

Comprendre et utiliser les API Catalyst Center

Table des matières

[Introduction](#)

[Informations générales](#)

[Conditions préalables](#)

[Exigences](#)

[Composants utilisés](#)

[Informations générales](#)

[Présentation du protocole HTTP](#)

[La requête HTTP](#)

[La réponse HTTP](#)

[Méthodes HTTP \(verbes\)](#)

[Codes d'état HTTP](#)

[Présentation de REST](#)

[API Catalyst Center et REST](#)

[Récupérer un jeton d'authentification à l'aide de Bruno](#)

[1. Identifier le point de terminaison API](#)

[2. Configurer Bruno pour l'authentification](#)

[3. Définir un en-tête global au Bruno](#)

[Exemple 1 : Créer un étage de site](#)

[Étape 1: Obtenir l'ID parent](#)

[Étape 2: Création du sol](#)

[Exemple 2 : Récupérer les informations d'interface](#)

[Étape 1: Identifier l'ID de périphérique](#)

[Étape 2: Créer une requête paramétrée dans Bruno](#)

[Conclusion](#)

Introduction

Ce document décrit comment accéder et utiliser les API CaTC Northbound (Intent) en utilisant un client API et les outils de développement intégrés de la plate-forme.

Informations générales

Il couvre les concepts de base HTTP/REST et fournit des exemples pratiques pour

l'automatisation et les tests.

Conditions préalables

Exigences

Cisco recommande de connaître les sujets suivants :

- Protocoles HTTP/1.1 et codes d'état.
- REPOS style architectural.
- Compréhension de base de Cisco Catalyst Center (CaTC)
- Compréhension de base de Bruno (ou de clients API similaires comme Postman).

Composants utilisés

Les informations contenues dans ce document sont basées sur les versions matérielles et logicielles suivantes :

- Cisco Catalyst Center versions 2.3.7.x et 3.x
- Bruno version 4.1.0

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. Si votre réseau est en ligne, assurez-vous de bien comprendre l'incidence possible des commandes.

Informations générales

Les API ascendantes (Intent API) de Catalyst Center permettent aux utilisateurs d'interagir avec le contrôleur en dehors de l'interface utilisateur graphique. Cela permet l'automatisation des workflows sur mesure, l'intégration avec des outils tiers et une gestion efficace du réseau à grande échelle.

Présentation du protocole HTTP

Le protocole HTTP (Hypertext Transfer Protocol) est la base de la communication de données sur

le Web. Il adhère à un modèle client-serveur dans lequel le client envoie une requête et le serveur renvoie une réponse.

La requête HTTP

Une requête HTTP se compose de quatre parties :

1. Ligne de demande : Spécifie la méthode (y compris GET et POST), l'URI et la version du protocole (telle que GET /api/v1/nodes HTTP/1.1).
2. Header (En-tête) : Fournir des métadonnées telles que les identifiants d'authentification et le type de contenu.
3. Ligne vide : Sépare les en-têtes du corps.
4. Corps du message (facultatif) : Contient les données envoyées au serveur (courant dans les tests POST/PUT/PATCH).

La réponse HTTP

Une réponse HTTP reflète la structure de la requête :

1. Ligne d'état : Inclut le code d'état (par exemple, 200 OK).
2. Header (En-tête) : Détails sur le serveur et les données de réponse.
3. Corps du message : Données réelles requises (généralement au format JSON pour Catalyst Center).

Méthodes HTTP (verbes)

- GET : Extrait des données d'une ressource.
- POST : Crée une nouvelle ressource.
- PUT : Remplace une ressource existante.
- CORRECTIF : Applique des modifications partielles à une ressource.
- DELETE : Supprime une ressource.

Codes d'état HTTP

Les codes d'état HTTP sont des codes numériques normalisés renvoyés par un serveur dans le cadre d'une réponse HTTP. Ils indiquent le résultat général d'une demande du client et aident à comprendre si la demande a réussi, nécessite une action supplémentaire ou a rencontré une erreur.

Les codes sont regroupés en cinq catégories principales selon leur premier chiffre.

- Les codes 1xx sont informatifs et indiquent que la demande a été reçue et que le traitement se poursuit.
- Les codes 2xx indiquent que la demande a abouti, ce qui signifie que le serveur a bien reçu, compris et traité la demande.
- Les codes 3xx sont utilisés pour la redirection et indiquent qu'une action supplémentaire peut être requise pour terminer la demande.
- Les codes 4xx représentent des erreurs du client, ce qui signifie qu'il y a un problème avec la requête envoyée par le client.
- Les codes 5xx représentent des erreurs de serveur, ce qui signifie que le serveur a rencontré un problème lors de la tentative de traitement d'une requête.

Présentation de REST

REST (Representational State Transfer) est un style architectural permettant de concevoir des applications en réseau. Les API RESTful utilisent HTTP comme mécanisme de transport et utilisent généralement JSON pour l'échange de données en raison de sa nature légère.

Lorsqu'un client interagit avec une API REST, le serveur renvoie une représentation de l'état actuel d'une ressource. Cette représentation est transférée au client dans le cadre de la réponse et peut être exprimée dans un format différent.

REST n'est pas un protocole. Il s'agit plutôt d'un style hiérarchique qui définit un ensemble de principes et de contraintes pour la création de systèmes distribués et d'API. Dans les API modernes, JSON est l'un des formats les plus courants car il est léger et facile à traiter.

Une interaction REST type est conforme à un modèle basé sur le client (ce point a déjà été mentionné dans l'article). Le client envoie une requête HTTP à l'aide de méthodes telles que GET, POST, PUT, PATCH ou DELETE. Le serveur traite et renvoie une réponse HTTP contenant les informations demandées ou le résultat de l'opération.

Les API RESTful utilisent généralement HTTP comme mécanisme de communication pour effectuer des opérations sur les ressources. Dans ce modèle, le client identifie une ressource/un point de terminaison, puis utilise la méthode HTTP pour indiquer l'action à effectuer sur cette ressource. Une API REST peut être décrite à l'aide de plusieurs éléments principaux. Tout d'abord, l'URI est utilisé pour identifier la ressource ou le point d'extrémité auquel le client veut accéder. L'URI fournit l'emplacement ou l'identificateur de cette ressource dans l'API. Deuxièmement, le client utilise des méthodes HTTP standard telles que GET, POST, PUT, PATCH et DELETE pour définir l'opération de requête. Par exemple, GET peut être utilisé pour récupérer une ressource, POST pour en créer une et DELETE pour en supprimer entièrement.

une. Enfin, la ressource est échangée via une représentation, qui est définie par son type de support. Cette représentation est la forme sous laquelle les données de ressource sont transférées entre le client et le serveur, telle que JSON ou XML.

