# 개념
## ONTAP Select

NetApp
May 07, 2026

# 목차

# 개념

## ONTAP Select 클러스터 배포 및 관리를 위한 REST 웹 서비스 기반

REST(Representational State Transfer)는 분산 웹 애플리케이션을 생성하기 위한 스타일입니다. 웹 서비스 API 설계에 적용될 경우, 서버 기반 리소스를 노출하고 상태를 관리하기 위한 기술 및 모범 사례 세트를 설정합니다. 주요 프로토콜과 표준을 사용하여 ONTAP Select 클러스터를 배포하고 관리하기 위한 유연한 기반을 제공합니다.

### 아키텍처 및 기존 제약 조건

REST는 2000년 UC Irvine에서 Roy Fielding이 박사 "논문" 학위 논문에서 공식적으로 정립했습니다. 이는 일련의 제약 조건을 통해 아키텍처 스타일을 정의하며, 이러한 제약 조건들은 웹 기반 애플리케이션과 기반 프로토콜을 개선합니다. 이러한 제약 조건은 상태 비저장 통신 프로토콜을 사용하는 클라이언트/서버 아키텍처 기반의 RESTful 웹 서비스 애플리케이션을 구축합니다.

### 리소스 및 상태 표현

리소스는 웹 기반 시스템의 기본 구성 요소입니다. REST 웹 서비스 애플리케이션을 개발할 때 초기 설계 작업에는 다음이 포함됩니다.

- 시스템 또는 서버 기반 리소스 식별 모든 시스템은 리소스를 사용하고 유지 관리합니다. 리소스는 파일, 비즈니스 트랜잭션, 프로세스 또는 관리 엔티티일 수 있습니다. REST 웹 서비스를 기반으로 하는 애플리케이션을 설계할 때 가장 먼저 해야 할 일 중 하나는 리소스를 식별하는 것입니다.
- 리소스 상태 및 관련 상태 연산의 정의 리소스는 항상 유한한 수의 상태 중 하나에 있습니다. 상태와 상태 변경에 사용되는 관련 연산은 명확하게 정의되어야 합니다.

클라이언트와 서버 간에 메시지가 교환되어 일반적인 CRUD(Create, Read, Update, Delete) 모델에 따라 리소스 상태에 액세스하고 변경합니다.

### URI 엔드포인트

모든 REST 리소스는 잘 정의된 주소 지정 체계를 사용하여 정의되고 사용 가능해야 합니다. 리소스가 위치하고 식별되는 엔드포인트는 URI(Uniform Resource Identifier)를 사용합니다. URI는 네트워크의 각 리소스에 고유한 이름을 생성하기 위한 일반적인 프레임워크를 제공합니다. URL(Uniform Resource Locator)은 웹 서비스에서 리소스를 식별하고 액세스하는 데 사용되는 URI의 한 유형입니다. 리소스는 일반적으로 파일 디렉터리와 유사한 계층 구조로 노출됩니다.

### HTTP 메시지

HTTP(Hypertext Transfer Protocol)는 웹 서비스 클라이언트와 서버가 리소스에 대한 요청 및 응답 메시지를 교환하는 데 사용되는 프로토콜입니다. 웹 서비스 애플리케이션을 설계할 때 GET 및 POST와 같은 HTTP 동사는 리소스 및 해당 상태 관리 작업에 매핑됩니다.

HTTP는 상태를 저장하지 않습니다. 따라서 관련된 요청과 응답들을 하나의 트랜잭션으로 묶으려면 요청/응답 데이터 흐름과 함께 전달되는 HTTP 헤더에 추가 정보를 포함해야 합니다.

### JSON 형식

클라이언트와 서버 간에 정보를 구조화하고 전송하는 방법은 여러 가지가 있지만, 가장 일반적인 방법(그리고 Deploy REST API에서 사용되는 방법)은 JavaScript Object Notation(JSON)입니다. JSON은 간단한 데이터 구조를 일반 텍스트로 표현하기 위한 업계 표준이며, 리소스를 설명하는 상태 정보를 전송하는 데 사용됩니다.

# ONTAP Select Deploy API에 액세스하는 방법

REST 웹 서비스의 고유한 유연성 덕분에 ONTAP Select Deploy API는 여러 가지 방식으로 액세스할 수 있습니다.

ONTAP Select Deploy에 포함된 REST API에는 버전 번호가 할당됩니다. API 버전 번호는 Deploy 릴리스 번호와는 무관합니다. ONTAP Select 9.17.1 Deploy 관리 유틸리티에는 REST API 버전 3이 포함되어 있습니다.

