

Catalyst Center-APIs verstehen und verwenden

Inhalt

[Einleitung](#)

[Hintergrundinformationen](#)

[Voraussetzungen](#)

[Anforderungen](#)

[Verwendete Komponenten](#)

[Hintergrundinformationen](#)

[Einführung in HTTP](#)

[Die HTTP-Anforderung](#)

[Die HTTP-Antwort](#)

[HTTP-Methoden \(Verben\)](#)

[HTTP-Statuscodes](#)

[REST - Überblick](#)

[Catalyst Center- und REST-APIs](#)

[Authentifizierungstoken mit Bruno abrufen](#)

[1. API-Endpunkt identifizieren](#)

[2. Bruno für die Authentifizierung konfigurieren](#)

[3. Setzen Sie eine globale Header in Bruno](#)

[Beispiel 1: Erstellen einer Site-Etage](#)

[Schritt 1: Übergeordnete ID abrufen](#)

[Phase 2: Erstellen Sie den Boden](#)

[Beispiel 2: Schnittstelleninformationen abrufen](#)

[Schritt 1: Identifizieren der Geräte-ID](#)

[Phase 2: Parametrisierte Anforderung in Bruno erstellen](#)

[Schlussfolgerung](#)

Einleitung

In diesem Dokument wird beschrieben, wie Sie mithilfe eines API-Clients und der integrierten Entwicklungstools der Plattform auf CaTC Northbound (Intent)-APIs (APIs) zugreifen und diese nutzen.

Hintergrundinformationen

Es beschreibt grundlegende HTTP/REST-Konzepte und bietet praktische Beispiele für Automatisierung und Tests.

Voraussetzungen

Anforderungen

Cisco empfiehlt, sich mit folgenden Themen vertraut zu machen:

- HTTP/1.1-Protokolle und Statuscodes.
- Architekturstil REST.
- Grundlegendes Verständnis von Cisco Catalyst Center (CaTC)
- Grundlegendes Verständnis von Bruno (oder ähnlichen API-Clients wie Postman).

Verwendete Komponenten

Die Informationen in diesem Dokument basieren auf folgenden Hardware- und Software-Versionen:

- Cisco Catalyst Center Versionen 2.3.7.x und 3.x
- Bruno Version 4.1.0

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die möglichen Auswirkungen aller Befehle kennen.

Hintergrundinformationen

Catalyst Center Northbound-APIs (Intent APIs) ermöglichen die Interaktion mit dem Controller außerhalb der grafischen Benutzeroberfläche. Dies ermöglicht die Automatisierung maßgeschneiderter Workflows, die Integration mit Drittanbieter-Tools und ein effizientes skalierbares Netzwerkmanagement.

Einführung in HTTP

Das Hypertext Transfer Protocol (HTTP) bildet die Grundlage für die Datenkommunikation im Internet. Es entspricht einem Client-Server-Modell, bei dem der Client eine Anforderung sendet und der Server eine Antwort zurückgibt.

Die HTTP-Anforderung

Eine HTTP-Anfrage besteht aus vier Teilen:

1. Anforderungszeile: Gibt die Methode (einschließlich GET und POST), den URI und die Protokollversion (z. B. GET /api/v1/nodes HTTP/1.1) an.
2. Header: Geben Sie Metadaten an, z. B. Authentifizierungsdaten und Inhaltstyp.
3. Leere Zeile: Trennt Header vom Textkörper.
4. Nachrichtentext (optional): Enthält an den Server gesendete Daten (häufig in POST/PUT/PATCH).

Die HTTP-Antwort

Eine HTTP-Antwort spiegelt die Anforderungsstruktur:

1. Statuszeile: Enthält den Statuscode (z. B. 200 OK).
2. Header: Details über den Server und die Antwortdaten.
3. Nachrichtentext: Angeforderte Daten (in der Regel im JSON-Format für Catalyst Center)

HTTP-Methoden (Verben)

- ABRUFEN: Ruft Daten von einer Ressource ab.
- POST: Erstellt eine neue Ressource.
- PUT: Ersetzt eine vorhandene Ressource.
- PATCH: Wendet partielle Änderungen an einer Ressource an.
- LÖSCHEN: Entfernt eine Ressource.

HTTP-Statuscodes

HTTP-Statuscodes sind standardisierte numerische Codes, die von einem Server als Teil einer HTTP-Antwort zurückgegeben werden. Sie geben das allgemeine Ergebnis einer Clientanforderung an und helfen zu verstehen, ob die Anforderung erfolgreich war, zusätzliche Maßnahmen erfordert oder ein Fehler aufgetreten ist.

Die Codes sind entsprechend ihrer ersten Ziffer in fünf Hauptkategorien eingeteilt.

- 1xx-Codes sind informativ und weisen darauf hin, dass die Anfrage empfangen wurde und die Verarbeitung fortgesetzt wird.
- 2xx Codes geben an, dass die Anfrage erfolgreich war, d. h. der Server hat die Anfrage erfolgreich empfangen, verstanden und verarbeitet.
- 3xx-Codes werden für die Umleitung verwendet und weisen darauf hin, dass zusätzliche Maßnahmen erforderlich sein können, um die Anfrage abzuschließen.
- 4xx-Codes stehen für Client-Fehler, d. h. es liegt ein Problem mit der vom Client gesendeten Anforderung vor.
- 5xx-Codes stellen Serverfehler dar, d. h., der Server hat beim Verarbeiten einer Anforderung ein Problem festgestellt.

REST - Überblick

REST (Representational State Transfer) ist eine Architektur für die Entwicklung von Netzwerkanwendungen. RESTful-APIs verwenden HTTP als Transportmechanismus und verwenden aufgrund seines geringen Gewichts in der Regel JSON für den Datenaustausch.

Wenn ein Client mit einer REST-API interagiert, gibt der Server eine Darstellung des aktuellen Zustands einer Ressource zurück. Diese Darstellung wird als Teil der Antwort an den Client übertragen und kann in einem anderen Format ausgedrückt werden.

REST ist kein Protokoll. Stattdessen ist es ein hierarchischer Stil, der eine Reihe von Prinzipien und Einschränkungen für die Erstellung verteilter Systeme und APIs definiert. In modernen APIs ist JSON eines der am häufigsten verwendeten Formate, da es leicht und einfach zu verarbeiten ist.

Eine typische REST - Interaktion hängt von einem kundenbasierten Modell ab (dies wurde bereits im Artikel erwähnt). Der Client sendet eine HTTP-Anforderung mithilfe von Methoden wie GET, POST, PUT, PATCH oder DELETE. Der Serverprozess gibt eine HTTP-Antwort zurück, die die angeforderten Informationen oder das Ergebnis des Vorgangs enthält.

RESTful-APIs verwenden in der Regel HTTP als Kommunikationsmechanismus für die Ausführung von Vorgängen auf Ressourcen. In diesem Modell identifiziert der Client eine Ressource/einen Endpunkt und verwendet dann die HTTP-Methode, um die Aktion anzugeben, die für diese Ressource ausgeführt werden soll. Eine REST-API kann mithilfe mehrerer Hauptelemente beschrieben werden. Zunächst wird URI verwendet, um die Ressource oder den Endpunkt zu identifizieren, auf die bzw. den der Client zugreifen möchte. Der URI stellt den Speicherort oder den Bezeichner dieser Ressource innerhalb der API bereit. Zweitens verwendet der Client standardmäßige HTTP-Methoden wie GET, POST, PUT, PATCH und DELETE, um den

Anforderungsvorgang zu definieren. GET kann beispielsweise zum Abrufen einer Ressource verwendet werden, POST zum Erstellen einer Ressource und DELETE zum vollständigen Entfernen einer Ressource. Schließlich wird die Ressource über eine Darstellung ausgetauscht, die durch ihren Medientyp definiert ist. Bei dieser Darstellung handelt es sich um die Form, in der die Ressourcendaten zwischen dem Client und dem Server übertragen werden, z. B. JSON oder XML.

