cockpit
IT Service Manager

Supervision - Configuration du contrôle "Requête HTTP API (XML - JSON attendu)

Document FAQ

Table des matières

Introduction

I. Objectif

- Présenter le fonctionnement et le paramétrage du contrôle « Web - Requête HTTP API (XML - JSON attendu) ».

II. Pré-requis

- Il est préférable d'avoir des notions en XML, Xpath, JSON et JSONPath.
- Il est conseillé, lors du paramétrage du contrôle, de s'appuyer sur un utilitaire permettant d'effectuer des requêtes API (ex : Insomnia, Postman).
  Dans ce document nous utilisons pour les exemples « Insomnia ».

I. Paramètres génériques

- Disponibilité équipement : Oui
- Statut inversé : Non
- Graphique : Oui
- Double seuil : Non
- Disponible dans les rapports : Oui (Supervision)

II. Alertes de configuration

- Problèmes dans le paramétrage du contrôle, exemples :
  - 1006 - Échec / ProtocolException: Target host is not specified => vérifier le champ « Adresse (URL) » du contrôle.
  - 1000 - Erreur interne / Internal server error executing check => vérifier les champs de l'authentification (Secret, Utilisateur, scope, etc.)
- Problèmes d'interprétation du résultat, exemple :
  - 1006 - Échec / Cannot parse response, invalid expression => vérifier le format des éléments renvoyés par l'API, s'il s'agit bien de JSON ou de XML.
  - 1002 - Résultat inexploitable => le type de données attendu ne correspond pas.

Paramètres spécifiques

I. Type de requête

Il est possible d'utiliser deux types de requêtes :

- « GET » : Méthode pour récupérer les informations de l'API.
- « POST » : Méthode pour transmettre des données à l'API.

Des éléments peuvent être postés lors de l'exécution des requêtes (GET ou POST) :

- Dans le champ « Type de média », sélectionner le format des éléments à poster :
  - « text/plain »
  - « application/json »
  - « application/x-www-form-urlencoded »
  - « application/xml »
- Dans le champ « Éléments à poster », renseigner les éléments à envoyer lors de l'exécution de la requête.

  Exemple avec l'API publique de Cockpit ITSM, pour récupérer la liste des équipements, il est nécessaire d'indiquer – à minima – le nombre d'occurrence par page :

II. Adresse (URL)

URL où la requête est envoyée, l'URL doit être :

- http://*
- https://*

III. Empreinte du certificat SSL

Quand l'URL API contactée a un certificat valide ce champ n'est pas utile et peut être laissé vide.
Dans le cas d'un certificat expiré ou auto-signé, renseigner l'empreinte SHA-1 du certificat SSL.
Les séparateurs « » ou « : » entre les paires de caractères sont acceptés.

Authentification et Entêtes

I. Mode d'authentification

Trois types d'authentification sont possibles.

A. HTTP - Basique

Le mode d'authentification « HTTP - Basique » est la forme la plus simple d'authentification, seuls un nom d'utilisateur et un mot de passe sont requis.
Ce mode de connexion peut être utilisé pour des connexions aux APIs ne requérant aucune authentification, il suffit dans ce cas de laisser les champs « Utilisateur » et « Mot de passe » vides.

Champs	Description
Utilisateur	Identifiant de l'utilisateur
Mot de passe	Mot de passe de l'utilisateur

B. OAuth2 - Identifiants du propriétaire de la ressource

Processus d'authentification avec jetons faisant intervenir 3 acteurs : le client, le serveur d'autorisation et le propriétaire de la ressource.
Ce type d'authentification est utilisée pour les connexions à l'API publique du portail Cockpit ITSM, pour plus de détails se reporter aux documents FAQ suivants :
https://faq.cockpit-itsm.com/COCKPIT_FAQ_ADMINISTRATION_api_connection_fr.pdf
https://support.cockpit-itsm.com/apidocs/index.html#/

Champs	Description
« Scope »	Périmètre définissant le type d'accès ou permissions aux données
Utilisateur	Identifiant de l'utilisateur
Mot de passe	Mot de passe de l'utilisateur
URL jeton d'accès	URL de génération de jetons (token) d'accès
ID client	Identifiant du client
Secret client	Mot de passe du client

C. OAuth2 - Identifiants client

Processus d'authentification avec jetons faisant intervenir 2 acteurs : le client et serveur d'autorisation.