API Catalyst Center et REST

CaTC simplifie l'exploitation des réseaux de campus en centralisant l'automatisation, la configuration, la gestion des politiques, la surveillance et l'analyse. Elle permet aux administrateurs réseau de déployer et de gérer les périphériques plus efficacement, de maintenir des politiques cohérentes, de surveiller le comportement du réseau et d'améliorer la visibilité opérationnelle globale. Ces fonctionnalités sont accessibles via l'interface utilisateur graphique (GUI), les API et les intégrations externes.

La plate-forme CaTC fournit également des API qui peuvent être utilisées pour créer des intégrations personnalisées et automatiser les opérations réseau. En combinant les API CaTC aux solutions Cisco existantes ou aux applications tierces, les entreprises peuvent étendre les fonctionnalités de la plate-forme et créer des workflows personnalisés pour des opérations ou des besoins métier spécifiques.

Après avoir traité des concepts fondamentaux, ce document se concentre sur les API ascendantes, également appelées API d'intention, dans Cisco Catalyst Center. Ces API exposent des fonctionnalités spécifiques de Catalyst Center par le biais d'une abstraction basée sur des politiques d'intention commerciale, permettant aux administrateurs et aux développeurs de se concentrer sur le résultat souhaité plutôt que sur les étapes de configuration individuelles requises pour y parvenir.

Les API d'intention Catalyst Center sont des API RESTful qui utilisent le format de données JSON pour interagir avec la plate-forme. Ils peuvent être utilisés pour découvrir des informations réseau, récupérer des données opérationnelles et effectuer des tâches de contrôle et d'automatisation prises en charge sur le réseau.

Récupérer un jeton d'authentification à l'aide de Bruno

Avant d'effectuer des appels API, vous devez vous authentifier pour recevoir un jeton.

1. Identifier le point de terminaison API

1. Connectez-vous à Catalyst Center.
2. Accédez à Menu > Platform > Developer Toolkit.

3. Dans le menu de gauche, sélectionnez Authentication > Authentication API.

The screenshot shows the Cisco Catalyst Center Developer Toolkit interface. On the left, a sidebar menu has 'Authentication' highlighted with a red box. The main area is titled 'Authentication' and contains a table of API endpoints. The 'Authentication API' entry is highlighted with a red box. Below the table, the 'Cisco DNA Center System' section is visible.

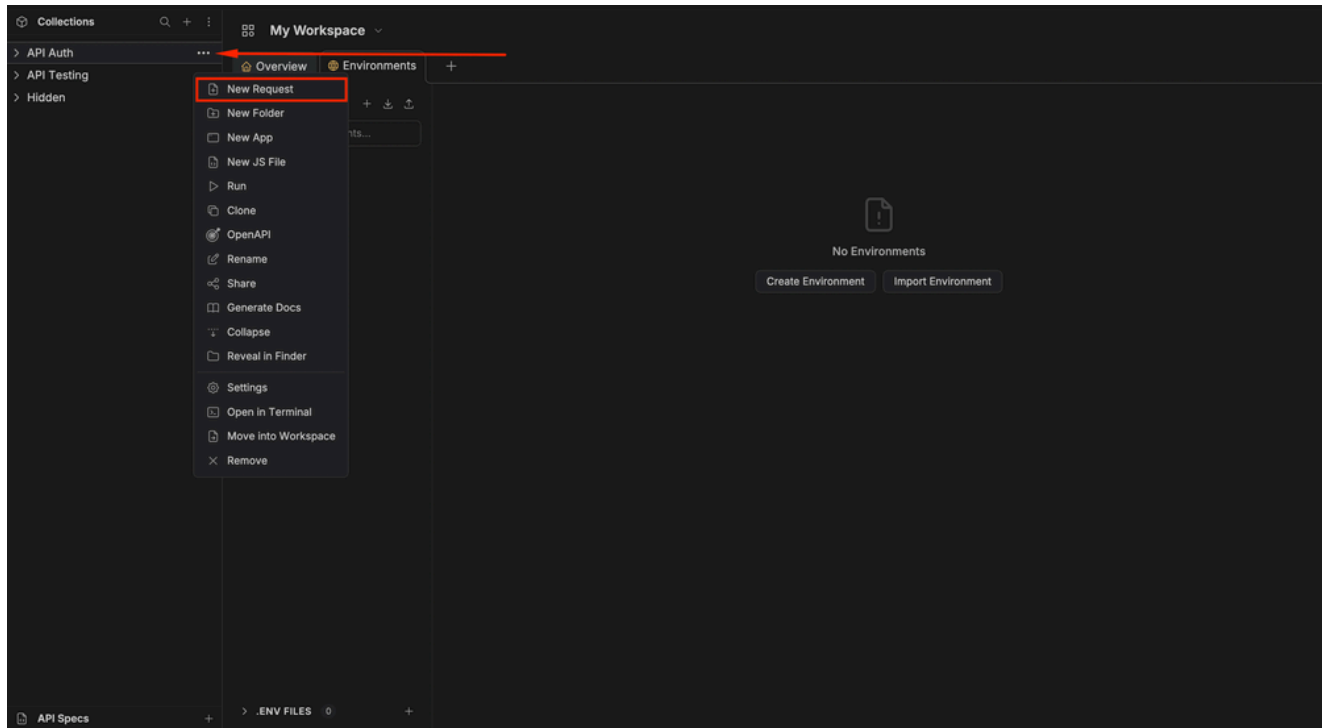
Method	Name	Description	URL	Actions
POST	importCertificate	This API enables a user to import a PEM certificate and its key for the controller and/or disaster recovery.	/dna/intent/api/v1/certificate	...
POST	importCertificateP12	This API enables a user to import a PKCS12 certificate bundle for the controller 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	...