### 유틸리티 네이티브 사용자 인터페이스 배포

API에 액세스하는 주요 방법은 ONTAP Select Deploy 웹 사용자 인터페이스를 통하는 것입니다. 브라우저는 API를 호출하고 사용자 인터페이스의 설계에 따라 데이터를 재구성합니다. 또한 Deploy 유틸리티 명령줄 인터페이스를 통해서도 API에 액세스할 수 있습니다.

### ONTAP Select Deploy 온라인 설명서 페이지

ONTAP Select Deploy 온라인 문서 페이지는 브라우저를 사용할 때 또 다른 접근 경로를 제공합니다. 이 페이지는 개별 API 호출을 직접 실행하는 방법 외에도 각 호출에 대한 입력 매개변수 및 기타 옵션을 포함한 API에 대한 자세한 설명을 제공합니다. API 호출은 여러 기능 영역 또는 범주로 구성되어 있습니다.

### 맞춤형 프로그램

여러 가지 프로그래밍 언어 및 도구를 사용하여 Deploy API에 액세스할 수 있습니다. 널리 사용되는 언어로는 Python, Java, cURL 등이 있습니다. API를 사용하는 프로그램, 스크립트 또는 도구는 REST 웹 서비스 클라이언트 역할을 합니다. 프로그래밍 언어를 사용하면 API를 더 잘 이해하고 ONTAP Select 배포를 자동화할 수 있습니다.

# ONTAP Select Deploy API 기본 작동 특성

REST는 공통된 기술과 모범 사례를 제시하지만, 각 API의 세부 사항은 설계 선택에 따라 달라질 수 있습니다. ONTAP Select Deploy API를 사용하기 전에 해당 API의 세부 사항과 운영 특성을 숙지해야 합니다.

### 하이퍼바이저 호스트와 ONTAP Select 노드

_하이퍼바이저 호스트_는 ONTAP Select 가상 머신을 호스팅하는 핵심 하드웨어 플랫폼입니다. ONTAP Select 가상 머신이 하이퍼바이저 호스트에 배포되어 활성화되면 해당 가상 머신은 _ONTAP Select 노드_로 간주됩니다. 배포 REST API 버전 3부터는 호스트와 노드 객체가 분리되어 있습니다. 이를 통해 하나 이상의 ONTAP Select 노드가 동일한 하이퍼바이저 호스트에서 실행될 수 있는 일대다 관계가 가능해집니다.

## 객체 식별자

각 리소스 인스턴스 또는 객체는 생성될 때 고유 식별자가 할당됩니다. 이러한 식별자는 ONTAP Select Deploy의 특정 인스턴스 내에서 전역적으로 고유합니다. 새 객체 인스턴스를 생성하는 API 호출을 실행하면 관련 ID 값이 HTTP 응답의 `location` 헤더에 포함되어 호출자에게 반환됩니다. 이 식별자를 추출하여 이후 호출에서 해당 리소스 인스턴스를 참조할 때 사용할 수 있습니다.

객체 식별자의 내용 및 내부 구조는 언제든지 변경될 수 있습니다. 따라서 관련 객체를 참조할 때는 필요에 따라 해당 API 호출에서만 식별자를 사용해야 합니다.

## 요청 식별자

모든 성공적인 API 요청에는 고유 식별자가 할당됩니다. 이 식별자는 관련 HTTP 응답의 `request-id` 헤더에 반환됩니다. 요청 식별자를 사용하여 특정 API 요청-응답 트랜잭션의 모든 활동을 총칭하여 참조할 수 있습니다. 예를 들어, 요청 ID를 기반으로 특정 트랜잭션에 대한 모든 이벤트 메시지를 검색할 수 있습니다.

## 동기식 호출과 비동기식 호출

서버가 클라이언트로부터 받은 HTTP 요청을 수행하는 주요 방법은 두 가지입니다.

- 동기식 서버가 요청을 즉시 수행하고 상태 코드 200, 201 또는 204로 응답합니다.
- 비동기 방식입니다. 서버는 요청을 수락하고 상태 코드 202로 응답합니다. 이는 서버가 클라이언트의 요청을 수락하고 요청 처리를 위한 백그라운드 작업을 시작했음을 나타냅니다. 최종 성공 또는 실패 여부는 즉시 확인할 수 없으며, 추가 API 호출을 통해 확인해야 합니다.

## 장기 실행 작업의 완료를 확인합니다

일반적으로 완료하는 데 시간이 오래 걸릴 수 있는 모든 작업은 서버에서 백그라운드 작업을 사용하여 비동기적으로 처리됩니다. Deploy REST API에서는 모든 백그라운드 작업에 Job 객체가 연결되어 있으며, 이 객체는 작업을 추적하고 현재 상태와 같은 정보를 제공합니다. 백그라운드 작업이 생성되면 고유 식별자를 포함한 Job 객체가 HTTP 응답으로 반환됩니다.