Catalyst Center- und REST-APIs

CaTC vereinfacht den Betrieb des Campus-Netzwerks durch die Zentralisierung von Automatisierung, Konfiguration, Richtlinienmanagement, Überwachung und Analysen. Sie ermöglichen Netzwerkadministratoren eine effizientere Bereitstellung und Verwaltung von Geräten, die Einhaltung konsistenter Richtlinien, die Überwachung des Netzwerkverhaltens und die Verbesserung der allgemeinen Transparenz des Betriebs. Auf diese Funktionen kann über die grafische Benutzeroberfläche (GUI) sowie über APIs und externe Integrationen zugegriffen werden.

Die CaTC-Plattform stellt außerdem APIs bereit, mit denen benutzerdefinierte Integrationen erstellt und Netzwerkabläufe automatisiert werden können. Durch die Kombination von CaTC APIs mit vorhandenen Lösungen von Cisco oder Anwendungen von Drittanbietern können Unternehmen die Plattformfunktionalität erweitern und maßgeschneiderte Workflows für spezifische Betriebs- oder Geschäftsanforderungen erstellen.

Nach der Erläuterung der grundlegenden Konzepte konzentriert sich dieses Dokument auf Northbound-APIs (auch als Intent APIs bezeichnet) in Cisco Catalyst Center. Über diese APIs werden bestimmte Catalyst Center-Funktionen über eine richtlinienbasierte Abstraktion von Geschäftszielen bereitgestellt, sodass sich Administratoren und Entwickler auf das gewünschte Ergebnis konzentrieren können und nicht auf die einzelnen Konfigurationsschritte, die erforderlich sind, um dieses zu erreichen.

Catalyst Center Intent APIs sind RESTful APIs, die das JSON-Datenformat für die Interaktion mit der Plattform verwenden. Sie können verwendet werden, um Netzwerkinformationen zu erkennen, Betriebsdaten abzurufen und unterstützte Kontroll- und Automatisierungsaufgaben im gesamten Netzwerk auszuführen.

Authentifizierungstoken mit Bruno abrufen

Bevor Sie API-Aufrufe durchführen, müssen Sie sich authentifizieren, um ein Token zu erhalten.

1. API-Endpunkt identifizieren

1. Melden Sie sich bei Catalyst Center an.
2. Navigieren Sie zu Menü > Plattform > Developer Toolkit.
3. Wählen Sie im Menü links Authentication > Authentication API aus.

The screenshot shows the Catalyst Center Developer Toolkit interface. On the left, a sidebar menu lists various categories, with 'Authentication' highlighted and expanded. Under 'Authentication', the 'Authentication API' is selected. The main content area displays a table of API endpoints. The table has columns for Method, Name, Description, URL, and Actions. The 'Authentication API' row is highlighted with a red box. The URL for this API is /dna/system/api/v1/auth/token.

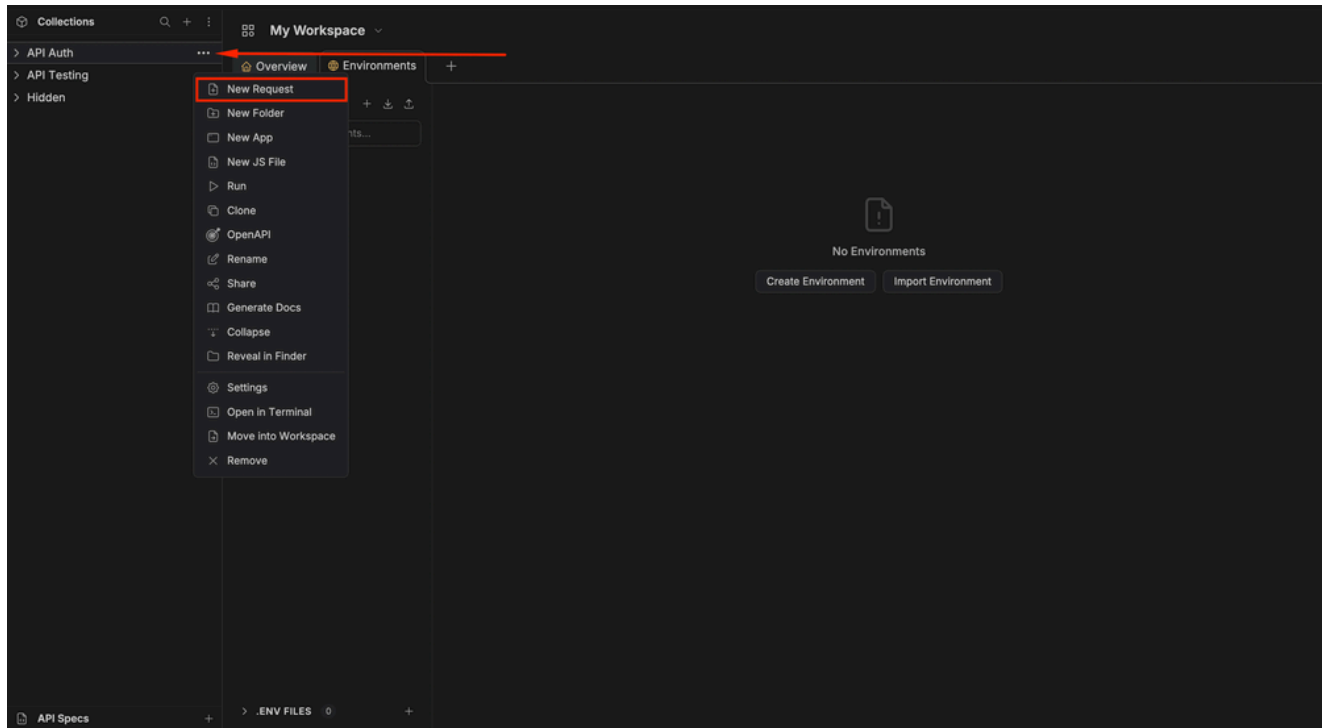
Method	Name	Description	URL	Actions
POST	importCertificate	This API enables a user to import a PEM certificate and its key for the controller and/or disaster recovery.	/dna/intent/api/v1/certificate	...
POST	importCertificateP12	This API enables a user to import a PKCS12 certificate bundle for the controller and/or disaster recovery.	/dna/intent/api/v1/certificate-p12	...
POST	Authentication API	API to obtain an access token, which remains valid for 1 hour. The token obtained using this API is required to be set as value to the X-Auth-Token HTTP...	/dna/system/api/v1/auth/token	...

