시작하십시오
Astra Automation

NetApp
August 11, 2025

목차

시작하십시오

시작하기 전에

아래 단계를 검토하여 Astra Control REST API를 시작할 수 있도록 빠르게 준비할 수 있습니다.

Astra 계정 자격 증명을 가지고 있어야 합니다

Astra 웹 사용자 인터페이스에 로그인하고 API 토큰을 생성하려면 Astra 자격 증명이 필요합니다. Astra Control Center를 사용하면 이러한 자격 증명을 로컬로 관리할 수 있습니다. Astra Control Service를 사용하면 * Auth0 * 서비스를 통해 계정 자격 증명에 액세스할 수 있습니다.

Kubernetes의 기본 개념에 대해 잘 알고 있어야 합니다

Kubernetes의 몇 가지 기본 개념에 익숙해야 합니다. 을 참조하십시오 "기본 Kubernetes 개념" 를 참조하십시오.

REST 개념 및 구현을 검토합니다

반드시 검토하십시오 "핵심 REST 구현" REST 개념과 Astra Control REST API의 설계 방식에 대한 자세한 내용은

자세한 정보를 확인하십시오

에 나와 있는 추가 정보 리소스를 숙지해야 합니다 "추가 리소스".

API 토큰을 가져옵니다

Astra Control REST API를 사용하려면 Astra API 토큰을 얻어야 합니다.

개요

API 토큰은 Astra에 호출자를 식별하며 모든 REST API 호출에 포함되어야 합니다.

- Astra 웹 사용자 인터페이스를 사용하여 API 토큰을 생성해야 합니다.
- 토큰 생성 절차는 Astra 배포 모델 모두에 대해 동일합니다. Astra에 액세스하는 데 사용되는 URL만 다릅니다.
- 토큰과 관련 권한으로 전달된 사용자 ID는 토큰을 만든 사용자에 의해 결정됩니다.
- 토큰은 인가 HTTP 요청 헤더에 포함되어야 합니다.
- 토큰이 생성된 후에는 만료되지 않습니다.
- Astra 웹 사용자 인터페이스에서 토큰을 취소할 수 있습니다.

관련 정보

- "API 토큰을 취소합니다"

Astra API 토큰을 생성합니다

다음 단계에서는 Astra API 토큰을 생성하는 방법을 설명합니다.

시작하기 전에

Astra 계정에 대한 자격 증명이 필요합니다.

이 작업에 대해

이 작업은 Astra 웹 인터페이스에서 API 토큰을 생성합니다. 또한 API 호출 시 필요한 계정 ID도 검색해야 합니다.

단계

1. 다음과 같이 계정 자격 증명을 사용하여 Astra에 로그인합니다.
   - Astra 제어 서비스: "https://astra.netapp.io"
   - Astra Control Center: 설치 중에 설정한 로컬 환경에 대한 URL을 사용합니다
2. 페이지 오른쪽 상단의 그림 아이콘을 클릭하고 * API access * 를 선택합니다.
3. 페이지에서 * API 토큰 생성 * 을 클릭한 다음 팝업 창에서 * API 토큰 생성 * 을 클릭합니다.
4. 아이콘을 클릭하여 토큰 문자열을 클립보드에 복사하고 편집기에 저장합니다.
5. 동일한 페이지에서 사용할 수 있는 계정 ID를 복사하여 저장합니다.

작업을 마친 후

curl 또는 프로그래밍 언어를 통해 Astra Control REST API에 액세스하는 경우 HTTP에 API 베어러 토큰을 포함해야 합니다 `Authorization` 요청 헤더.

첫 번째 API 호출입니다

워크스테이션 CLI에서 간단한 curl 명령을 실행하여 Astra Control REST API를 사용하여 사용 가능 여부를 확인할 수 있습니다.

시작하기 전에

cURL 유틸리티는 로컬 워크스테이션에서 사용할 수 있어야 합니다. 또한 API 토큰과 관련 계정 식별자가 있어야 합니다. 자세한 내용은 을 "API 토큰을 가져옵니다" 참조하십시오.