Kit de développement Catalyst Center

4. Copiez l'URL (généralement POST /dna/system/api/v1/auth/token).

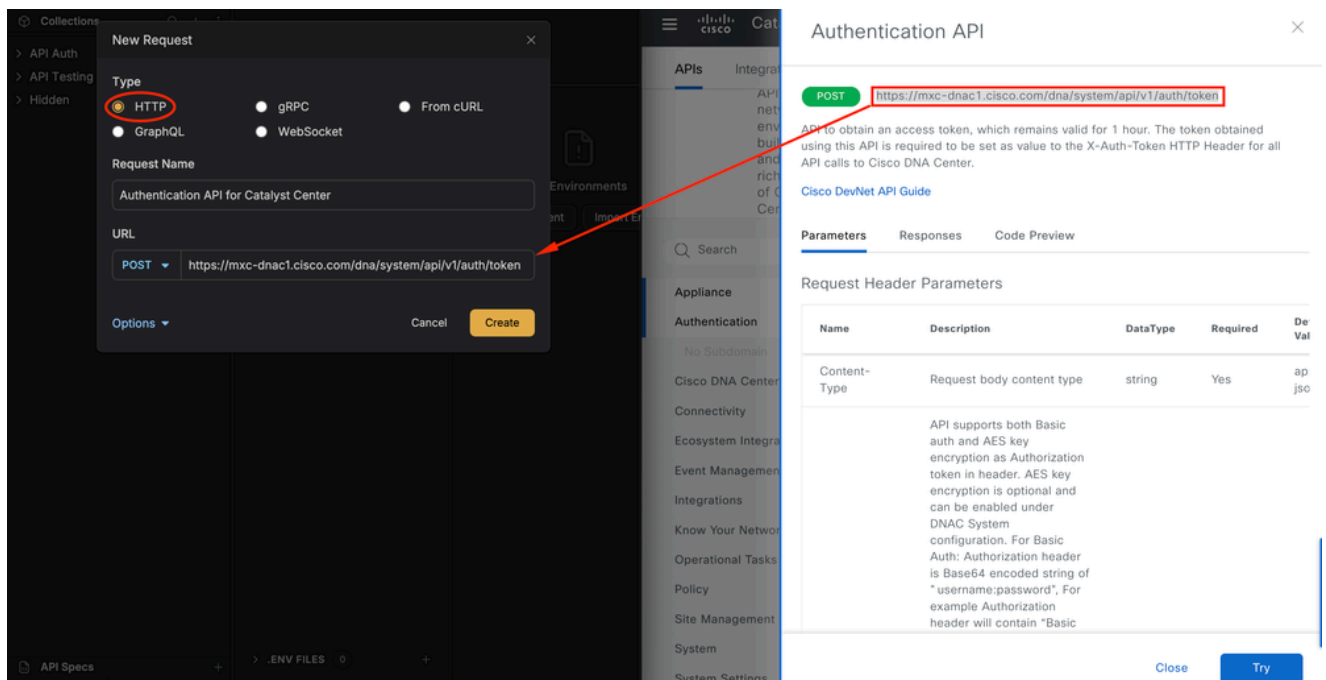
2. Configurer Bruno pour l'authentification