Champs	Description
« Scope »	Périmètre définissant le type d'accès ou permissions aux données

URL jeton d'accès	URL de génération de jetons (token) d'accès
ID client	Identifiant du client
Secret client	Mot de passe du client

II. Entêtes

Ajouter des entêtes en cliquant sur le bouton « Plus ».

Chaque entête est composé d'un nom et d'une valeur.

Filtre

Le but de la partie « Filtre » est de renseigner une expression XPath ou JSONPath afin de rechercher une valeur dans les données renvoyées par l'API.

I. Type de réponse

Sélectionner le type de réponse renvoyée par l'API, le contrôle peut traiter des réponses au format XML ou JSON.

II. Expression

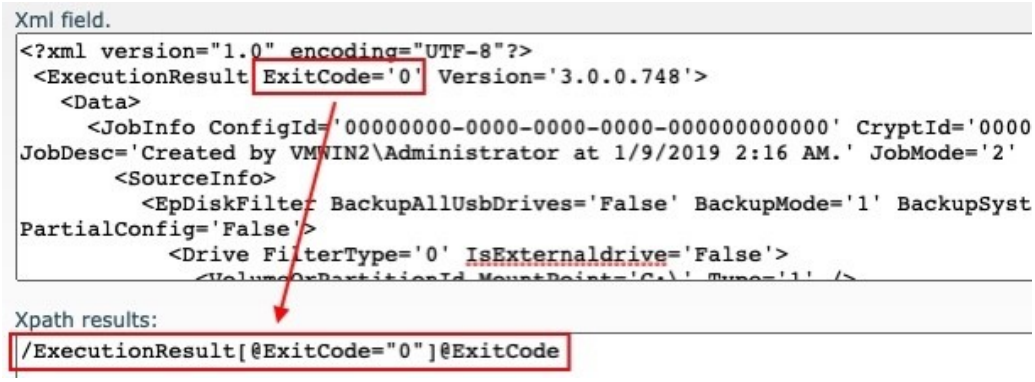
Ce champ est obligatoire, il permet de rechercher un élément dans la réponse afin de l'évaluer au seuil d'alerte.

A. XPath

1. Principes

Quand le type de réponse renvoyé par l'API est du XML, il faut rechercher les éléments en utilisant le langage XPath.
Ci-dessous un tutoriel traitant du XPath :
https://www.w3schools.com/xml/xpath_intro.asp

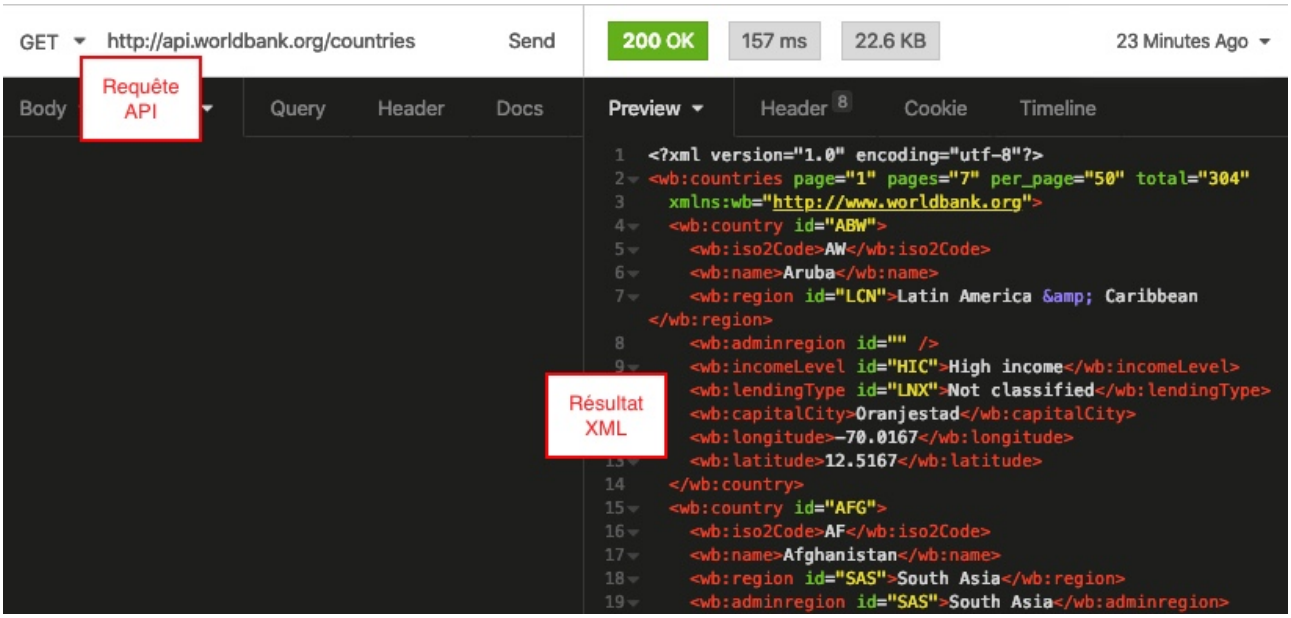
Le site suivant permet de renseigner la réponse de l'API au format XML dans un champ, puis de cliquer sur l'élément que l'on souhaite relever, l'expression XPath est alors générée. Cette expression peut être utilisée dans le contrôle.
https://xmltoolbox.appspot.com/xpath_generator.html
Exemple avec le résultat d'un backup VEEAM :