Catalyst Center Entwickler-Toolkit

4. Kopieren Sie die URL (normalerweise POST /dna/system/api/v1/auth/token).

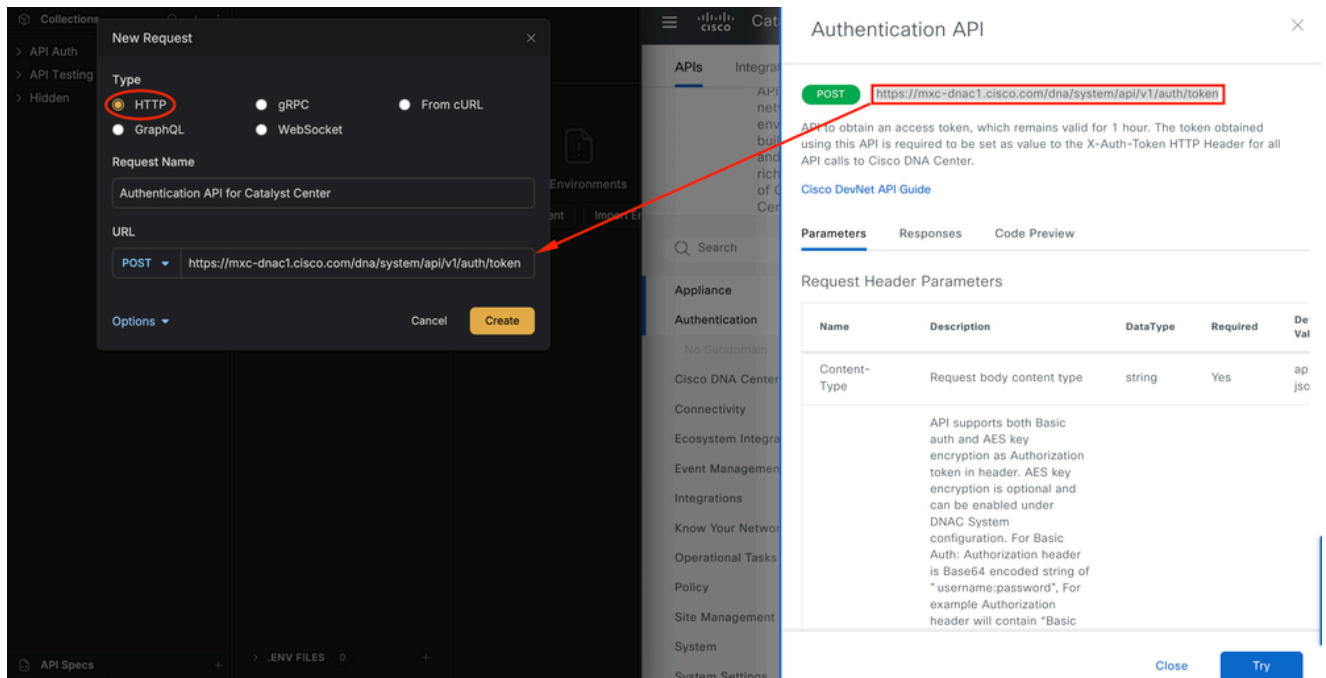
2. Bruno für die Authentifizierung konfigurieren