컬의 예

다음 curl 명령은 Astra 사용자 목록을 조회한다. 표시된 대로 적절한 $ACCOUNT_ID 및 $API_TOKEN을 제공하십시오.

```
curl --request GET  \
--location "https://astra.netapp.io/accounts/$ACCOUNT_ID/core/v1/users" \
--include \
--header "Content-Type: application/json" \
--header "Accept: */*" \
--header "Authorization: Bearer $API_TOKEN"
```

JSON 출력 예

```
{
  "items": [
    [
    "David",
    "Anderson",
    "844ec6234-11e0-49ea-8434-a992a6270ec1"
    ],
    [
      "Jane",
      "Cohen",
      "2a3e227c-fda7-4145-a86c-ed9aa0183a6c"
    ]
  ],
  "metadata": {}
}
```

기본 Kubernetes 개념

Astra REST API를 사용할 때는 몇 가지 Kubernetes 개념을 이해해야 합니다.

오브젝트

Kubernetes 환경 내에 유지되는 객체는 클러스터 구성을 나타내는 영구 엔터티입니다. 이러한 오브젝트는 클러스터 워크로드를 포함한 시스템의 상태를 종합적으로 설명합니다.

네임스페이스

네임스페이스는 단일 클러스터 내에서 리소스를 격리하는 기술을 제공합니다. 이 조직 구조는 작업, 사용자 및 자원의 유형을 나눌 때 유용합니다. _namespace scope_의 개체는 네임스페이스 내에서 고유해야 하지만 _cluster scope_를 가진 개체는 전체 클러스터에서 고유해야 합니다.

라벨

라벨은 Kubernetes 오브젝트와 연결될 수 있습니다. 이들은 키 값 쌍을 사용하여 특성을 설명하고, 조직에 유용할 수 있지만 핵심 Kubernetes 운영 이외의 클러스터에 임의 조직을 적용할 수 있습니다.

워크플로우 사용을 준비하십시오

실제 배포와 함께 사용하기 전에 Astra 워크플로의 구성 및 형식을 숙지해야 합니다.

소개

Workflow _ 은(는) 특정 관리 작업 또는 목표를 달성하는 데 필요한 하나 이상의 단계 시퀀스입니다. Astra Control 워크플로우의 각 단계는 다음 중 하나입니다.

- REST API 호출(curl 및 JSON 예 등의 세부 정보 포함)
- 다른 Astra 워크플로 호출
- 기타 관련 작업(예: 필요한 설계 결정)

워크플로에는 각 작업을 수행하는 데 필요한 핵심 단계와 매개 변수가 포함됩니다. 또한 자동화 환경을 사용자 지정할 수 있는 출발점을 제공합니다.

> 워크플로는 한 단계만 포함할 수 있습니다. 이러한 단일 단계 워크플로는 여러 단계가 포함된 워크플로와 약간 다르게 서식이 지정됩니다. 예를 들어, 명시적인 단계 이름이 제거됩니다. 작업 또는 작업은 워크플로 제목에 따라 명확해야 합니다.

워크플로 범주

배포 모델에 따라 두 가지 범주의 Astra 워크플로를 사용할 수 있습니다. Astra Control Center를 사용하는 경우 인프라 워크플로로 시작한 다음 관리 워크플로로 넘어가야 합니다. Astra Control Service를 사용하는 경우 일반적으로 관리 워크플로로 직접 이동할 수 있습니다.

> 워크플로우의 curl 샘플에서는 Astra Control Service의 URL을 사용합니다. 사내 Astra Control Center를 사용하는 경우 해당 환경에 맞게 URL을 변경해야 합니다.

인프라 워크플로우

이러한 워크플로우에서는 자격 증명, 버킷 및 스토리지 백엔드를 포함한 Astra 인프라를 처리합니다. Astra Control Center와 함께 필요하지만 대부분의 경우 Astra Control Service와 함께 사용할 수 있습니다. 워크플로우는 Astra 관리 클러스터를 설정하고 유지하는 데 필요한 작업에 중점을 둡니다.