1. Dans Bruno, créez une nouvelle demande.
2. Définissez le type sur HTTP et la méthode sur POST.



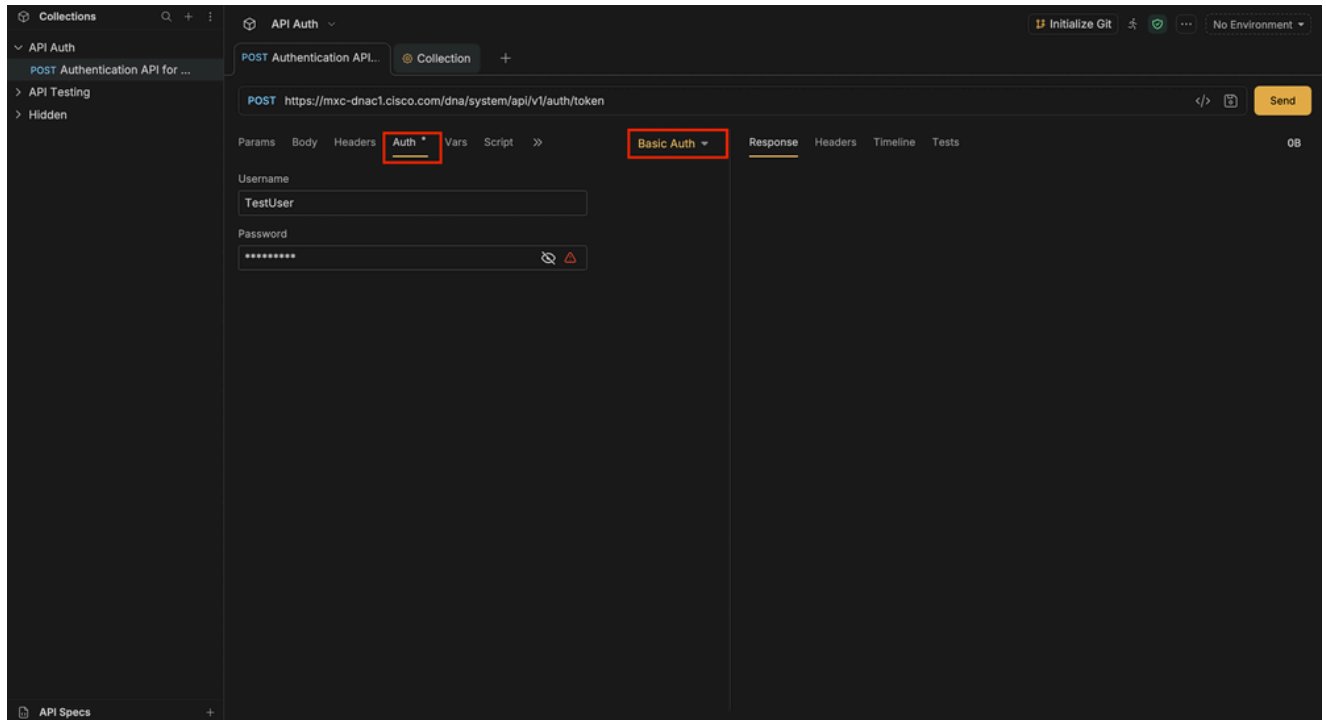
Création d'une nouvelle demande à Bruno

3. Saisissez l'URL de l'API d'authentification.



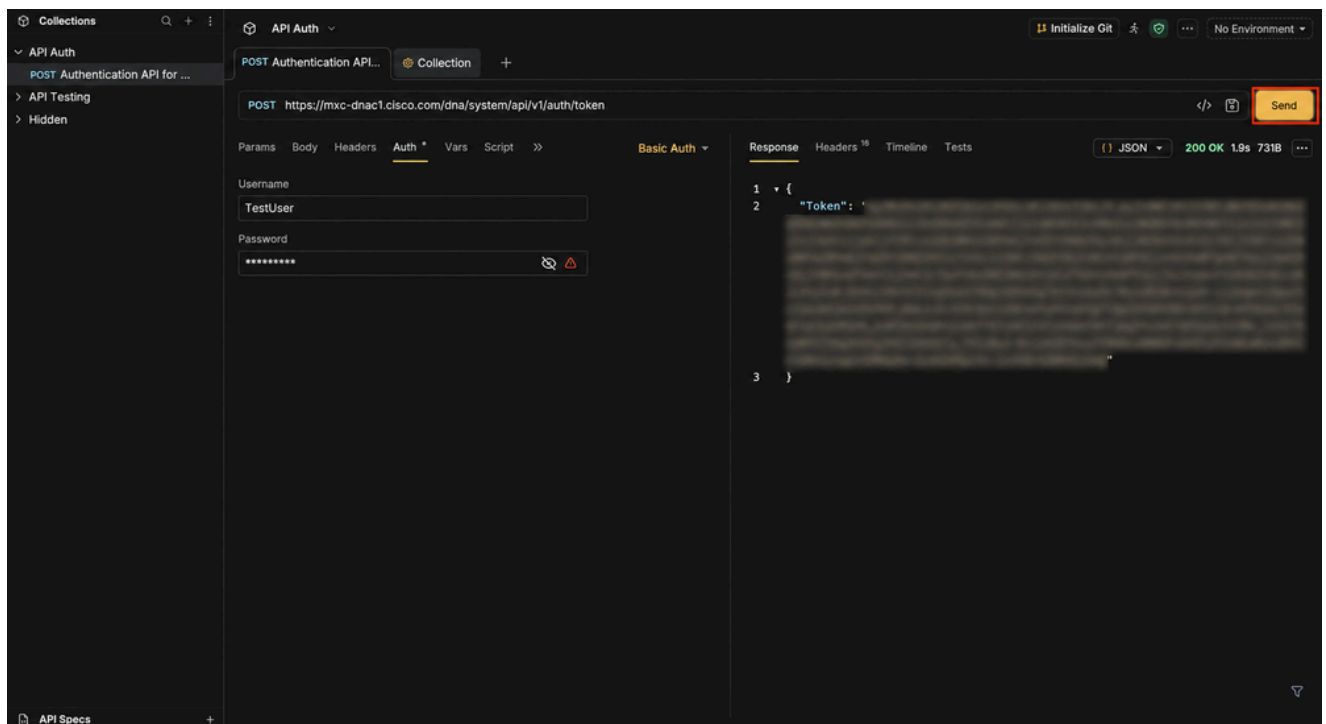
Création d'une nouvelle demande à Bruno

4. Accédez à l'onglet Auth et sélectionnez Basic Auth.



Authentification de base pour l'API d'authentification

5. Saisissez vos informations d'identification SUPER ADMIN pour Catalyst Center.
6. Cliquez sur Envoyer.



Exécution de l'API d'authentification

Le serveur renvoie un état 200 OK et une charge utile JSON contenant le jeton.

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

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

Obtention du codage base64 pour le nom d'utilisateur et le mot de passe

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==

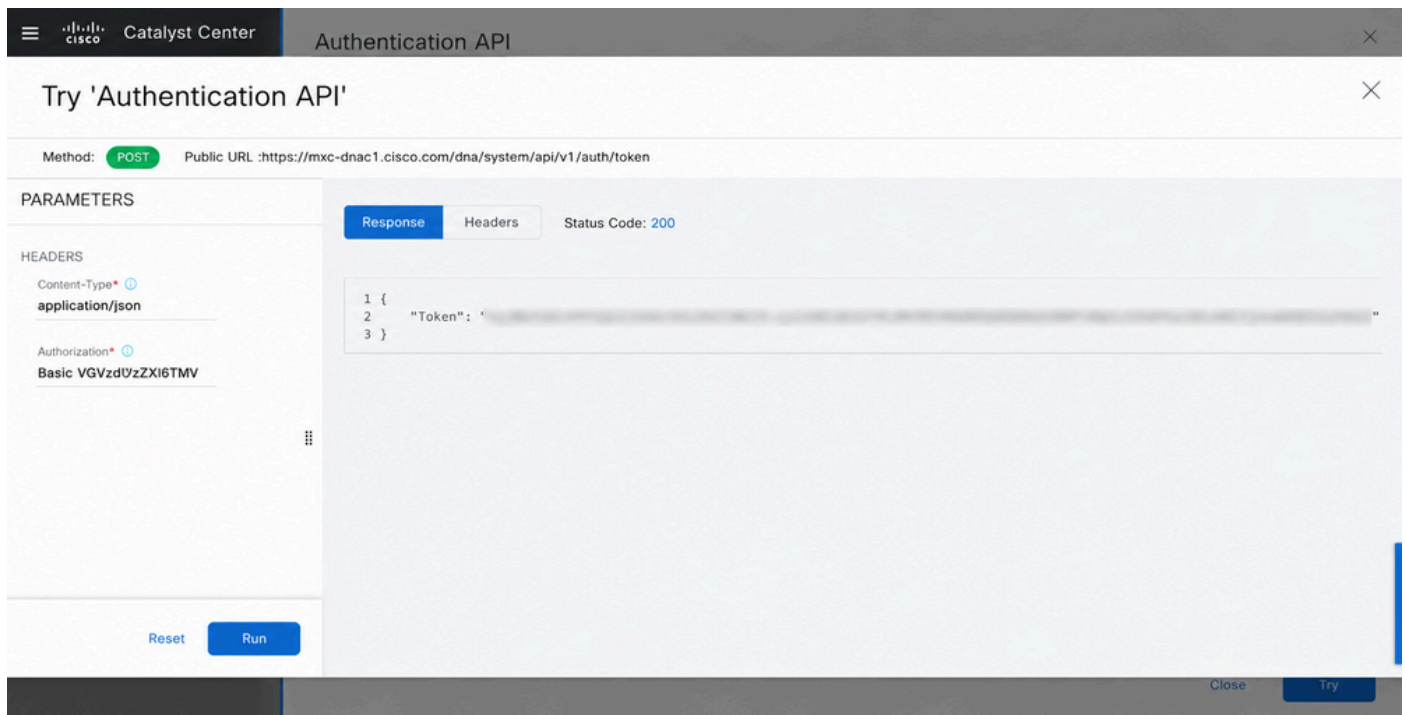
Reset Run

Enter Parameters

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

Close Try

Saisie d'authentification de base avec nom d'utilisateur et mot de passe codés base64



Obtention du jeton d'authentification à partir de la GUI de Catalyst Center.