1. Erstellen Sie in Bruno eine neue Anforderung.
2. Legen Sie den Typ auf HTTP und die Methode auf POST fest..



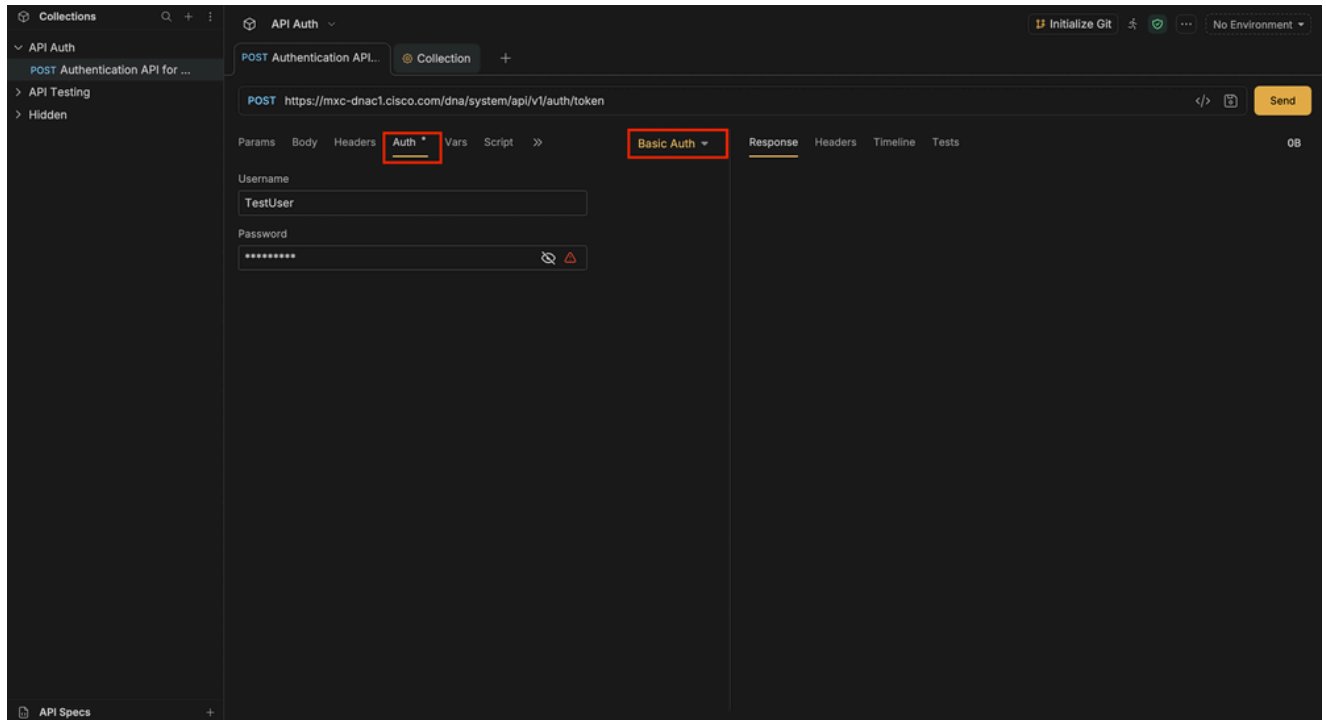
Erstellen einer neuen Anfrage in Bruno

3. Geben Sie die URL für die Authentifizierungs-API ein.



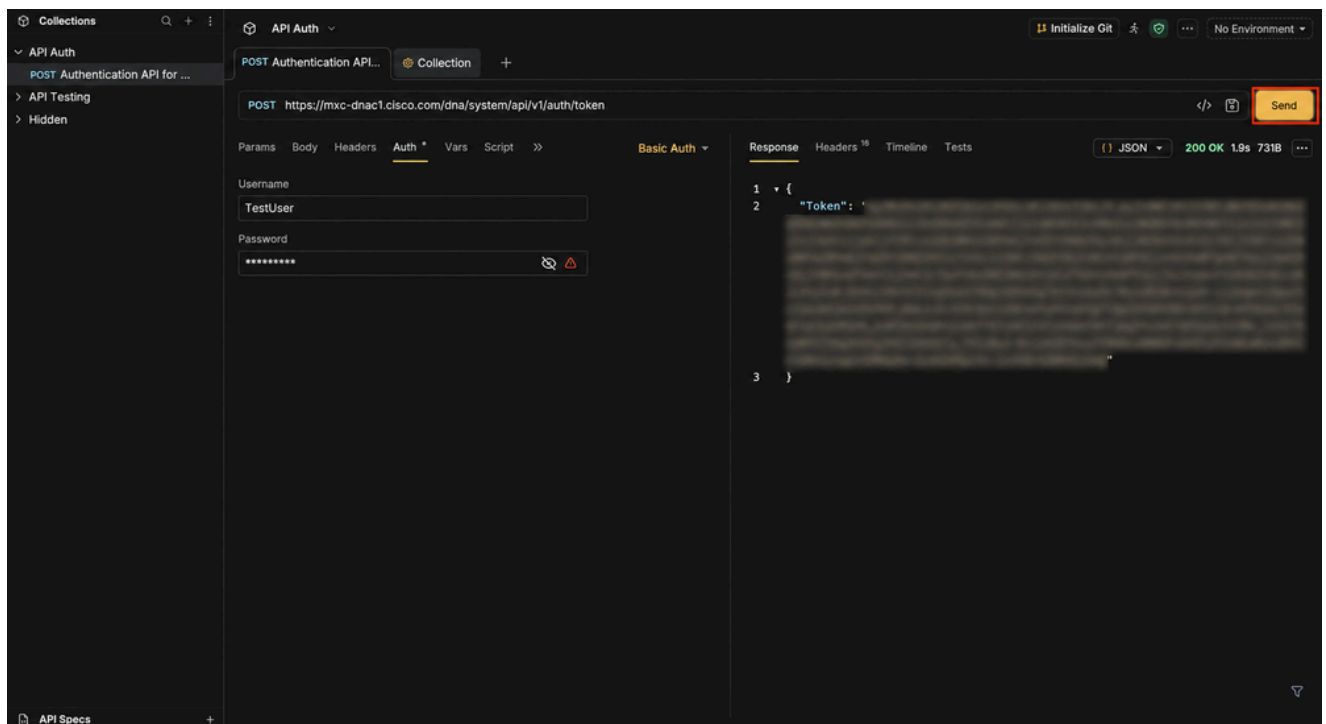
Erstellen einer neuen Anfrage in Bruno

4. Navigieren Sie zur Registerkarte Auth, und wählen Sie Einfache Auth.



Standardauthentifizierung für die Authentifizierungs-API

5. Geben Sie Ihre Catalyst Center SUPER ADMIN-Anmeldeinformationen ein.
6. Klicken Sie auf Senden.



Ausführen der Authentifizierungs-API

Der Server gibt einen 200 OK-Status und eine JSON-Payload zurück, die das Token enthält.

Alternative (Terminal): You can also generate the token by base64-encoding your `username:password` and

```
[ ~ % echo -n 'TestUser:Test123456789' | base64
VGZzdFVzZXI6VGZzdDEyMyEhIQ==
```

Abrufen der base64-Codierung für Benutzername und Kennwort

Catalyst Center Authentication API

Try 'Authentication API'

Method: POST Public URL :https://mxc-dnac1.cisco.com/dna/system/api/v1/auth/token

PARAMETERS

HEADERS

Content-Type* application/json

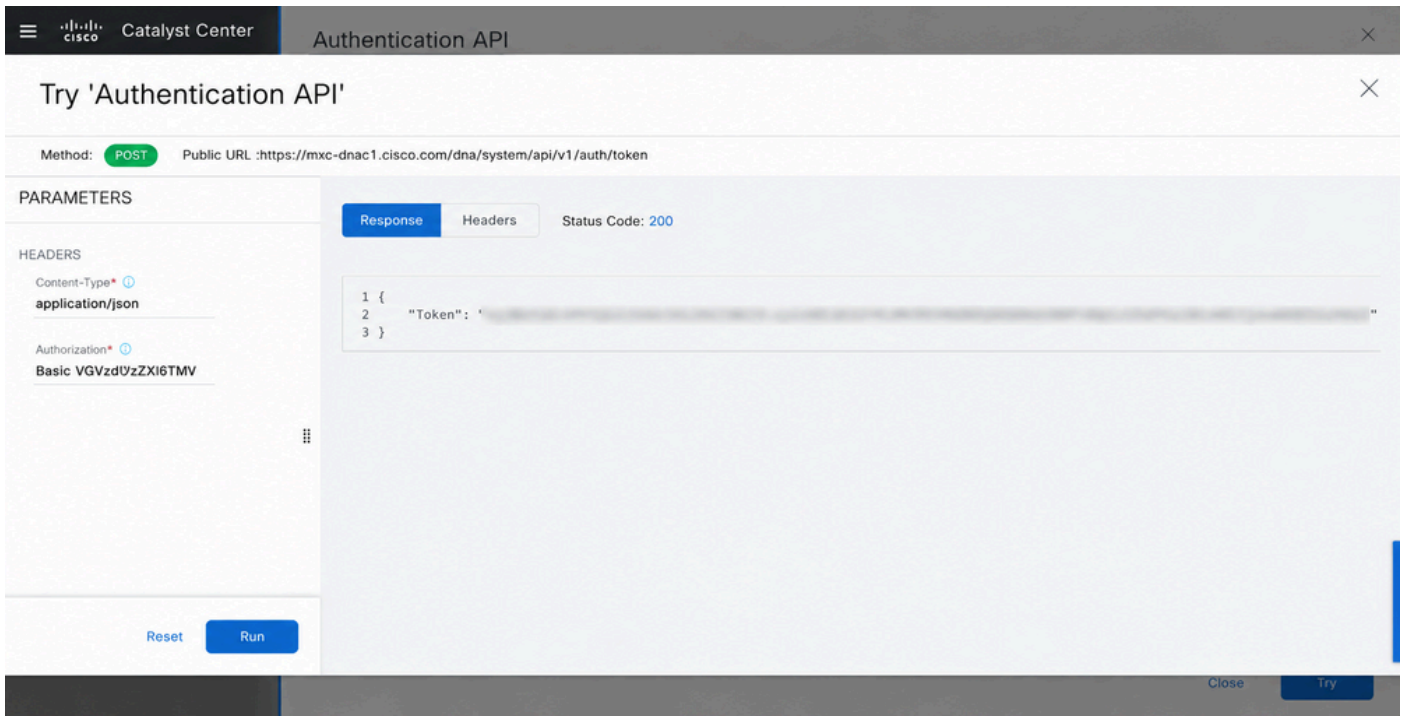
Authorization* Basic VGZzdFVzZXI6VGZzdDEyMyEhIQ==

Reset Run

Enter Parameters

First enter in some necessary parameters on the left and then click Run.

Eingeben von Basic Auth mit verschlüsseltem base64-Benutzernamen und -Kennwort



Authentifizierungstoken wird von der Catalyst Center-GUI abgerufen.