Job 객체를 직접 조회하여 관련 API 호출의 성공 또는 실패 여부를 확인할 수 있습니다. 자세한 내용은 _Job 객체를 사용한 비동기 처리_를 참조하십시오.

Job 객체를 사용하는 것 외에도 다음과 같은 방법으로 요청의 성공 또는 실패 여부를 판단할 수 있습니다.

- 이벤트 메시지 특정 API 호출과 관련된 모든 이벤트 메시지는 원래 응답에 포함된 요청 ID를 사용하여 검색할 수 있습니다. 이벤트 메시지에는 일반적으로 성공 또는 실패 여부가 표시되며 오류 상황을 디버깅할 때도 유용하게 사용할 수 있습니다.
- 리소스 상태 여러 리소스는 상태 값을 유지하며, 이 값을 조회하여 요청의 성공 또는 실패 여부를 간접적으로 확인할 수 있습니다.

## 보안

Deploy API는 다음과 같은 보안 기술을 사용합니다.

- 전송 계층 보안 Deploy 서버와 클라이언트 간에 네트워크를 통해 전송되는 모든 트래픽은 TLS를 통해 암호화됩니다. 암호화되지 않은 채널을 통한 HTTP 프로토콜 사용은 지원되지 않습니다. TLS 버전 1.2가 지원됩니다.

- HTTP 인증 기본 인증은 모든 API 트랜잭션에 사용됩니다. base64 문자열로 사용자 이름과 암호를 포함하는 HTTP 헤더가 모든 요청에 추가됩니다.

# ONTAP Select에 대한 요청 및 응답 API 트랜잭션

모든 Deploy API 호출은 Deploy 가상 머신에 대한 HTTP 요청으로 수행되며, 가상 머신은 이에 대한 응답을 클라이언트로 전송합니다. 이러한 요청/응답 쌍을 API 트랜잭션이라고 합니다. Deploy API를 사용하기 전에 요청을 제어하는 데 사용할 수 있는 입력 변수와 응답 출력 내용을 숙지해야 합니다.

## API 요청을 제어하는 입력 변수

HTTP 요청에 설정된 매개변수를 통해 API 호출 처리 방식을 제어할 수 있습니다.

### 요청 헤더

다음을 포함한 여러 헤더를 HTTP 요청에 포함해야 합니다.

- content-type 요청 본문에 JSON이 포함된 경우 이 헤더는 application/json으로 설정해야 합니다.
- accept 응답 본문에 JSON이 포함될 경우, 이 헤더는 application/json으로 설정해야 합니다.
- authorization 기본 인증은 base64 문자열로 인코딩된 사용자 이름과 비밀번호로 설정해야 합니다.

### 요청 본문

요청 본문의 내용은 특정 호출에 따라 다릅니다. HTTP 요청 본문은 다음 중 하나로 구성됩니다.

- 입력 변수(예: 새 클러스터의 이름)가 포함된 JSON 객체
- 비어 있음

### 객체 필터링

GET을 사용하는 API 호출을 실행할 때 모든 속성을 기반으로 반환된 객체를 제한하거나 필터링할 수 있습니다. 예를 들어 일치시킬 정확한 값을 지정할 수 있습니다.

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

정확히 일치하는 항목 외에도, 특정 값 범위에 걸쳐 객체 집합을 반환하는 데 사용할 수 있는 다른 연산자가 있습니다. ONTAP Select는 아래에 표시된 필터링 연산자를 지원합니다.

| 운영자 | 설명 |
| --- | --- |
| = | 같음 |
| < | 미만 |
| > | 보다 큼 |
| ⇐ | 이하 |
| >= | 크거나 같음 |

| 운영자 | 설명 |
| --- | --- |
|  | 또는 |
| ! | 같지 않음 |
| * | 탐욕적 와일드카드 |

쿼리의 일부로 null 키워드 또는 그 부정형(!null)을 사용하면 특정 필드가 설정되었는지 여부에 따라 객체 집합을 반환할 수도 있습니다.

객체 필드 선택

기본적으로 GET을 사용하여 API 호출을 실행하면 객체를 고유하게 식별하는 속성만 반환됩니다. 이러한 최소 필드 집합은 각 객체의 키 역할을 하며 객체 유형에 따라 다릅니다. 다음과 같은 방법으로 fields 쿼리 매개변수를 사용하여 추가 객체 속성을 선택할 수 있습니다.