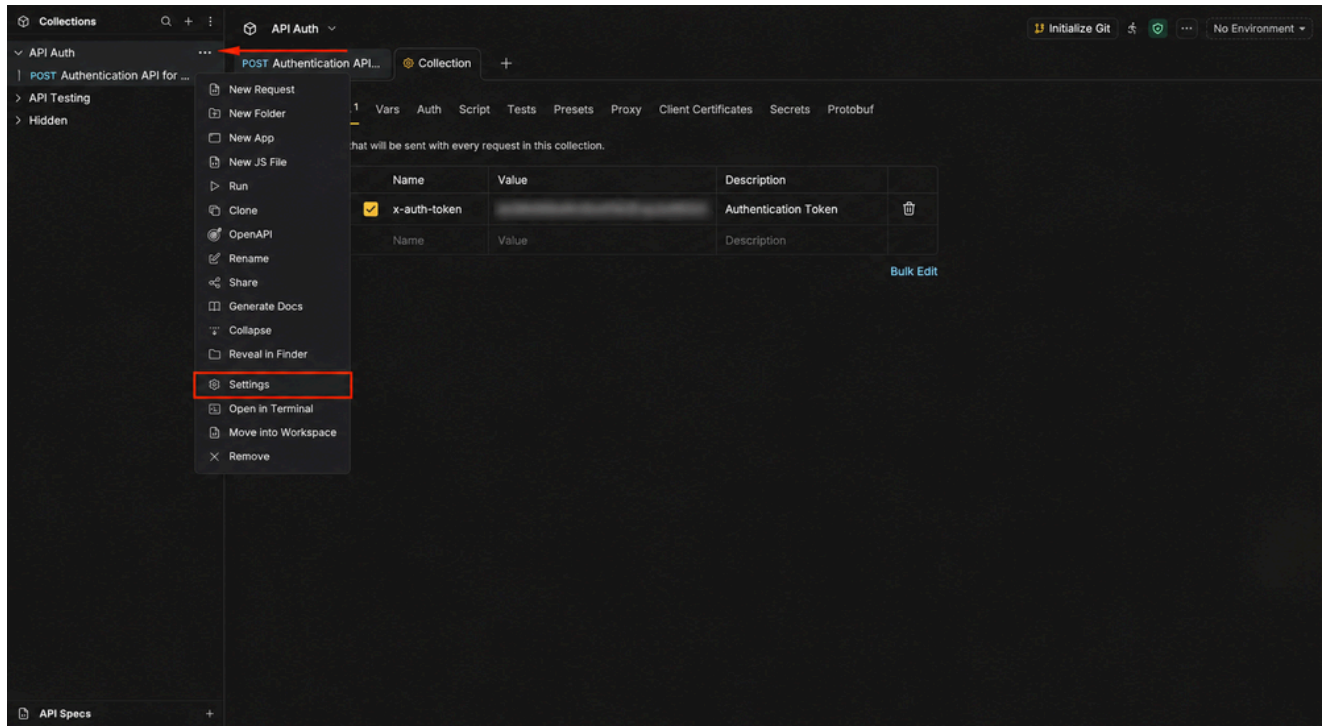
À ce stade, vous avez le jeton d'authentification, soit sur la réponse Bruno ou sur la réponse faite à partir de la boîte à outils pour développeurs Catalyst Center.

Maintenant, ajoutez le jeton en tant qu'En-tête sur les paramètres de collection dans Bruno, cela facilite toute autre demande créée dans la collection pour utiliser le jeton, de sorte que vous n'avez pas besoin d'entrer le jeton manuellement en tant qu'en-tête pour chaque demande.

3. Définir un en-tête global au Bruno

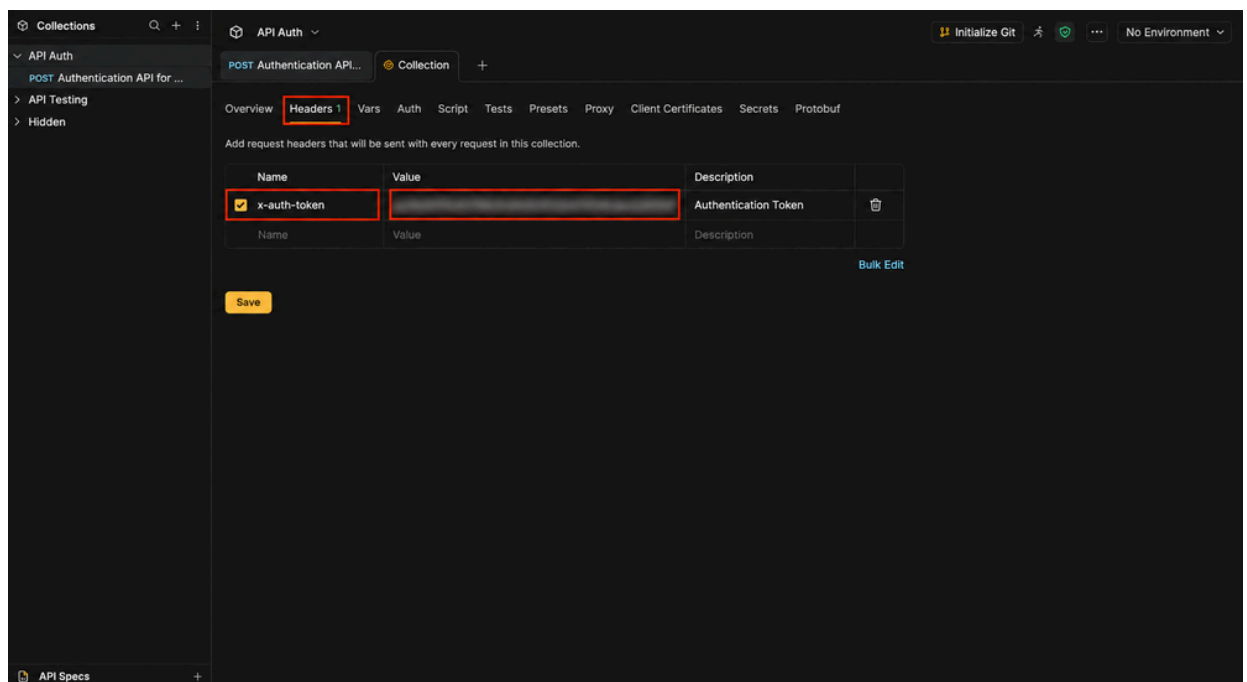
Pour éviter de réauthentifier chaque demande, définissez le jeton en tant qu'en-tête de niveau collection :

1. Cliquez avec le bouton droit sur votre collection et sélectionnez Paramètres.



Paramètres de collection

2. Accédez à l'onglet En-têtes.
3. Ajouter un nouvel en-tête :
 - Name : x-auth-token
 - Valeur: [Collez votre jeton ici]