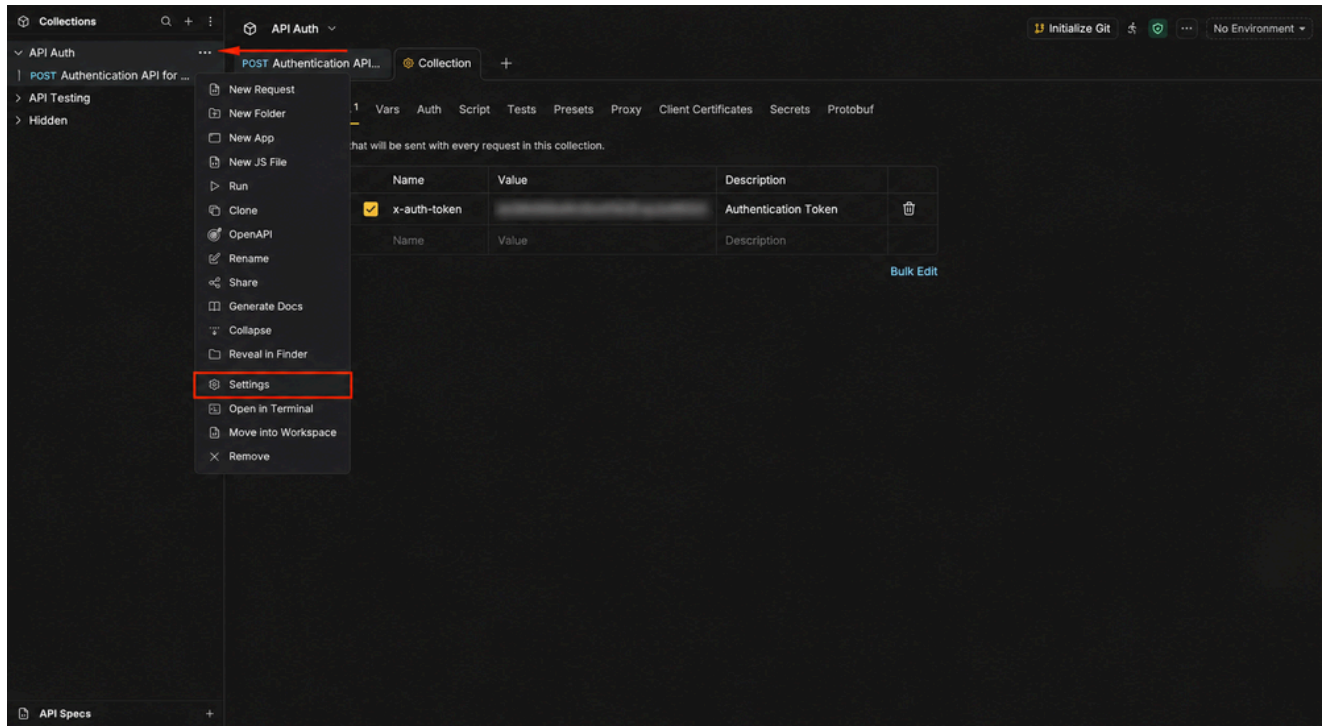
An diesem Punkt befindet sich das Authentifizierungstoken, entweder auf der Bruno-Antwort oder auf der Antwort des Catalyst Center Developer Toolkit.

Fügen Sie nun das Token als Header in den Sammlungseinstellungen in Bruno hinzu. Dies erleichtert jede andere erstellte Anforderung innerhalb der Sammlung, das Token zu verwenden, sodass Sie das Token nicht manuell als Header für jede Anforderung eingeben müssen.

3. Setzen Sie eine globale Header in Bruno

Um eine erneute Authentifizierung für jede Anforderung zu vermeiden, legen Sie das Token als Sammlungsebenenkopf fest:

1. Klicken Sie mit der rechten Maustaste auf Ihre Sammlung, und wählen Sie Einstellungen aus.

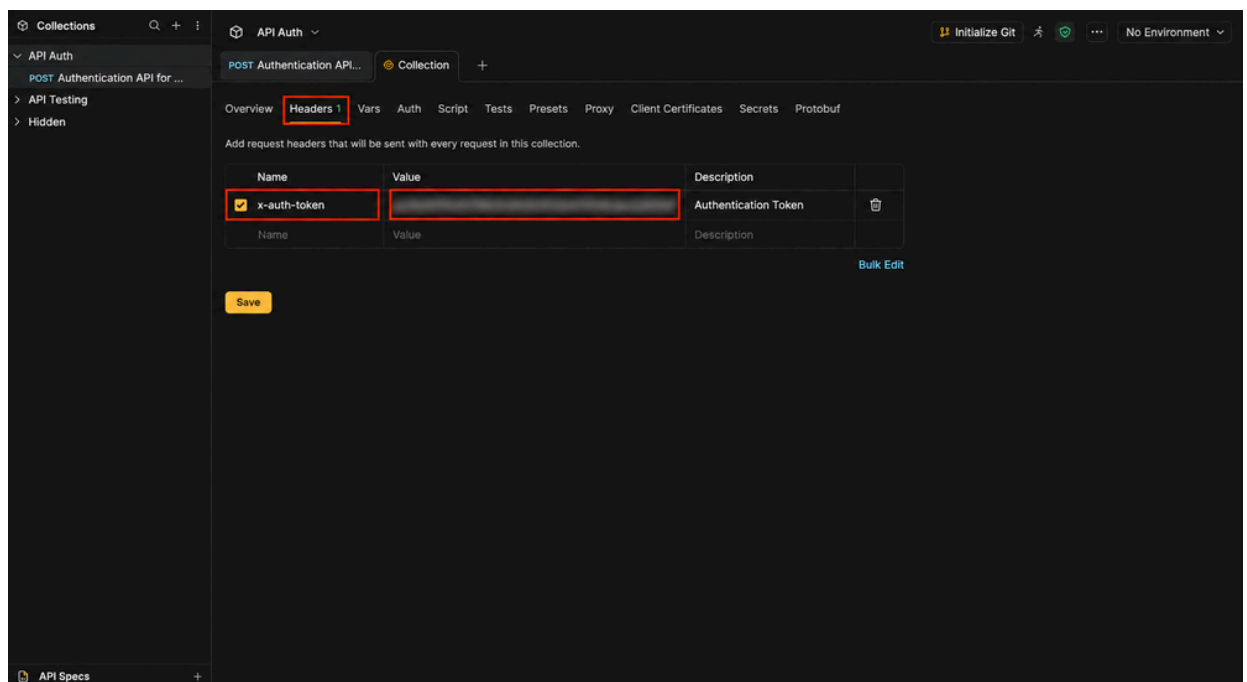


Sammlungseinstellungen

2. Wechseln Sie zur Registerkarte Headers.

3. Neuen Header hinzufügen:

- Name: x-auth-token
- Wert: [Ihr Token hier einfügen]