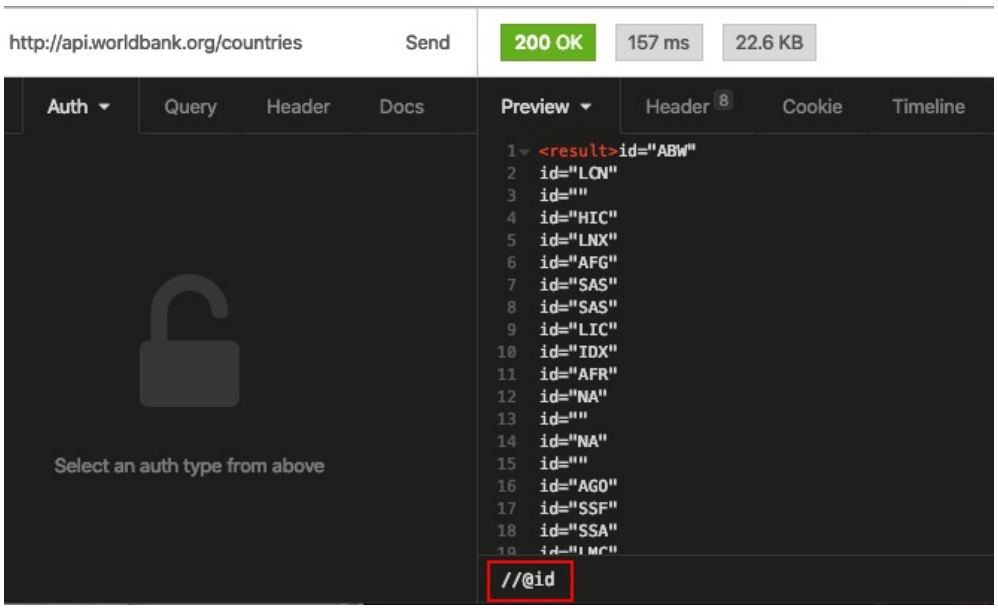
L'expression XPath va chercher la valeur de l'attribut « ExitCode ».

Il est également possible de rechercher le résultat de la requête API via un utilitaire de type « Insomnia ».

Dans l'exemple ci-dessous le résultat de la requête à l'API http://api.worldbank.org/countries est affiché en XML :



Tester ensuite l'expression XPath que l'on souhaite utiliser :



Dans cet exemple l'expression « //@id » remonte tous les attributs « id », le seuil d'alerte peut être : Est-ce que la réponse alphanumérique contient « AFG » ou un autre « id ».

2. Exemple

Ci-dessous un exemple de format XML avec quelques commandes décrites dans le tableau suivant :

```
<?xml version="1.0" encoding="UTF-8" ?>
<store>
 <book>
  <category>reference</category>
  <author>Nigel Rees</author>
  <title>Sayings of the Century</title>
```

```
  <price>8.95</price>
 </book>
 <book>
  <category>fiction</category>
  <author>Evelyn Waugh</author>
  <title>Sword of Honour</title>
  <price>12.99</price>
 </book>
 <book>
  <category>fiction</category>
  <author>Herman Melville</author>
  <title>Moby Dick</title>
  <isbn>0-553-21311-3</isbn>
  <price>8.99</price>
 </book>
 <book>
  <category>fiction</category>
  <author>J. R. R. Tolkien</author>
  <title>The Lord of the Rings</title>
  <isbn>0-395-19395-8</isbn>
  <price>22.99</price>
 </book>
 <bicycle>
  <color>red</color>
  <price>19.95</price>
 </bicycle>
</store>
```