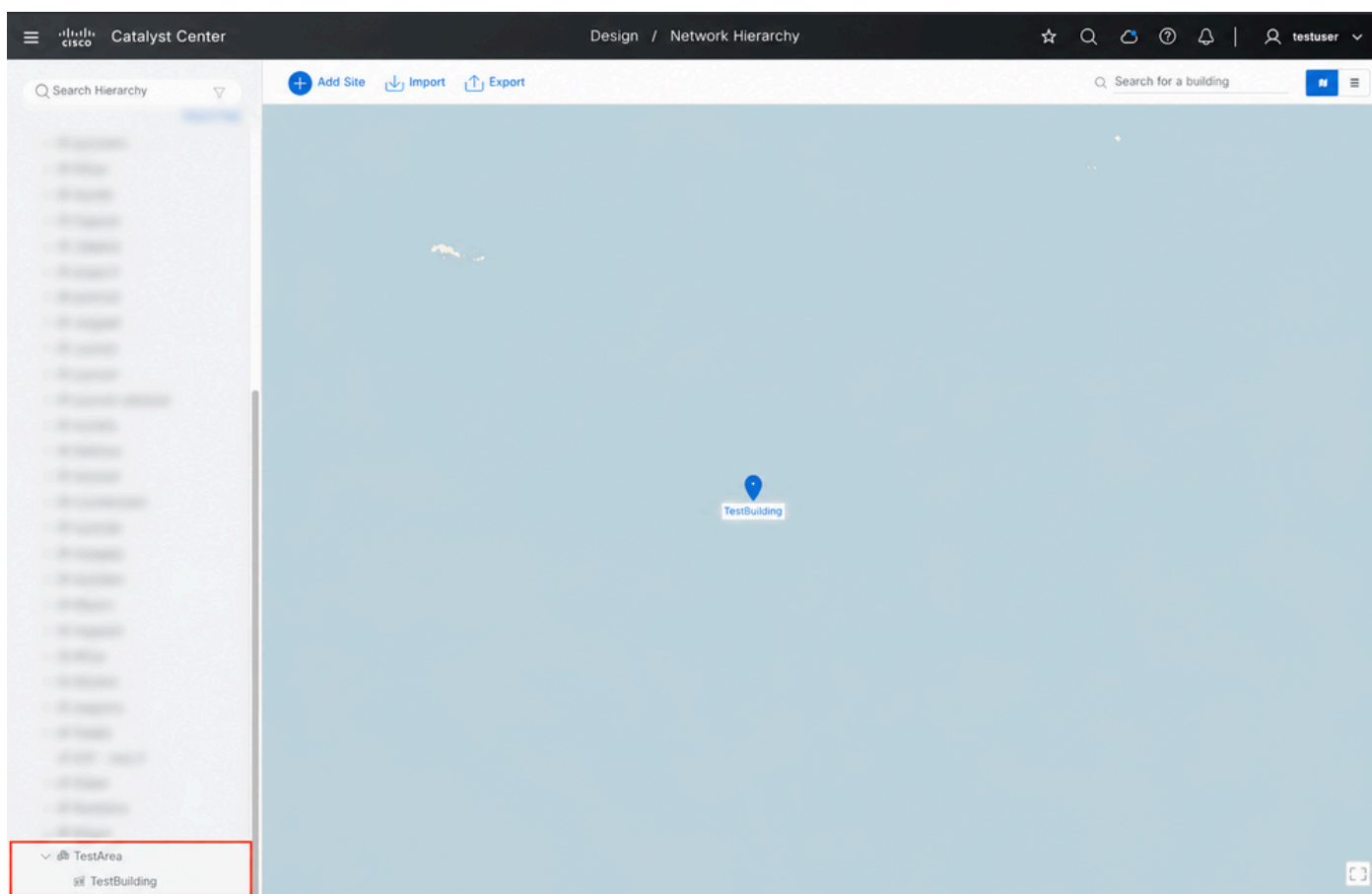
Saisie des en-têtes globaux pour la collection API dans Bruno.

4. Cliquez sur Save.

À ce stade, vous avez généré le jeton d'authentification afin d'effectuer toute autre demande d'API Catalyst Center.

Exemple 1 : Créer un étage de site

Cet exemple montre comment créer un étage dans la hiérarchie réseau (Global > TestArea > TestBuilding).

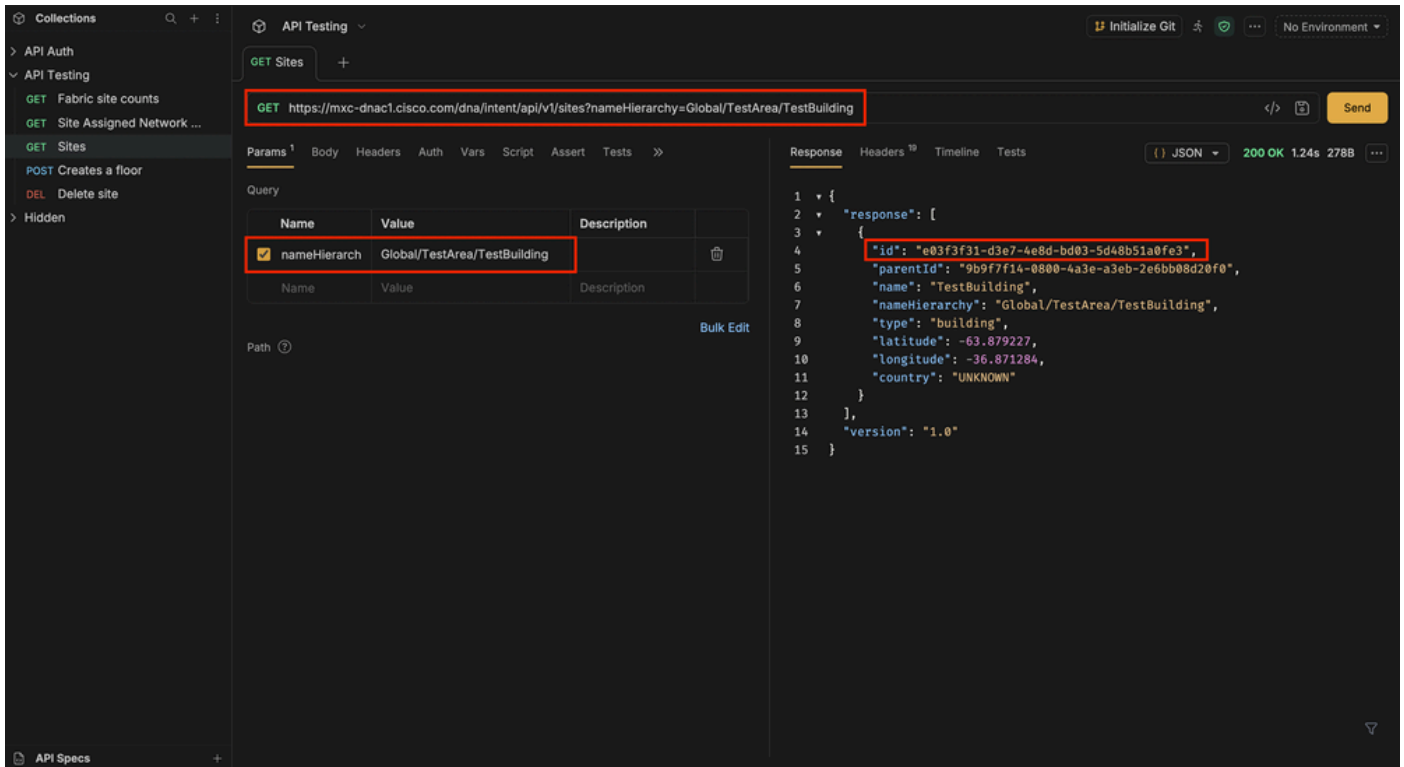


Hiérarchie réseau

Étape 1: Obtenir l'ID parent

Utilisez l'API GET `/dna/intent/api/v1/sites` pour trouver l'ID de TestBuilding.