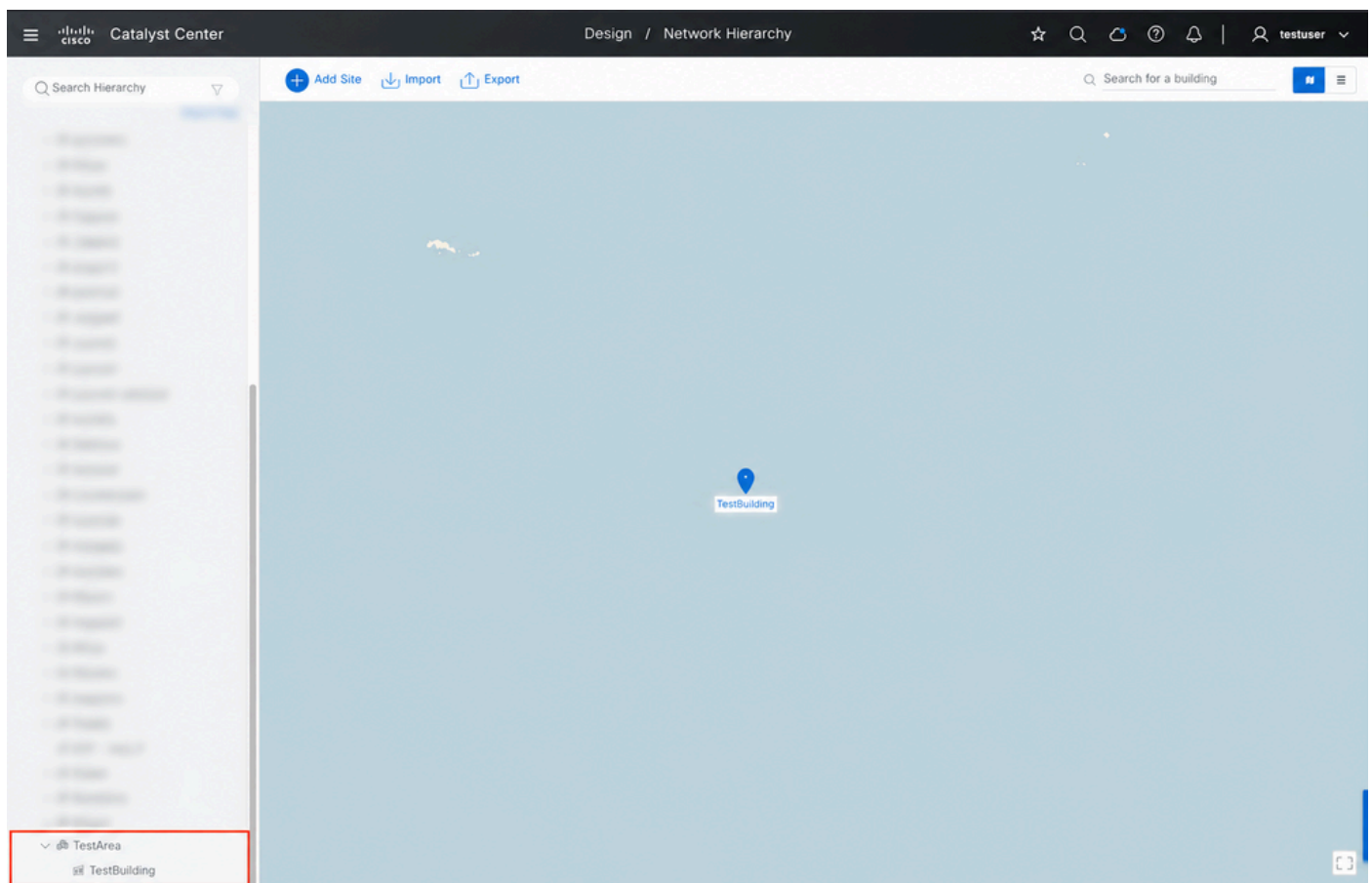
Eingabe von globalen Headern für die API Collection in Bruno.

4. Klicken Sie auf Speichern.

An diesem Punkt haben Sie das Authentifizierungstoken generiert, um andere Catalyst Center API-Anforderungen zu erfüllen.

Beispiel 1: Erstellen einer Site-Etage

In diesem Beispiel wird veranschaulicht, wie eine Ebene innerhalb der Netzwerkhierarchie erstellt wird (Global > TestArea > TestBuilding).

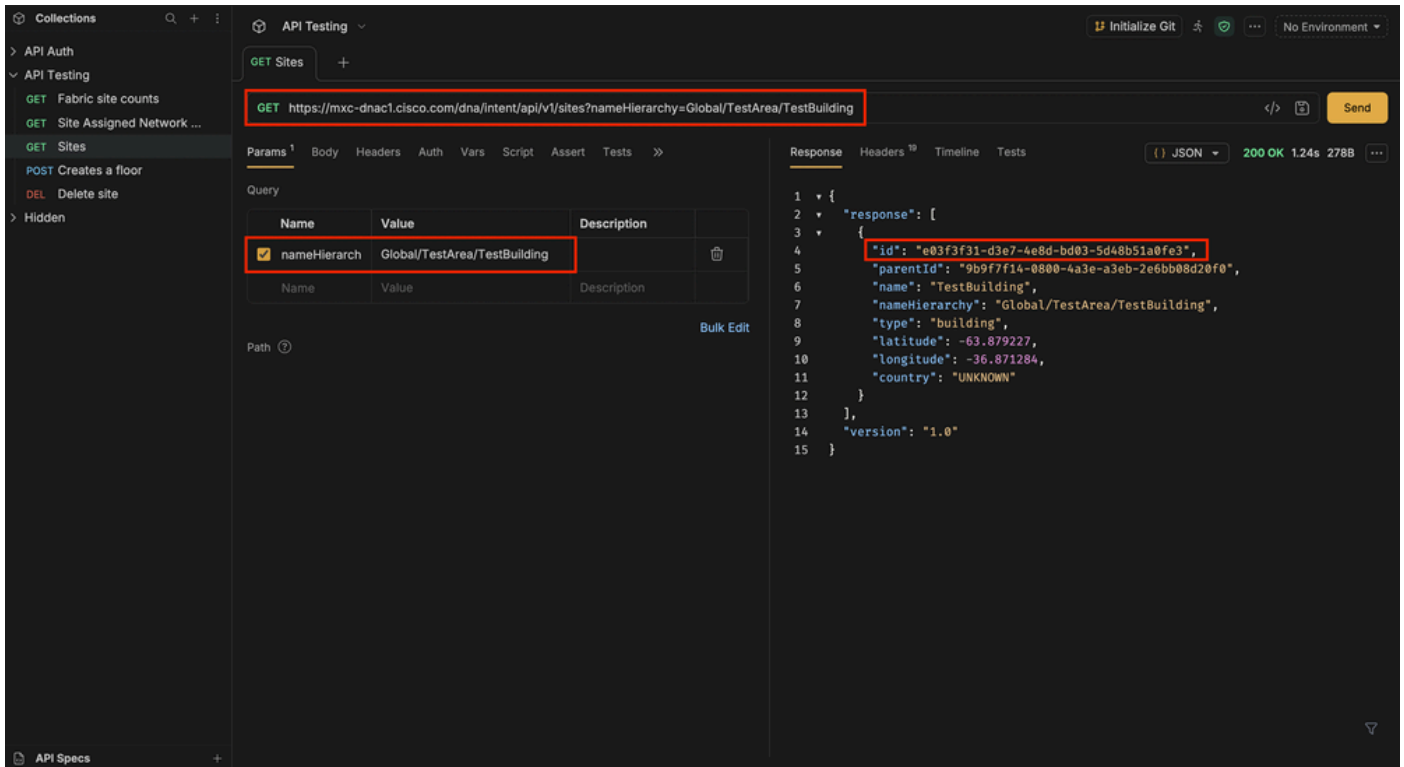


Netzwerkhierarchie

Schritt 1: Übergeordnete ID abrufen

Verwenden Sie die API GET `/dna/intent/api/v1/sites`, um die ID von TestBuilding zu ermitteln.

- Abfrageparameter: `nameHierarchy=Global/TestArea/TestBuilding`
- Kopieren Sie die ID aus der Antwort. Dies ist Ihre `parentId`.



