より小さい
>	次の値より大きい
←	以下
>=	以上
	または
なんだ	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を使用してAPI呼び出しを実行すると、オブジェクトを一意に識別する属性のみが返されます。このフィールドの最小セットは、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。fields query パラメータを使用すると、次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。

- 安価なフィールドローカルサーバメモリに保持されているオブジェクトフィールドを取得するか、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指定します fields=*。
- 高価なフィールドは、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を取得するように指定します fields=**。
- カスタムフィールドの選択目的のフィールドを正確に指定するために使用し `fields=FIELDNAME` ます。

複数のフィールドを要求する場合は、値をカンマで区切ってスペースなしで指定する必要があります。



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。安価なフィールドや高コストのフィールドは、必要な場合にのみ取得してください。低コストでコストな分類は、ネットアップが社内パフォーマンス分析に基づいて決定します。特定のフィールドの分類は、いつでも変更できます。

出力セット内のオブジェクトをソートする

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。次のようにフィールド名とソート方向を指定したORDER_BYクエリパラメータを使用して順序を変更できます

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

たとえば、typeフィールドを降順に並べ替え、idを昇順に並べ替えることができます。

```
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。

ページ付け

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、一致するすべてのオブジェクトがデフォルトで返されます。必要に応じて、max_records クエリパラメータを要求とともに使用して返されるレコード数を制限することもできます。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。たとえば、次の例では、指定した時間が経過したあとに生成されたシステムイベントが最大10個返されます。

```
time⇒ 2019-04-04T15:41:29.140265Z&max_records=10
```

複数の要求を問題で送信して、各イベント（または任意のオブジェクトタイプ）をページングできます。以降の各API呼び出しでは、最後の結果セットの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 応答を解釈します

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次に示します。

コード	意味	製品説明
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	作成済み	オブジェクトが作成されました。場所の応答ヘッダーにはオブジェクトの一意の識別子が含まれています。
202	承認済み	長時間の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、処理はまだ完了していません。

コード	意味	製品説明
400	無効な要求です	要求の入力が認識されないか不適切です。
403	禁止	認証エラーによりアクセスが拒否されました。
404	見つかりません	要求で参照されているリソースが存在しません。
405	許可されていないメソッド	要求内の HTTP 動詞はリソースで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	実装されていません	URI は既知ですが、要求を実行できません。

応答ヘッダー

Deploy サーバによって生成される HTTP 応答には、次のようなヘッダーが含まれています。

- request-id 成功したすべての API 要求には、一意の要求 ID が割り当てられます。
- Location : オブジェクトが作成されると、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 が格納されます。

応答の本文

API 要求に関連する応答の内容は、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。応答の本文は JSON 形式になります。

- 単一のオブジェクト単一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 を使用すると、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。
- 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、オブジェクトインスタンスの配列を含むレコードとレコードの数を示す一貫した形式が使用され num_records ます。たとえば、特定のクラスタに定義されているすべての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、クラスタの導入に使用された POST 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、既存の名前を使用してクラスタを作成しようとするエラーが表示されます。
- 空の場合もあります。データが返されず、応答の本文が空になっていることもあります。たとえば、DELETE を使用して既存のホストを削除したあとは、応答の本文が空になります。

ジョブオブジェクトを使用した非同期処理

Deploy API 呼び出し、特にリソースの作成や変更を行う呼び出しは、他の呼び出しよりも完了に時間がかかることがあります。ONTAP Select Deploy は、これらの長時間実行される要求を非同期で処理します。

ジョブオブジェクトを使用した非同期要求の説明

非同期で実行されるAPI呼び出しを実行すると、HTTP応答コード202が返されます。これは、要求が正常に検証されて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、一意の識別子を含む、要求をアンカーするジョブオブジェクトが含まれます。



非同期的に処理するAPI呼び出しを決定するには、ONTAP Select Deploy のオンラインドキュメントページを参照してください。

API要求に関連付けられているジョブ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。

- キューに登録済み
- 実行中
- 成功
- 障害

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、次の2つの方法があります。

- 標準のポーリング要求の現在のジョブの状態がすぐに返されます
- ロングポーリング要求のジョブ状態は、次のいずれかの場合にのみ返されます。
 - 状態が、ポーリング要求で指定された日時の値よりも最近変更されました
 - タイムアウト値が期限切れ（1～120秒）

標準のポーリングとロングポーリングでは、同じAPI呼び出しを使用してジョブオブジェクトが照会されます。ただし、長いポーリング要求には、と `last_modified`` の2つのクエリパラメータが含まれています ``poll_timeout`。



Deploy 仮想マシンのワークロードを減らすためには、常に長いポーリングを使用してください。

非同期要求を発行するための一般的な手順

非同期API呼び出しを完了する手順の概要は次のとおりです。

1. 非同期API呼び出しを実行します。
2. 要求が正常に受け入れられたことを示す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、各サイクルで次の手順を実行します。
 - a. 長時間のポーリング要求で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（待機中、実行中）の場合は、もう一度ループを実行します。
5. ジョブが終了状態（ success または failure ）になったら停止します。

著作権に関する情報

Copyright © 2025 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权を有し、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权については、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。