Exemples d'expressions XPath avec le XML précédent :

Expression XPath	Résultat
/store/book/author	Tous les éléments « Author » de la hiérarchie « /store/book ».
//author	Tous les éléments « author » quelque soit leur niveau dans la hiérarchie.
/store/*	Tout ce qui se trouve dans « store ».
/store//price	Tous les éléments « price » qui se trouvent dans « store ».
//book[3]	Le 3ème élément « book » par ordre d'apparition.
//book[last()]	Le dernier élément « book » par ordre d'apparition.
//book[position()<3]	Les 2 premiers éléments « book ».
//book[isbn]	Tous les éléments « book » possédant l'attribut « isbn ».
//book[price<10]	Tous les éléments « book » dont l'attribut « price » est inférieur à « 10 ».
//*	Tous les éléments du document XML.

B. JSONPath

1. Principes

Quand le type de réponse renvoyé par l'API est du JSON, il faut rechercher les éléments en utilisant le langage JSONPath.

JSONPath est un langage permettant d'interroger JSON, un peu comme XPath le fait pour XML.

Une expression JSONPath spécifie un chemin vers un élément ou un ensemble d'éléments dans une structure JSON.

Les expressions JSONPath sont sensibles à la casse.

Ci-dessous quelques éléments syntaxiques de JSONPath :

Élément syntaxique	Description
$	Placé en début d'expression, le caractère « $ » représente l'objet racine ou le tableau (dans le cas où le JSON n'a d'élément racine mais plusieurs tableaux). Le caractère « $ » peut être omis, exemple : $.foo.bar équivaut à foo.bar
@	On utilise « @ » dans les filtres des expressions pour faire référence au nœud courant en cours de traitement.
. **ou** **[' ']**	Opérateurs servant à spécifier le chemin vers l'élément recherché. Les chemins peuvent utiliser la notation par points ou par crochets, les 2 expressions suivantes sont similaires : $.store.book[0].title équivaut à $['store']['book'][0]['title'] Note : utiliser la notation avec crochets si le nom de la propriétés contient des caractères spéciaux ou des espaces, ou commence par un caractère autre que A...Z / a...z.
..	Opérateur récursif descendant. Recherche récursivement le nom de la propriété spécifiée et renvoie un tableau de toutes les valeurs de ce nom de propriété.
*****	Sélectionne tous les éléments d'un objet ou d'un tableau (wildcard).
[n]	Sélectionne le énième élément d'un tableau, la numérotation commence à « 0 ».
[: *n*]	Sélectionne les « n » premiers éléments d'un tableau.
[*-n* :]	Sélectionne les « n » derniers éléments d'un tableau.
[*index1* , *index2* , …]	Sélectionne les éléments du tableau indiqués.
[*start* : *end*] **[*start* :]**	Sélectionne les éléments du tableau à partir de l'index de début et jusqu'à l'index de fin, mais sans les inclure. Si la fin est omise, sélectionne tous les éléments du début jusqu'à la fin du tableau.
[?(*expression*)]	Utilisation d'un filtre.

