# ツールを使用した API の開発/試験

この章の内容は、次のとおりです。



# API インスペクタの使用

## GUI 内の API 交換の表示

APIC グラフィカルユーザインターフェイス（GUI）でタスクを実行すると、GUI は内部 API メッセージを作成してタスクを実行するためのオペレーティング システムに送信します。 APIC の組み込み型ツールである API インスペクタ を使用して、これらの API メッセージを表示およびコピーできます。 ネットワーク管理者は、主要操作を自動化するためにこれらのメッセージを複製したり、API を使用する外部アプリケーションを開発するためにこれらのメッセージを例として使用できます。

### 手順

**ステップ 1** APIC GUI にログインします。

**ステップ 2** APIC ウィンドウの右上隅で、「welcome, <name>」メッセージをクリックしてドロップダウン リストを表示します。

**ステップ 3** ドロップダウン リストで、[Show API Inspector] を選択します。
[API Inspector] が新しいブラウザ ウィンドウで開きます。

**ステップ 4** [API Inspector] ウィンドウの [Filters] ツールバーで、表示する API ログ メッセージのタイプを選択します。

表示されたメッセージは選択されたメッセージのタイプに応じて色分けされます。 次のテーブルに、使用可能なメッセージ タイプを表示します。

| 名前 | 説明 |
|---|---|
| trace | トレース メッセージを表示します。 |
| debug | デバッグ メッセージを表示します。 このタイプには、ほとんどの API コマンドと応答が含まれます。 |
| info | 情報メッセージを表示します。 |
| warn | 警告メッセージを表示します。 |
| error | エラー メッセージを表示します。 |
| fatal | 重大メッセージを表示します。 |
| all | このチェックボックスをオンにすると、他のチェックボックスすべてがオンになります。 他のチェックボックスのいずれかをオフにすると、このチェックボックスもオフになります。 |

**ステップ 5** [Search] ツールバーで、正確な文字列に対し表示されるメッセージまたは正規表現で表示されるメッセージを検索できます。
次の表に、検索のコントロールを示します。

| 名前 | 説明 |
|---|---|
| Search | このテキスト ボックスに、直接検索の文字列を入力するか、または regex 検索の正規表現を入力します。 入力に応じて、ログ リストの最初に一致したフィールドが強調表示されます。 |
| Reset | [Search] テキスト ボックスの内容を削除するには、このボタンをクリックします。 |
| Regex | [Search] テキスト ボックスの内容を検索の正規表現として使用するには、このチェックボックスをオンにします。 |
| Match case | 検索で大文字と小文字が区別されるようにするには、このチェックボックスをオンにします。 |
| Disable | 検索を無効にし、ログ リストの検索一致結果の強調表示をクリアするには、このチェックボックスをオンにします。 |
| Next | ログ リストを次の一致したエントリまでスクロールするには、このボタンをクリックします。 このボタンは、検索がアクティブである場合にのみ表示されます。 |

| 名前 | 説明 |
|---|---|
| Previous | ログ リストを前の一致したエントリまでスクロールするには、このボタンをクリックします。 このボタンは、検索がアクティブである場合にのみ表示されます。 |
| Filter | 一致しない行を非表示にするには、このチェックボックスをオンにします。 このチェックボックスは、検索がアクティブである場合にのみ表示されます。 |
| Highlight all | すべての一致したフィールドを強調表示するには、このチェックボックスをオンにします。 このチェックボックスは、検索がアクティブである場合にのみ表示されます。 |

**ステップ 6** [Options] ツールバーで、表示されるメッセージを並べ替えることができます。
次の表に、使用可能なオプションを示します。

| 名前 | 説明 |
|---|---|
| Log | ロギングをイネーブルにするには、このチェックボックスをオンにします。 |
| Wrap | ログ リストの水平スクロールを無効にするために行の折り返しをイネーブルにするには、このチェックボックスをオンにします。 |
| Newest at the top | ログ エントリを逆の時系列で表示するには、このチェックボックスをオンにします。 |
| Scroll to latest | 最新のログ エントリに迅速にスクロールするには、このチェックボックスをオンにします。 |
| Clear | ログ リストを削除するには、このボタンをクリックします。 |
| Close | API インスペクタを閉じるには、このボタンをクリックします。 |