- 비용이 저렴한 필드 `fields=*`를 지정하여 로컬 서버 메모리에 유지되거나 액세스하는 데 처리량이 거의 필요하지 않은 객체 필드를 검색합니다.
- 비용이 많이 드는 필드 `fields=**`를 지정하면 액세스하는 데 추가 서버 처리가 필요한 필드를 포함하여 모든 객체 필드를 검색합니다.
- 사용자 지정 필드 선택 `fields=FIELDNAME`을 사용하여 원하는 필드를 정확하게 지정할 수 있습니다. 여러 필드를 요청할 경우 값은 공백 없이 쉼표로 구분해야 합니다.

모범 사례로, 항상 원하는 특정 필드를 식별해야 합니다. 필요할 때만 저렴한 필드 또는 비싼 필드 세트를 가져오십시오. 저렴한 필드와 비싼 필드 분류는 NetApp의 내부 성능 분석을 기반으로 결정됩니다. 특정 필드에 대한 분류는 언제든지 변경될 수 있습니다.

출력 세트의 객체를 정렬합니다

리소스 컬렉션의 레코드는 객체에 정의된 기본 순서대로 반환됩니다. 다음과 같이 필드 이름과 정렬 방향을 지정하여 order_by 쿼리 매개변수를 사용하면 순서를 변경할 수 있습니다:

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

예를 들어, type 필드를 내림차순으로 정렬한 다음 id 필드를 오름차순으로 정렬할 수 있습니다.

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

여러 매개변수를 포함할 경우 필드를 쉼표로 구분해야 합니다.

페이지 매김

동일한 유형의 객체 컬렉션에 접근하기 위해 GET 메서드를 사용하는 API 호출 시, 기본적으로 일치하는 모든 객체가 반환됩니다. 필요한 경우, 요청에 max_records 쿼리 매개변수를 사용하여 반환되는 레코드 수를 제한할 수 있습니다. 예를 들면 다음과 같습니다.

```
max_records=20
```

필요한 경우 이 매개변수를 다른 쿼리 매개변수와 결합하여 결과 집합을 좁힐 수 있습니다. 예를 들어 다음은 지정된 시간 이후에 생성된 시스템 이벤트를 최대 10개까지 반환합니다:

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

이벤트(또는 모든 객체 유형)를 페이지별로 보기 위해 여러 번 요청을 보낼 수 있습니다. 각 후속 API 호출은 이전 결과 집합의 최신 이벤트를 기준으로 새로운 시간 값을 사용해야 합니다.

## API 응답 해석

각 API 요청은 클라이언트로 응답을 반환합니다. 응답을 검토하여 요청이 성공했는지 여부를 확인하고 필요에 따라 추가 데이터를 가져올 수 있습니다.

### HTTP 상태 코드

Deploy REST API에서 사용하는 HTTP 상태 코드는 다음과 같습니다.

| 코드 | 의미 | 설명 |
|---|---|---|
| 200 | OK | 새 객체를 생성하지 않는 호출이 성공했음을 나타냅니다. |
| 201 | 생성됨 | 객체가 성공적으로 생성되었습니다. 위치 응답 헤더에 해당 객체의 고유 식별자가 포함되어 있습니다. |
| 202 | 수락됨 | 요청을 처리하기 위해 장시간 실행되는 백그라운드 작업이 시작되었지만, 아직 작업이 완료되지 않았습니다. |
| 400 | 잘못된 요청 | 요청 입력값이 인식되지 않거나 부적절합니다. |
| 403 | 금지됨 | 권한 오류로 인해 액세스가 거부되었습니다. |
| 404 | 찾을 수 없음 | 요청에서 참조한 리소스가 존재하지 않습니다. |
| 405 | 메서드가 허용되지 않습니다 | 요청에 사용된 HTTP 동사는 해당 리소스에서 지원되지 않습니다. |
| 409 | 충돌 | 객체가 이미 존재하므로 객체를 생성하지 못했습니다. |
| 500 | 내부 오류 | 서버에서 일반적인 내부 오류가 발생했습니다. |
| 501 | 구현되지 않음 | URI는 알려져 있지만 요청을 수행할 수 없습니다. |

응답 헤더

Deploy 서버에서 생성되는 HTTP 응답에는 다음과 같은 여러 헤더가 포함됩니다.

- request-id 모든 성공적인 API 요청에는 고유한 요청 식별자가 할당됩니다.
- location 객체가 생성될 때 location 헤더에는 고유한 객체 식별자를 포함하여 새 객체의 전체 URL이 포함됩니다.