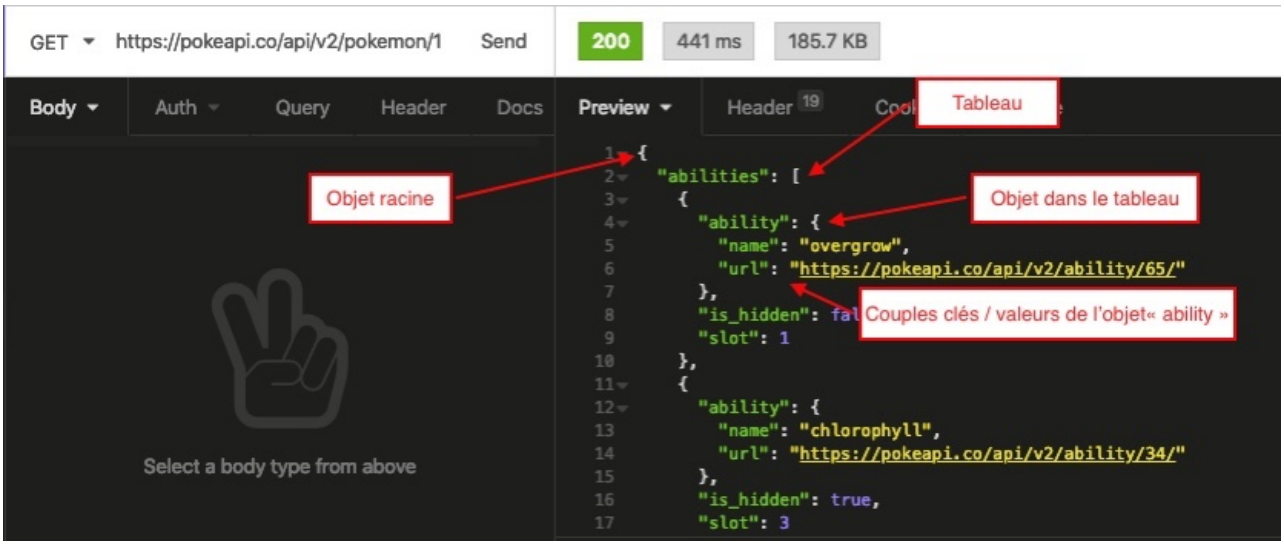
	Le filtre sélectionne tous les éléments d'un objet ou d'un tableau qui correspondent au filtre spécifié.
[(*expression*)]	Expression de script.
==	Opérateur « Égale à ». 1 et « 1 » sont considérés comme égaux. Les valeurs des chaînes doivent être mises entre quotes et non entre guillemets doubles : [?(@.color=='red')].
!=	Opérateur « Différent de ». Les valeurs des chaînes doivent être mises entre simple quote.
>	Opérateur « Supérieur à ».
>=	Opérateur « Supérieur ou égal à ».
<	Opérateur « Inférieur à ».
<=	Opérateur « Inférieur ou égal à ».
=~	Fait appel à une expression régulière de type JavaScript. Exemple : $..[?(@.author =~ /Nigel.*/i)] Recherche tous les éléments dont le champ « author » contient la valeur « Nigel ».
!	Correspond à « Ne contient pas ». Exemple : $..[?(!@.isbn)] Recherche tous les éléments qui ne possèdent pas le champ « isbn ».
&&	Opérateur « ET » permettant de combiner plusieurs expressions. Exemple : $..[?(@.category=='fiction' && @.price < 10)] Recherche tous les éléments dont le champ « category » a la valeur « fiction » ET dont le champ « price » est inférieur à la valeur « 10 ».
‖	Opérateur « OU » permettant de combiner plusieurs expressions. Exemple : $..[?(@.category=='fiction' || @.price < 10)] Recherche tous les éléments dont le champ « category » a la valeur « fiction » ET dont le champ « price » est inférieur à la valeur « 10 ».

2. Hiérarchie

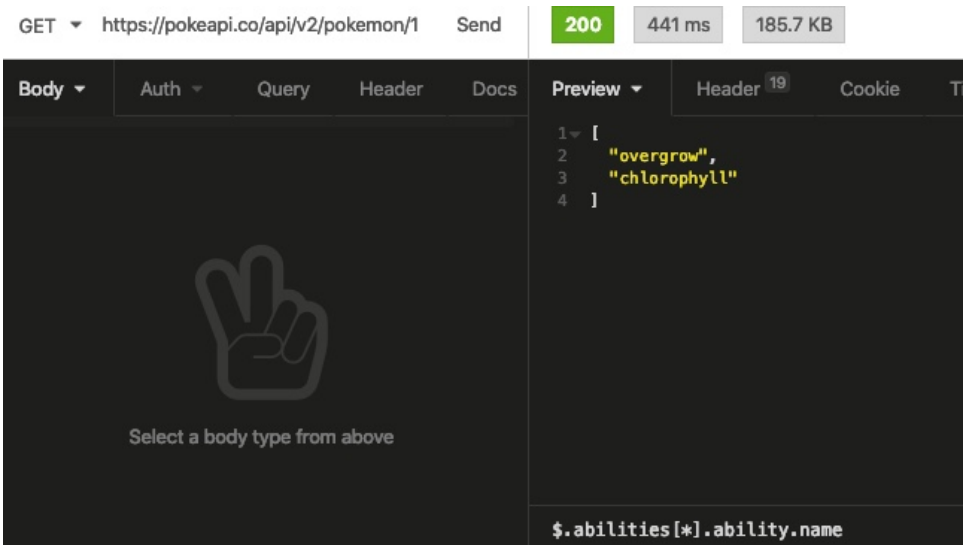
Pour indiquer le chemin d'un élément en JSONPath, il faut reconnaître les différents éléments suivants de la structure JSON :