### 例

次の例では、API インスペクタ ウィンドウの 2 つのデバッグ メッセージを示します。

```
13:13:36 DEBUG - method: GET url: http://192.0.20.123/api/class/infraInfra.json
response: {"imdata":[{"infraInfra":{"attributes":{"instanceId":"0:0","childAction":"",
"dn":"uni/infra","lcOwn":"local","name":"","replTs":"never","status":""}}}]}

13:13:40 DEBUG - method: GET url: http://192.0.20.123/api/class/l3extDomP.json?
query-target=subtree&subscription=yes
response: {"subscriptionId":"72057598349672459","imdata":[]}
```

# Managed Object Browser の使用

Managed Object Browser、つまり Visore は、APIC に組み込まれたユーティリティで、ブラウザを使用した管理対象オブジェクト（MO）のグラフィカル表示が提供されます。 Visore ユーティリティは、APIC REST API クエリー メソッドを使用して アプリケーション セントリック インフラストラクチャ ファブリック 内でアクティブな MO を参照するので、ユーザは情報を取得するために使用されたクエリーを確認できます。 Visore ユーティリティは、設定を行うためには使用できません。

（注） Firefox、Chrome および Safari ブラウザでのみ、Visore アクセスがサポートされます。

## Visore へのアクセス

**手順**

**ステップ 1** サポートされているブラウザを開き、APIC の URL とその後に `/visore.html` を入力します。

**例：**

```
https://192.0.20.123/visore.html
```

**ステップ 2** プロンプトが表示されたら、APIC CLI または GUI ユーザ インターフェイスへのログインと同じクレデンシャルを使用してログインします。
読み取り専用アカウントを使用できます。

## Visore のブラウザ ページ

### [Filter] 領域

フィルタ形式は大文字と小文字が区別されます。 この領域では、すべての単純な APIC REST API クエリー操作がサポートされます。

| 名前 | 説明 |
|---|---|
| [Class or DN] フィールド | 管理対象オブジェクトのオブジェクト クラス名または完全な識別名。 |

| 名前 | 説明 |
|---|---|
| [Property] フィールド | 結果をフィルタリングする管理対象オブジェクトのプロパティ。[Property] フィールドを空のままにすると、検索では特定のクラスのインスタンスすべてが返されます。 |
| [Op] ドロップダウン リスト | 結果をフィルタリングするプロパティの値の演算子。有効な演算子は次のとおりです。<br>• ==（等しい）<br>• !=（等しくない）<br>• <（より小さい）<br>• >（より大きい）<br>• ≤<=（以下）<br>• >=（以上）<br>• 間<br>• ワイルドカード<br>• anybit<br>• allbits |
| [Val1] フィールド | フィルタリングするプロパティの初期値。 |
| [Val2] フィールド | フィルタリングする 2 番目の値。 |

### [Display XML of Last Query] リンク

[Display XML of last query] リンクでは、Visore で実行された最も最近のクエリーの完全な APIC REST API 変換が表示されます。

### [Results] 領域

クエリーはURLで符号化されるため、ブラウザにクエリー結果のページをブックマークして再度表示できます。

（注） 管理対象オブジェクトの多くは内部でのみ使用され、APIC REST APIのプログラム開発には通常適用できません。

| 名前 | 説明 |
|---|---|
| 桃色の背景 | 個別の管理対象オブジェクトのインスタンスを切り離し、その下のオブジェクトのクラス名を表示します。 |
| 青色および緑色の背景 | 管理対象オブジェクトのプロパティ名を示します。 |
| 黄色およびベージュ色の背景 | プロパティ名の値を示します。 |
| [dn] プロパティ | オブジェクト モデルの各管理対象オブジェクトの絶対アドレス。 |
| [dn] リンク | クリックすると、その dn のすべての管理対象オブジェクトが表示されます。 |
| [Class name] リンク | クリックすると、そのクラスのすべての管理対象オブジェクトが表示されます。 |
| ← | クリックすると、管理対象オブジェクトの親オブジェクトに移動します。 |
| → | クリックすると、管理対象オブジェクトの子オブジェクトに移動します。 |
| 疑問符 | 管理対象オブジェクトの XML API ドキュメントにリンクします。 |

# Visore でのクエリーの実行

### 手順

**ステップ 1** [Class or DN] テキスト ボックスに MO のクラスまたは DN 名を入力します。

**ステップ 2** （任意） [Property] テキスト ボックスに MO のプロパティを入力し、[Op] テキスト ボックスに演算子を入力し、[Val1] および [Val2] テキスト ボックスに 1 個または 2 個の値を入力することでクエリーをフィルタリングすることができます。