관리 워크플로

관리형 클러스터를 설정한 후에는 이러한 워크플로우를 사용할 수 있습니다. 관리 워크플로우에서는 애플리케이션 보호 및 앱 백업, 복원, 클론 복제와 같은 지원 작업에 주력합니다.

공통 입력 매개변수

아래에 설명된 입력 매개 변수는 REST API 호출을 나타내는 데 사용되는 모든 curl 샘플에 공통으로 사용됩니다.

> 이러한 입력 매개 변수는 보편적으로 필요하기 때문에 개별 워크플로에서 더 이상 설명되지 않습니다. 특정 curl 예제에 추가 입력 매개 변수가 사용되는 경우 * 추가 입력 매개 변수 * 절에 설명되어 있습니다.

경로 매개 변수

모든 REST API 호출에 사용되는 엔드포인트 경로에는 다음 매개 변수가 포함됩니다. 도 참조하십시오 "URL 형식" 를 참조하십시오.

계정 **ID**입니다

API 작업이 실행되는 Astra 계정을 식별하는 UIDv4 값이다. 을 참조하십시오 "API 토큰을 가져옵니다" 계정 ID를 찾는 방법에 대한 자세한 내용은 를 참조하십시오.

요청 헤더

REST API 호출에 따라 몇 가지 요청 헤더를 포함해야 할 수도 있습니다.

권한 부여

워크플로의 모든 API 호출에는 사용자를 식별하기 위한 API 토큰이 필요합니다. '권한 부여' 요청 헤더에 토큰을 포함해야 합니다. 을 참조하십시오 "API 토큰을 가져옵니다" API 토큰 생성에 대한 자세한 내용은 를 참조하십시오.

콘텐츠 유형

HTTP POST 및 PUT 요청을 사용하여 요청 본문에 JSON이 포함된 경우 Astra 리소스를 기반으로 미디어 유형을 선언해야 합니다. 예를 들어, 관리 대상 애플리케이션에 대한 스냅샷을 생성할 때 "Content-Type:application/astra-appSnap+json" 머리글을 포함할 수 있습니다.

수락

Astra 리소스를 기반으로 응답에서 예상하는 콘텐츠의 특정 미디어 유형을 선언할 수 있습니다. 예를 들어, 관리 대상 애플리케이션의 백업을 나열할 때 "Accept:application/astra-appBackup+json" 머리글을 포함할 수 있습니다. 그러나 간단히 하기 위해 워크플로의 curl 샘플에는 모든 미디어 유형이 적용됩니다.

토큰 및 식별자 표시

curl 예제에 사용된 API 토큰과 기타 ID 값은 식별 가능한 의미 없이 불투명합니다. 따라서 샘플의 가독성을 향상시키기 위해 실제 토큰 및 ID 값은 사용되지 않습니다. 대신, 다음과 같은 몇 가지 이점을 제공하는 더 작은 예약 키워드가 사용됩니다.

- CURL 및 JSON 샘플은 보다 명확하고 이해하기 쉽습니다.
- 모든 키워드는 대괄호와 대문자를 사용하여 동일한 형식을 사용하기 때문에 삽입하거나 추출할 위치와 내용을 빠르게 식별할 수 있습니다.
- 원래 매개 변수를 복사하여 실제 배포에서 사용할 수 없으므로 값이 손실되지 않습니다.

변수는 Bash 셸 환경에서 사용할 수 있도록 형식이 지정됩니다. 각 변수는 달러 기호로 시작하고 필요에 따라 큰따옴표로 묶습니다. 그러면 Bash에서 인식할 수 있습니다. 대문자는 이름에 일관되게 사용됩니다.

다음은 curl 예제에 사용되는 몇 가지 일반적인 예약된 키워드입니다. 이 목록은 전체 목록이 아니며 필요에 따라 추가 키워드가 사용됩니다. 그들의 의미는 상황에 따라 분명해야 합니다.

키워드	유형	설명
$ACCOUNT_ID입니다	경로	API 작업이 실행되는 계정을 식별하는 UIDv4 값입니다.
$api_token입니다	머리글	호출자를 식별하고 승인하는 베어러 토큰.
$APP_ID입니다	경로	API 호출에 대한 응용 프로그램을 식별하는 UIDv4 값입니다.