응답 본문

API 요청과 관련된 응답의 내용은 객체, 처리 유형 및 요청의 성공 또는 실패에 따라 다릅니다. 응답 본문은 JSON으로 렌더링됩니다.

- 단일 객체 단일 객체는 요청에 따라 필드 집합과 함께 반환될 수 있습니다. 예를 들어, GET을 사용하여 고유 식별자를 통해 클러스터의 선택된 속성을 검색할 수 있습니다.
- 여러 객체 리소스 컬렉션에서 여러 객체를 반환할 수 있습니다. 모든 경우에 일관된 형식이 사용되며, `num_records`레코드 수와 객체 인스턴스 배열을 포함하는 레코드가 표시됩니다. 예를 들어 특정 클러스터에 정의된 모든 노드를 검색할 수 있습니다.
- Job 객체 API 호출이 비동기적으로 처리되는 경우 백그라운드 작업을 고정하는 Job 객체가 반환됩니다. 예를 들어, 클러스터를 배포하는 데 사용되는 POST 요청은 비동기적으로 처리되며 Job 객체를 반환합니다.
- Error 객체 오류가 발생하면 항상 Error 객체가 반환됩니다. 예를 들어, 이미 존재하는 이름으로 클러스터를 생성하려고 하면 오류가 발생합니다.

- 비어 있음 특정 경우에는 데이터가 반환되지 않고 응답 본문이 비어 있습니다. 예를 들어, DELETE를 사용하여 기존 호스트를 삭제한 후에는 응답 본문이 비어 있습니다.

# ONTAP Select용 Job 객체를 사용한 비동기 처리

Deploy API 호출 중 일부, 특히 리소스를 생성하거나 수정하는 호출은 다른 호출보다 완료하는 데 시간이 더 오래 걸릴 수 있습니다. ONTAP Select Deploy는 이러한 장시간 실행되는 요청을 비동기적으로 처리합니다.

## Job 객체를 사용하여 설명된 비동기 요청

비동기적으로 실행되는 API 호출 후 HTTP 응답 코드 202는 요청이 성공적으로 검증되고 수락되었지만 아직 완료되지 않았음을 나타냅니다. 요청은 백그라운드 작업으로 처리되며, 클라이언트에 대한 초기 HTTP 응답 이후에도 계속 실행됩니다. 응답에는 요청을 연결하는 Job 객체와 해당 고유 식별자가 포함됩니다.

비동기적으로 작동하는 API 호출을 확인하려면 ONTAP Select Deploy 온라인 설명서 페이지를 참조하십시오.

## API 요청과 연결된 Job 객체를 쿼리합니다

HTTP 응답에 반환되는 Job 객체에는 여러 속성이 포함되어 있습니다. state 속성을 조회하여 요청이 성공적으로 완료되었는지 확인할 수 있습니다. Job 객체는 다음 상태 중 하나일 수 있습니다.

- 대기 중
- 실행 중
- 성공
- 실패

Job 객체를 폴링하여 작업의 종료 상태(성공 또는 실패)를 감지하는 데 사용할 수 있는 두 가지 기술이 있습니다.

- 표준 폴링 요청 현재 작업 상태가 즉시 반환됩니다
- 장기 폴링 요청 작업 상태는 다음 중 하나가 발생할 때만 반환됩니다.
  - 상태가 폴 요청에 제공된 날짜-시간 값보다 최근에 변경되었습니다
  - 타임아웃 값이 만료되었습니다(1~120초)

표준 폴링과 롱 폴링은 동일한 API 호출을 사용하여 Job 객체를 쿼리합니다. 그러나 롱 폴링 요청에는 두 개의 쿼리 매개변수 `poll_timeout`및 `last_modified`가 포함됩니다.

Deploy 가상 머신의 작업 부하를 줄이려면 항상 롱 폴링을 사용해야 합니다.

## 비동기 요청을 발행하는 일반적인 절차

다음과 같은 상위 수준 절차를 사용하여 비동기 API 호출을 완료할 수 있습니다.

1. 비동기 API 호출을 실행합니다.

2. 요청이 성공적으로 수락되었음을 나타내는 HTTP 응답 202를 수신합니다.
3. 응답 본문에서 Job 객체의 식별자를 추출합니다.
4. 루프 내에서 각 주기마다 다음을 수행합니다.
    a. 롱폴 요청을 사용하여 작업의 현재 상태를 가져옵니다
    b. 작업이 비종료 상태(대기 중, 실행 중)인 경우 루프를 다시 수행합니다.
5. 작업이 최종 상태(성공, 실패)에 도달하면 중지합니다.