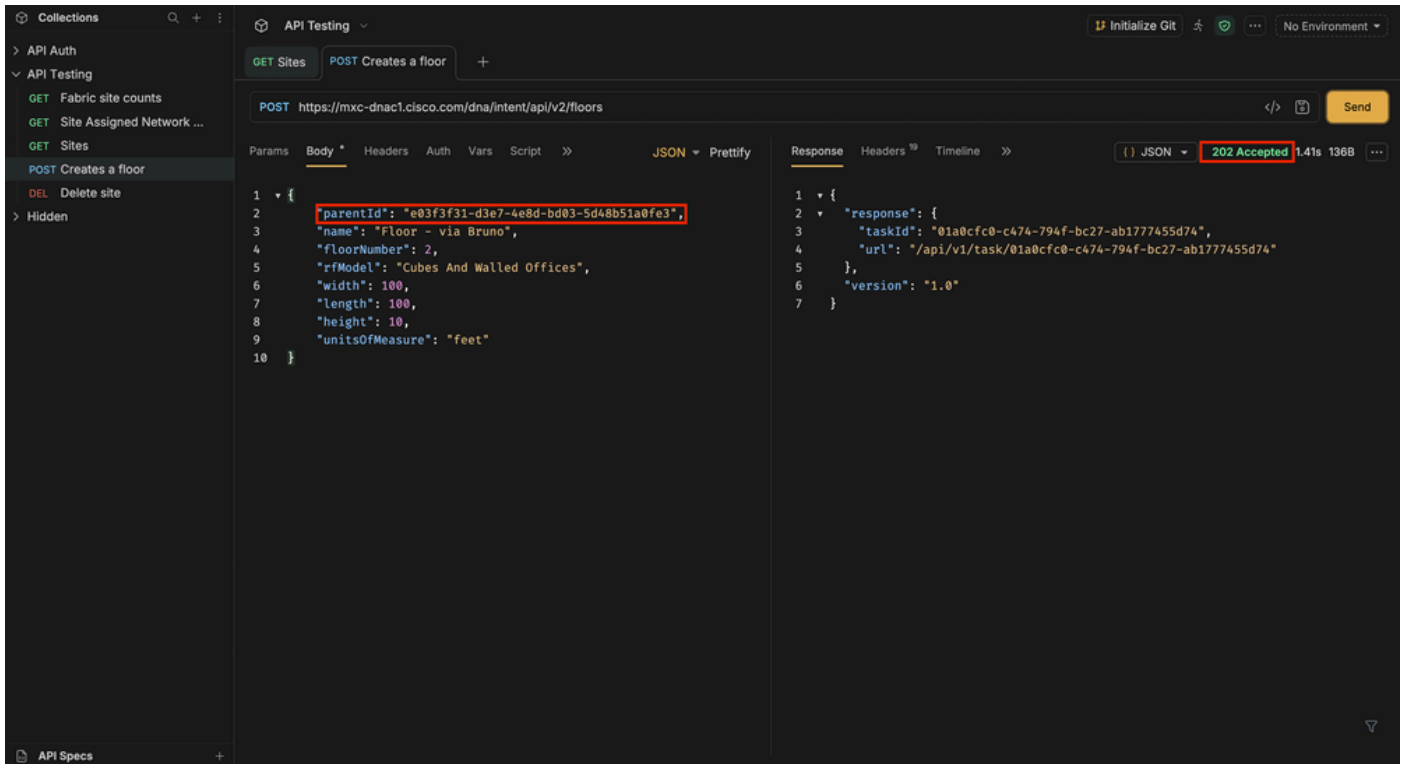
GET-Anforderung für `/dna/intent/api/v1/sites?nameHierarchy=Global/TestArea/TestBuilding`

Phase 2: Erstellen Sie den Boden

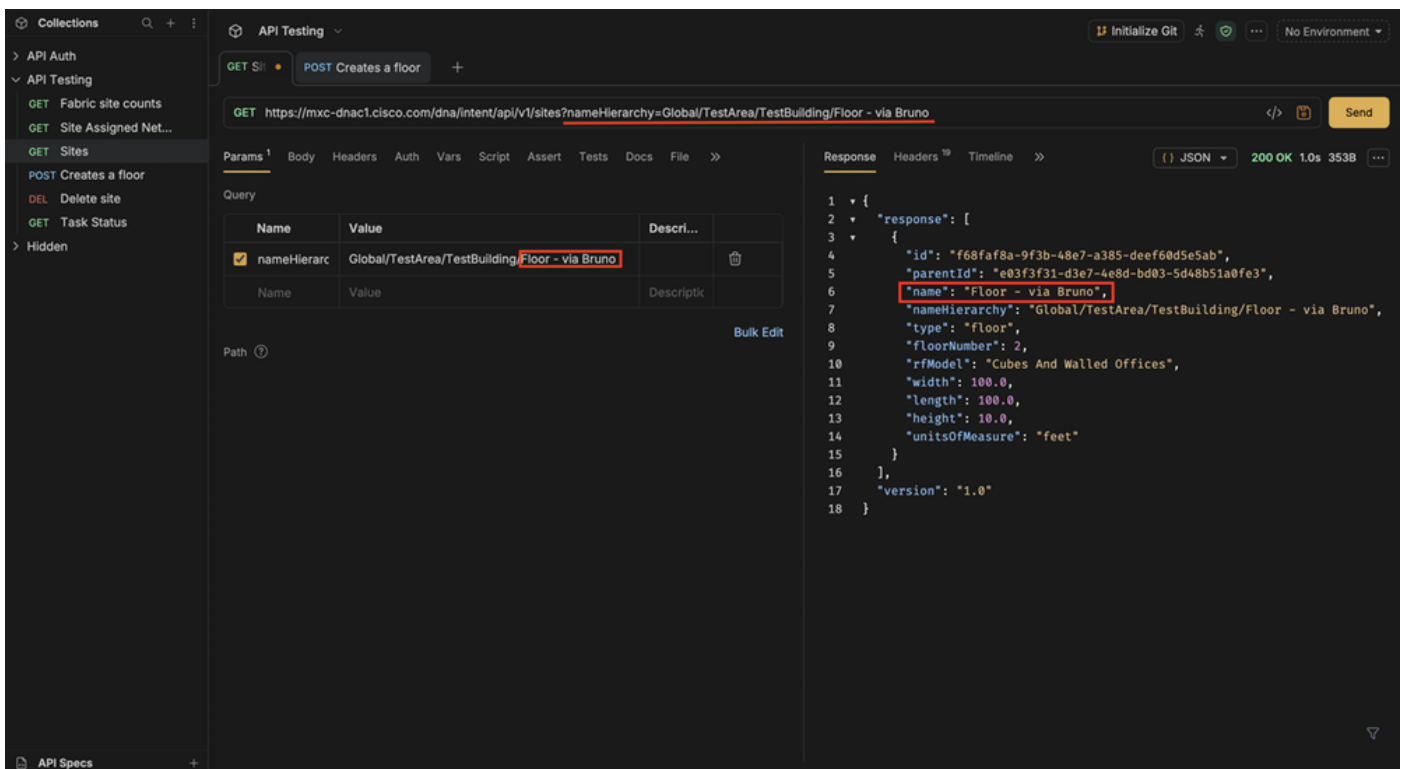
1. Erstellen Sie eine POST-Anforderung für `/dna/intent/api/v2/floors`.
2. Geben Sie auf der Registerkarte Body (JSON) die erforderlichen Parameter ein:

```
{
  "name": "Floor 1",
  "parentId": "your-building-id-here",
  "floorNumber": 1,
  "rfModel": "Cubic-Office",
  "width": 100,
  "length": 100,
  "height": 10,
  "unitsOfMeasure": "feet"
}
```

3. Klicken Sie auf Senden. Eine erfolgreiche Reaktion bestätigt die Bodenbildung.



POST-Anfrage zur Erstellung des Parkplatzes



GET-API zur Validierung der Stockwerkerstellung über eine POST-Anfrage.

Um zu überprüfen, ob die Fertigungshalle erfolgreich erstellt wurde, können Sie die vorherige API ausführen und die Abfrage entsprechend des oben gezeigten hierarchischen Pfads ändern.