- {…} : les accolades définissent un objet. Il peut avoir une hiérarchie entre les objets.
- […] : les crochets définissent un tableau, dans l'expression JSONPath, quand on arrive sur un tableau, il est nécessaire d'indiquer la ligne du tableau recherchée. Exemples :
  - [0] => Première ligne
  - [1] => Deuxième ligne
  - [*] => Toutes les lignes
  - etc.

Par exemple dans l'API suivante :



L'expression permettant de rechercher les champs « name » sera : $.abilities[*].ability.name



Important l'expression « $.abilities.ability.name » ne remontera rien.

3. Outil

Le site suivant permet de renseigner la réponse de l'API au format JSON dans un champ, de renseigner l'expression JSONPath que l'on souhaite utiliser, de vérifier la valeur renvoyée par l'expression, l'expression peut être ensuite utilisée dans le contrôle :

http://jsonpath.herokuapp.com/

Exemple :

L'expression JSONPath va chercher les valeurs numériques de tous les éléments « price » inférieures à « 10 ».

4. Exemples

Ci-dessous nous reprenons l'exemple précédemment utilisé pour le format XML, mais cette fois-ci au format les données sont au format JSON :

```
{ "store": {
    "book": [
     { "category": "reference",
       "author": "Nigel Rees",
       "title": "Sayings of the Century",
       "price": 8.95
     },
     { "category": "fiction",
       "author": "Evelyn Waugh",
       "title": "Sword of Honour",
       "price": 12.99
     },
     { "category": "fiction",
       "author": "Herman Melville",
       "title": "Moby Dick",
```

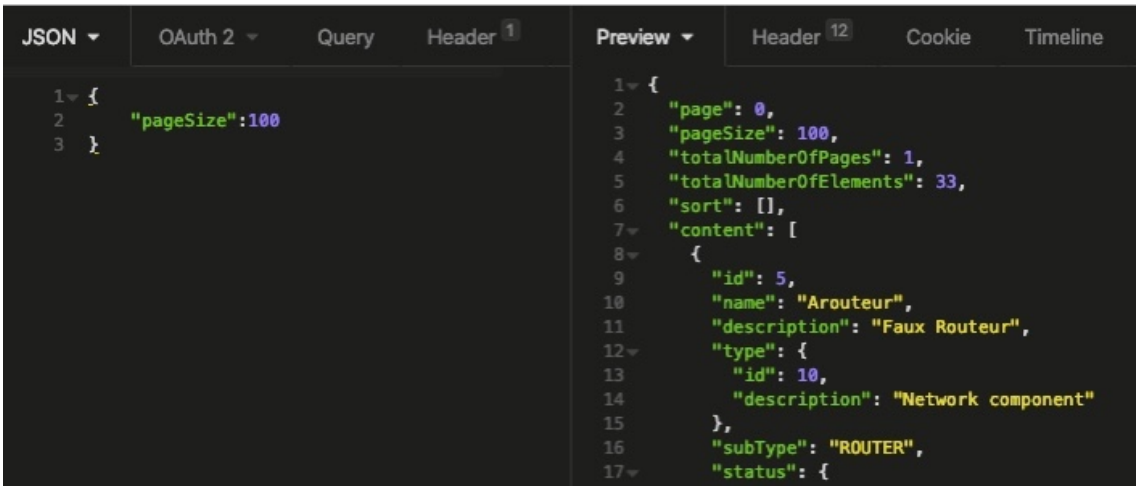
```
      "isbn": "0-553-21311-3",
      "price": 8.99
    },
    { "category": "fiction",
      "author": "J. R. R. Tolkien",
      "title": "The Lord of the Rings",
      "isbn": "0-395-19395-8",
      "price": 22.99
    }
  ],
  "bicycle": {
    "color": "red",
    "price": 19.95
  }
 }
}
```


Exemples d'expression JSONPath avec le JSON précédent :