Bash와 함께 예제 사용

워크플로 컬 예제를 직접 사용하는 경우 해당 변수에 포함된 변수를 환경에 적합한 값으로 업데이트해야 합니다. 예제를 수동으로 편집하거나 아래에 설명된 대로 Bash 셸을 사용하여 대신 사용할 수 있습니다.

Bash를 사용하면 curl 명령마다 한 번 설정하는 대신 셸 세션에서 한 번 변수 값을 설정할 수 있다는 이점이 있습니다.

단계

1. Linux 또는 유사한 운영 체제와 함께 제공되는 Bash 셸을 엽니다.
2. 실행할 컬링 예제에 포함된 변수 값을 설정합니다. 예를 들면 다음과 같습니다.

```
$API_TOKEN=SGgpXHeco6M8PLxzIlgbztA4k3_eX4UCa842hOXHBFA=
```

3. 워크플로 페이지에서 컬링 예제를 복사하여 셸 터미널에 붙여 넣습니다.
4. 다음 작업을 수행하려면 * ENTER * 를 누르십시오.
   a. 설정한 변수 값으로 대체합니다.
   b. curl 명령을 실행합니다.

온라인 **API** 참조입니다

Astra API 참조 문서에 액세스합니다

HTTP 메서드, 입력 매개 변수 및 응답을 포함하여 Astra Control REST API 호출의 세부 정보에 액세스할 수 있습니다. 이 전체 참조는 REST API를 사용하여 자동화 응용 프로그램을 개발할 때 유용합니다.

시작하기 전에

배포를 위해 Astra 웹 사용자 인터페이스에 로그인하려면 자격 증명이 필요합니다. 참조 문서에 액세스하는 절차는 Astra Control Service 및 Astra Control Center와 동일합니다. URL만 다릅니다. 참조 문서를 액세스하고 보는 데 API 토큰이 필요하지 않습니다.

단계

1. 다음과 같이 계정 자격 증명을 사용하여 Astra에 로그인합니다.
   - Astra 제어 서비스: "https://astra.netapp.io"
   - Astra Control Center: 설치 중에 설정한 로컬 환경에 대한 URL을 사용합니다
2. 페이지 오른쪽 상단의 그림 아이콘을 클릭하고 * API access * 를 선택합니다.
3. 페이지 맨 위에서 * API Documentation * 아래에 표시된 URL을 클릭합니다.

결과

Swagger 페이지가 새 창 또는 탭에서 시작됩니다. URL에는 로그인한 계정의 계정 ID가 포함되어 있습니다.

다음 단계

Swagger 페이지에서 API 호출을 선택적으로 실행할 수 있습니다. 을 참조하십시오 "Astra REST API 호출 전송" 를 참조하십시오.

Astra REST API 호출 전송

API 참조 문서 페이지에서 Astra Control REST API 호출을 실행할 수 있습니다.

시작하기 전에

Astra에 로그인하여 API 참조 페이지에 액세스해야 합니다. 을 참조하십시오 "Astra API 참조 문서에 액세스합니다" 를

참조하십시오. REST API를 사용하려면 토큰도 필요합니다. 을 참조하십시오 "API 토큰을 가져옵니다" API 토큰 생성에 대한 자세한 내용은 를 참조하십시오.

단계

1. API 참조 페이지 상단에서 * authorize * 를 클릭합니다.
2. 팝업 창의 필드에 API 토큰 값을 복사하여 붙여 넣고 * authorize * 를 클릭한 다음 * Close * 를 클릭합니다.
3. 페이지를 아래로 스크롤하여 원하는 API 호출을 엽니다.
4. 마우스 오른쪽 단추로 * 시험 사용 * 을 클릭합니다.
5. 동일한 API 호출 내에서 아래로 스크롤합니다. 필요한 매개 변수 값을 입력하고 * Execute * 를 클릭하여 호출을 실행합니다.

결과

API 호출이 실행되고 HTTP 상태 코드가 표시됩니다.