- Paramètre de requête : `nameHierarchy=Global/ZoneTest/TestBuilding`
- Copier l'ID de la réponse ; il s'agit de votre `parentId`.



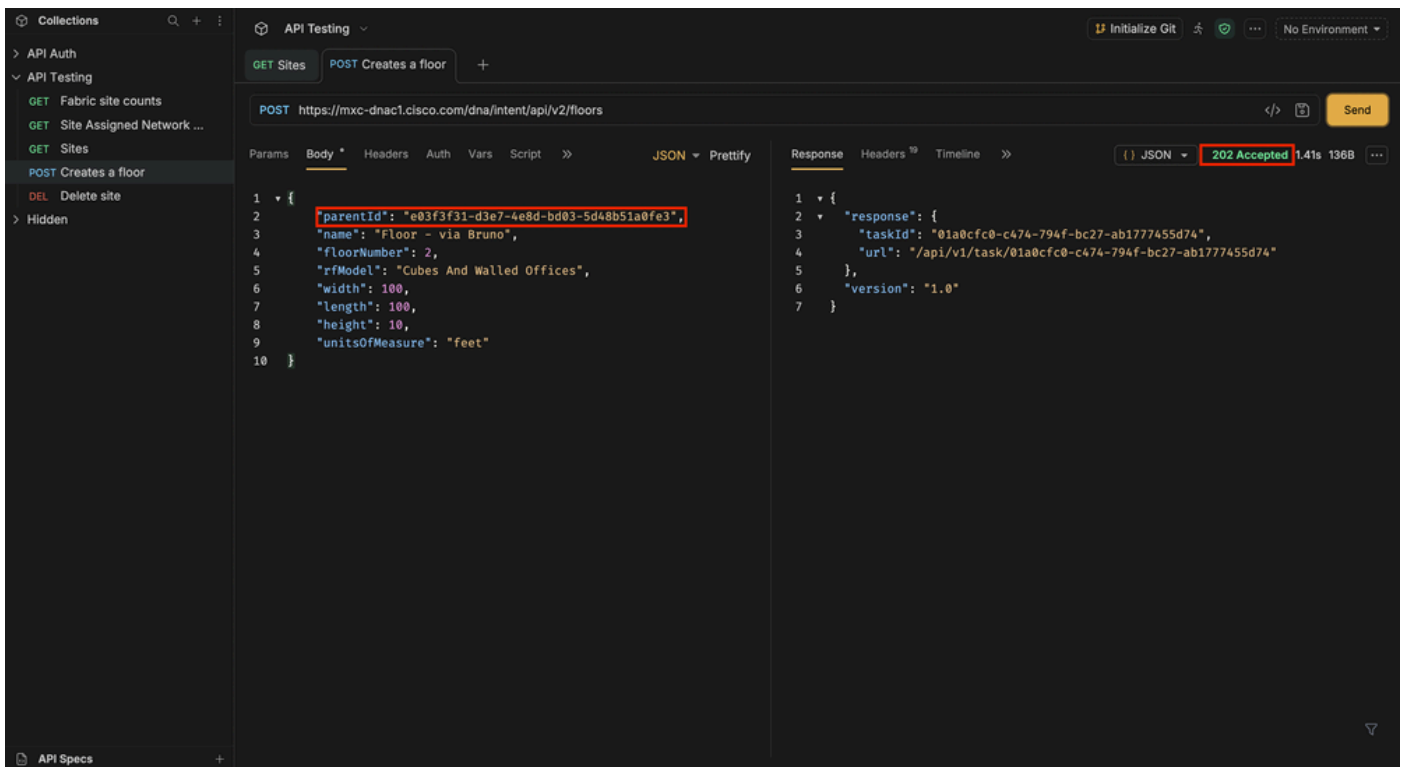
Requête GET pour /dna/intent/api/v1/sites?nameHierarchy=Global/TestArea/TestBuilding

Étape 2: Création du sol

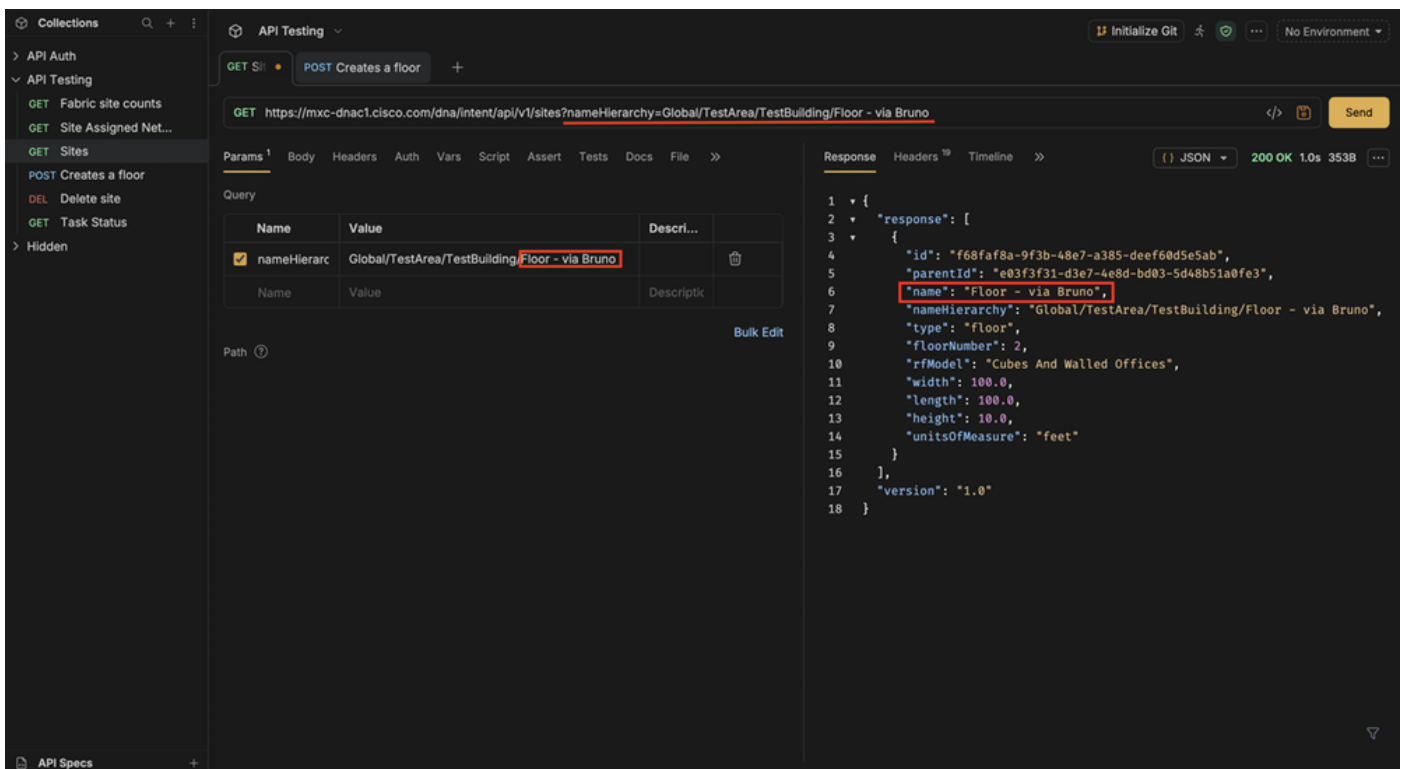
1. Créez une demande POST pour /dna/intent/api/v2/floor.
2. Dans l'onglet Body (JSON), entrez les paramètres requis :