**ステップ 3** [Run Query] をクリックします。
Visore によってクエリーが APIC に送信され、要求された MO が表形式で表示されます。

**ステップ 4** （任意） クエリーを実行した API コールを表示するには、[Display URI of last query] リンクをクリックします。

**ステップ 5** （任意） クエリーからの API 応答データ構造を表示するには、[Display last response] リンクをクリックします。

**ステップ 6** （任意） 表示された MO の親および子クラスを取得するには、MO 説明テーブルの [dn] フィールドで [<] および [>] アイコンをクリックします。
[>] をクリックすると、MO の子用のクエリーが APIC に送信されます。 [<] をクリックすると、MO の親用のクエリーが送信されます。

**ステップ 7** （任意） MO の統計情報、障害、または動作状態情報を表示するには、MO 説明テーブルの [dn] フィールドで追加のアイコンをクリックします。

# API のテスト

## ブラウザのアドオンを使用した API のテスト

### ブラウザの使用

API 要求をテストするために、ブラウザのアドオン ユーティリティを使用して HTTP メッセージを構築し、それを送信し、応答を検査できます。 最も一般的なブラウザのアドオンとして利用可能な RESTful API クライアントでは、API との対話に使いやすいインターフェイスが提供されます。 クライアントには次のものが含まれます。

- Firefox/Mozilla 用：Poster、RESTClient
- Chrome 用：高度な REST クライアント、Postman

ブラウザのアドオンでは、ペイロード データ構造にトークンを含める必要がないように、セッション トークンが Cookie として渡されます。

## cURL による API のテスト

URL 構文を使用してファイルを転送するツールである cURL を使用して、コンソールまたはコマンドライン スクリプトから API メッセージを送信できます。

POST メッセージを送信するには、JSON または XML コマンドの本文を含むファイルを作成し、次の形式で cURL コマンドを入力します。

```
curl -X POST --data "@<filename>" <URI>
```

ディスクリプタ ファイルの名前と API 操作の URI を指定する必要があります。

（注） ディスクリプタ ファイル名の前に「@」記号を必ず入力してください。

次に、ファイル「newtenant.json」で JSON データ構造を使用して、ExampleCorp という名前の新しいテナントを作成する例を示します。

```
curl -X POST --data "@newtenant.json" https://192.0.20.123/api/mo/uni/tn-ExampleCorp.json
```

Get メッセージを送信するには、次の形式で cURL コマンドを入力します。

```
curl -X GET <URI>
```

次に、JSON 形式でテナントに関する情報を読み取る例を示します。

```
curl -X GET https://192.0.20.123/api/mo/uni/tn-ExampleCorp.json
```

（注） cURL でテストするときは、API にログインし、認証トークンを保存し、トークンを後続の API 操作に含める必要があります。

### 関連トピック

例：cURL による JSON API を使用したユーザの追加

# Python による API のテスト

Python 要求モジュールを使用して、Python プログラムから API メッセージを送信できます。

次に、API にログインし、認証トークンを保存し、センサーを読み込む例を示します。

```
import json
import requests

base_url = 'https://192.0.20.123/api/'

# create credentials structure
name_pwd = {'aaaUser': {'attributes': {'name': 'georgewa', 'pwd': 'pa55word'}}}
json_credentials = json.dumps(name_pwd)

# log in to API
login_url = base_url + 'aaaLogin.json'
post_response = requests.post(login_url, data=json_credentials)

# get token from login response structure
auth = json.loads(post_response.text)
login_attributes = auth['imdata'][0]['aaaLogin']['attributes']
auth_token = login_attributes['token']

# create cookie array from token
cookies = {}
cookies['APIC-Cookie'] = auth_token

# read a sensor, incorporating token in request
sensor_url = base_url + 'mo/topology/pod-1/node-1/sys/ch/bslot/board/sensor-3.json'
get_response = requests.get(sensor_url, cookies=cookies)

# display sensor data structure
```

```
print get_response.json()
```

この例では、セッションの認証トークン（Cookie）を管理する必要があります。 Python 要求モジュールには、トークンを自動的に管理する Session() メソッドが含まれます。

Python 要求モジュールの詳細については、http://www.python-requests.org を参照してください。