Beispiel 2: Schnittstelleninformationen abrufen

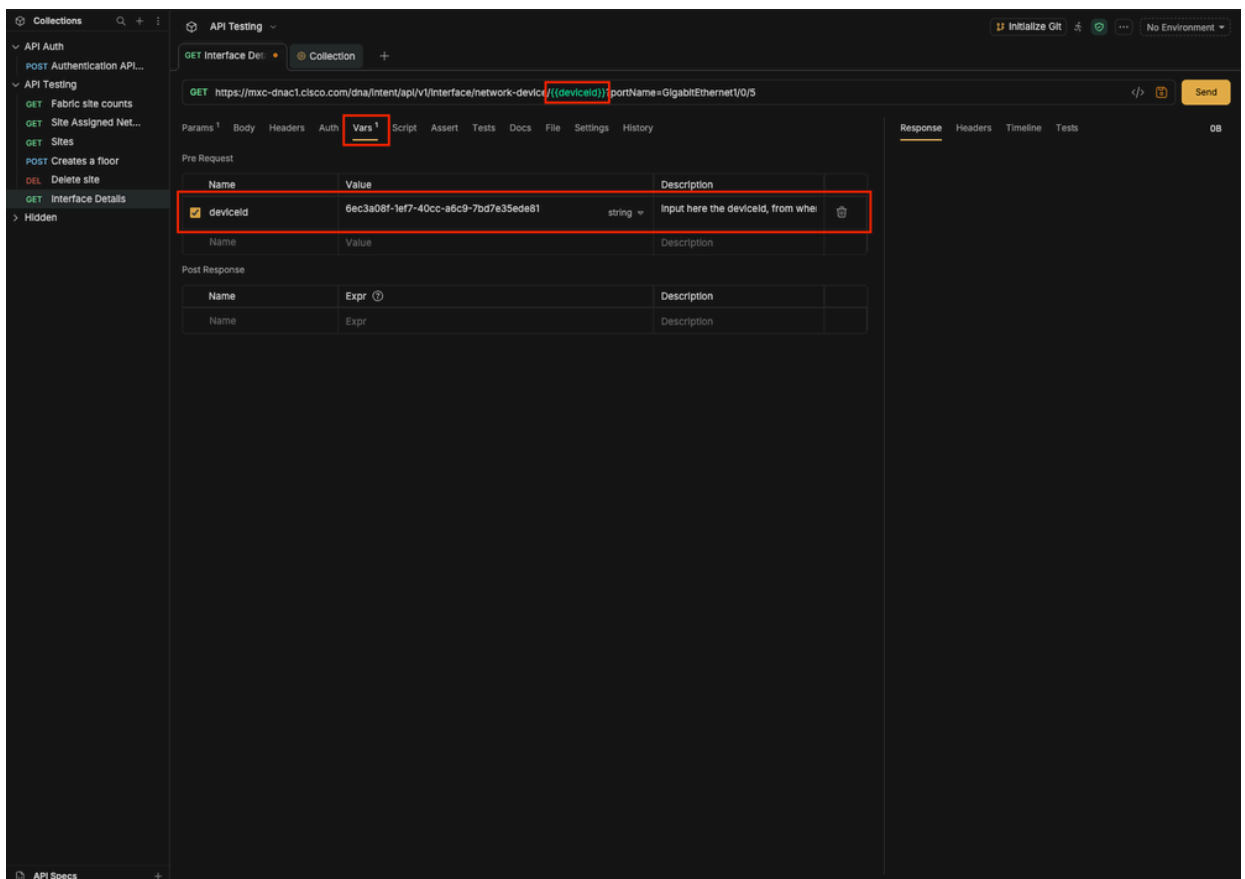
Dieses Beispiel zeigt, wie der Status einer bestimmten Schnittstelle auf einem Netzwerkgerät abgerufen wird.

Schritt 1: Identifizieren der Geräte-ID

Gehen Sie in der Catalyst Center-GUI zu Inventory, klicken Sie auf Ihr Gerät, und sehen Sie sich die URL in Ihrem Browser an. Das letzte Segment ist die Geräte-ID.

Phase 2: Parametrisierte Anforderung in Bruno erstellen

1. Erstellen Sie eine GET-Anforderung für `/dna/intent/api/v1/interface/network-device/{{deviceId}}`.
2. Fügen Sie auf der Registerkarte Vars der Anforderung eine Variable mit dem Namen `deviceId` hinzu, und fügen Sie Ihre ID ein.
3. Fügen Sie einen Abfrageparameter hinzu, um nach einem bestimmten Port zu filtern:
 - Wichtigste: `name`
 - Wert: `GigabitEthernet1/0/5`



Erstellen von Variablen in Bruno

4. Klicken Sie auf Senden.

The screenshot displays the API Testing tool interface. On the left, a sidebar lists collections including 'API Auth', 'API Testing', and 'Hidden'. The main panel shows a GET request to the URL `https://mxc-dnac1.cisco.com/dna/intent/api/v1/interface/network-device/{{deviceId}}?portName=GigabitEthernet1/0/5`. The 'Params' tab is active, showing a table with one parameter:

Name	Value	Description
portName	GigabitEthernet1/0/5	

Below the table is a 'Bulk Edit' button. The 'Response' tab on the right shows the JSON output of the request, which is a 200 OK status. The response is a JSON object containing detailed interface information for GigabitEthernet1/0/5.

```
1 {
2   "response": [
3     {
4       "addresses": [],
5       "adminStatus": "UP",
6       "className": null,
7       "deviceId": "6ce3a08f-1ef7-40cc-a6c9-7bd7e358de81",
8       "duplex": "FullDuplex",
9       "ifIndex": "12",
10      "interfaceType": "Physical",
11      "ipv4Address": null,
12      "ipv4Mask": null,
13      "isisSupport": "false",
14      "lastIncomingPacketTime": 1786685566000,
15      "lastOutgoingPacketTime": 1790265622000,
16      "lastUpdated": null,
17      "macAddress": "000000000000",
18      "mappedPhysicalInterfaceId": null,
19      "mappedPhysicalInterfaceName": null,
20      "mediaType": null,
21      "natu": "1500",
22      "nativeVlanId": "1",
23      "ospfSupport": "false",
24      "pid": "20",
25      "portMode": "access",
26      "portName": "GigabitEthernet1/0/5",
27      "portType": "Ethernet Port",
28      "serialNo": "000000000000",
29      "series": "000000000000",
30      "speed": "1000000",
31      "status": "up",
32      "vlanId": "1",
33      "voiceVlan": "",
34      "description": "",
35      "name": null,
36      "instanceUuid": "18208b55-383c-4970-ab8b-db942252974d",
37      "instanceTenantId": "66571d0b1efef061f865c6d5",
38      "id": "18208b55-383c-4970-ab8b-db942252974d"
39    }
40  ],
41  "version": "1.0"
42 }
```

GET-Anforderung, die eine Abfrage anwendet

Schlussfolgerung

Catalyst Center Northbound-APIs bieten Netzwerkadministratoren hohe Flexibilität. Durch den Übergang vom GUI-basierten Management zu API-basierten Workflows können Sie sich wiederholende Aufgaben automatisieren, eine konsistente Konfiguration sicherstellen und intelligente Netzwerkfunktionen in Ihre IT-Umgebung integrieren.

Informationen zu dieser Übersetzung

Cisco hat dieses Dokument maschinell übersetzen und von einem menschlichen Übersetzer editieren und korrigieren lassen, um unseren Benutzern auf der ganzen Welt Support-Inhalte in ihrer eigenen Sprache zu bieten. Bitte beachten Sie, dass selbst die beste maschinelle Übersetzung nicht so genau ist wie eine von einem professionellen Übersetzer angefertigte. Cisco Systems, Inc. übernimmt keine Haftung für die Richtigkeit dieser Übersetzungen und empfiehlt, immer das englische Originaldokument (siehe bereitgestellter Link) heranzuziehen.