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

3. Cliquez sur Envoyer. Une réponse positive confirme la création de l'étage.



Demande POST de création de l'étage



API GET pour valider la création d'étage via une requête POST.

Enfin, pour valider la création de l'étage, vous pouvez exécuter l'API précédente et modifier la requête pour qu'elle corresponde au chemin hiérarchique complet, comme indiqué ci-dessus.

Exemple 2 : Récupérer les informations d'interface

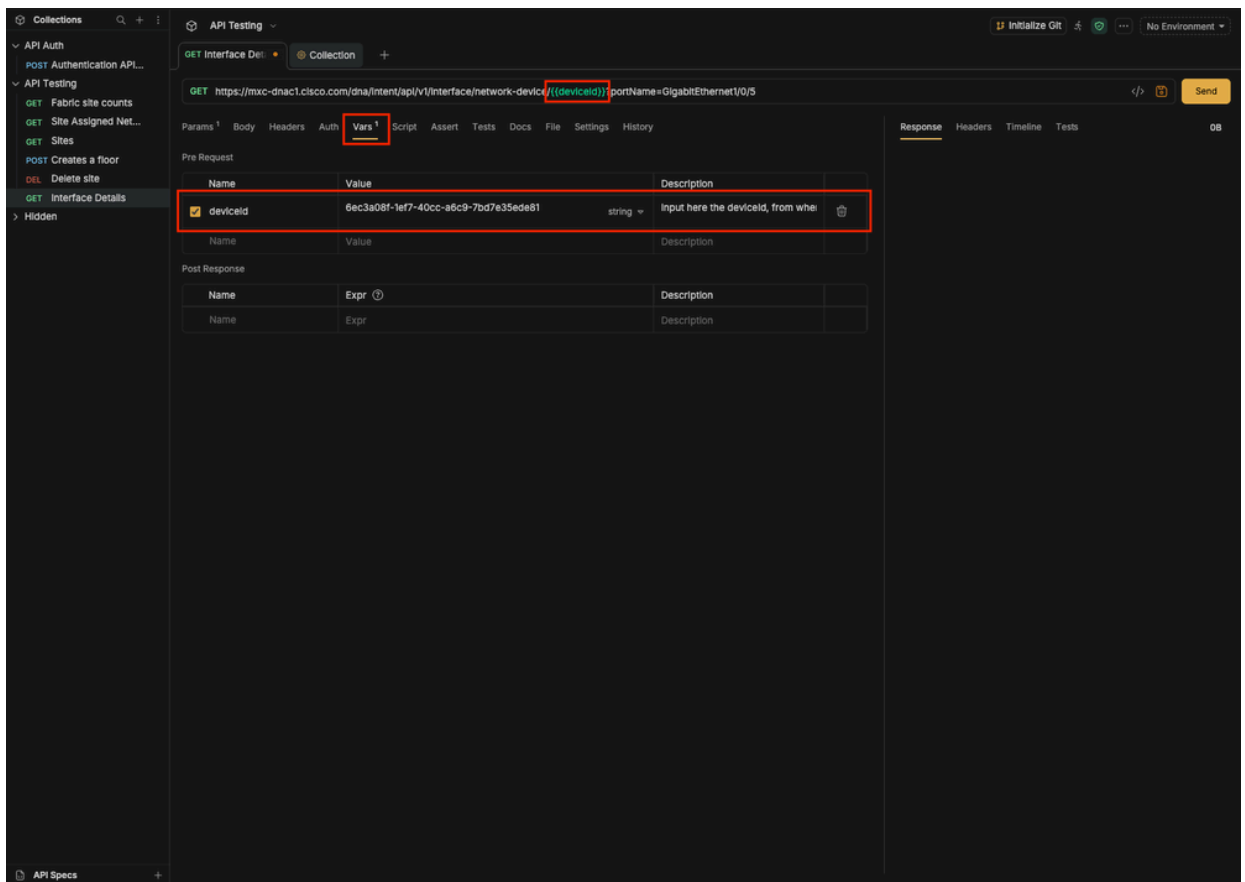
Cet exemple montre comment récupérer l'état d'une interface spécifique sur un périphérique réseau.

Étape 1: Identifier l'ID de périphérique

Dans l'interface graphique utilisateur de Catalyst Center, accédez à Inventory, cliquez sur votre périphérique et consultez l'URL dans votre navigateur. Le dernier segment est l'ID de périphérique.

Étape 2: Créer une requête paramétrée dans Bruno

1. Créez une demande GET pour `/dna/intent/api/v1/interface/network-device/{{deviceid}}`.
2. Dans l'onglet Vars de la demande, ajoutez une variable nommée `deviceid` et collez votre ID.
3. Pour filtrer un port spécifique, ajoutez un paramètre de requête :
 - Clé : `nom`
 - Valeur: `GigabitEthernet1/0/5`



Création de variables dans Bruno

4. Cliquez sur Envoyer.

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://mx-c-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 information about the network interface, including its status, type, and configuration details.

requête GET application d'une requête

Conclusion

Les API ascendantes de Catalyst Center offrent une grande souplesse aux administrateurs réseau. En passant d'une gestion basée sur une interface utilisateur graphique à des workflows basés sur une API, vous pouvez automatiser les tâches répétitives, assurer la cohérence de la configuration et intégrer l'intelligence du réseau dans votre écosystème informatique plus large.

À propos de cette traduction

Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.