Expression XPath	Résultat
$.store.book[*].author	Tous les éléments « Author » de la hiérarchie « /store/book ».
$..author	Tous les éléments « author » quelque soit leur niveau dans la hiérarchie.
$.store.*	Tout ce qui se trouve dans « store ».
$.store.bycicle.color	Remonte la couleur du vélo présent dans « store ».
$.store..price **$..price**	Tous les éléments « price » qui se trouvent dans « store ».
$..book[2]	Le 3ème élément « book » par ordre d'apparition, le premier ayant la valeur « 0 ».
$..book[-1:]	Le dernier élément « book » par ordre d'apparition.
$..book[0,1] **$..book[:2]**	Les 2 premiers éléments « book ».
$..book[?(@.author=='J. R. R. Tolkien')].title	Affiche les valeurs des éléments « title » dont le champ « author » est « J. R. R. Tolkien ».
$..book[?(@.isbn)]	Tous les éléments « book » possédant l'élément « isbn ».
$..book[?(!@.isbn)]	Tous les éléments « book » ne possédant pas l'élément « isbn ».
$..book[?(@.price < 10)]	Tous les éléments « book » dont l'attribut « price » est inférieur à « 10 ».
`$..book[?(@.author =~ /.*Tolkien/i)]`	Tous les éléments « book » dont le champ « author » se termine par « Tolkien » (sensible à la casse).
$..book[?(@.category == 'fiction' || @.category == 'reference')]	Tous les éléments « book » appartenant à la catégorie « Fiction » ou « Reference ».
$..*	Tous les membres de la structure JSON sous la racine (objets enfants, valeurs des propriétés individuelles, éléments de tableau), combinés dans un tableau.

Seuil d'alerte

Pour illustrer les seuils d'alerte nous utilisons l'outil « Insomnia » afin de vérifier les valeurs renvoyées par les APIs et comment les données sont filtrées avec les expressions qui seront renseignées dans les contrôles :



I. Seuil : Existence d'un élément

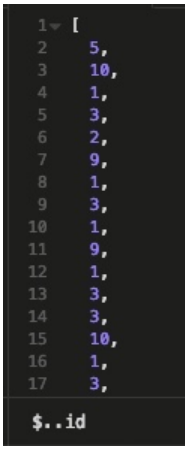
Ce seuil d'alerte évalue si le résultat est « existant » ou « inexistant ».

II. Seuil : Valeur numérique

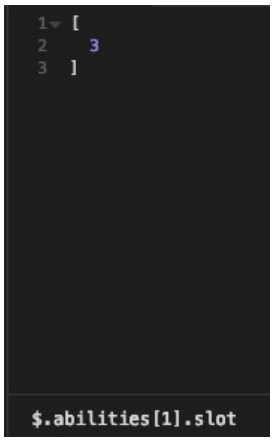
Ce seuil d'alerte compare la valeur relevée par rapport à un seuil avec les opérateurs suivants :

- égale à
- différent de
- supérieur à
- inférieur à

Il est nécessaire de ne récupérer qu'une seule valeur, par exemple si la requête récupère un tableau comme ci-dessous, cela génèrera une alerte de paramétrage :



Pour évaluer un nombre il ne faut en récupérer un seul :



III. Seuil : Valeur alphanumérique

Ce seuil d'alerte vérifie la présence ou l'absence d'une chaîne de caractères avec les opérateurs suivants :

- égale à
- différent de
- contient
- ne contient pas

Pour les opérateurs « Contient » et « Ne contient pas », il est possible d'effectuer le contrôle sur une liste d'éléments. Dans l'exemple ci-dessous, si on cherche les caractères « chloro », ils seront trouvés car présents dans « chlorophyll » :

Pour l'opérateur « Égale à » il est nécessaire de ne remonter qu'une seule valeur et non une liste :

IV. Alertes consécutives et métriques

- Nombre d'exécutions consécutives en dépassement de seuil avant la génération de l'alerte :
  - Renseigner une valeur entre 1 et 99.
  - Si on ne souhaite pas avoir d'alerte, décocher cette option (ce peut être le cas si on souhaite uniquement avoir le relevé des métriques).
- Conserver les résultats (métriques) :
  - Cocher cette option pour obtenir un graphique des valeurs retournées par les exécutions du contrôle.
  - Cette option fonctionne uniquement quand la valeur retournée est de type « Numérique ».

Fin du document