V2 - (READ) /api/v2/checks

Sommaire

Objectifs

Méthode POST de type READ qui permet de récupérer les données de supervision de checks et éventuellement de leur père, comme sur l'Interface de visualisation,

- Filtrées (optionnel)
- Triées
- Rangées :
  - En arbres (hôtes/clusters => checks)
  - Tous au même niveau

Paramètres

Cet appel utilise les 4 paramètres suivants :

- **filterX**
  - Pour la propriété **type**, le fait d'utiliser des valeurs différentes de **check**, **check_host**, et **check_cluster** est interdit sur cette route.
- **sort**
- **output_format**
- **ouput_field**

(*voir la page* V2 - Les paramètres des API du broker-module-livedata)

Réponse

Codes de retour

Codes de retour	Explications
200	OK
400	Paramètre invalide
401	Accès nécessite une authentification ou un Token valide.
403	Authentification de l'utilisateur OK , mais droits non suffisant.
500	L'appel est valide, mais un problème d'exécution est survenu.

Retour du code 200

Les propriétés retournées doivent être choisies avec l'option **output_field**

Mais les propriétés suivantes seront au minimum automatiquement retournées :

- **nb_element**
- **type** (si le paramètre **output_format** vaut *elements_on_same_level)*
- **father_uuid**
- **father_name**
- par check :
  - **check_uuid**
  - **check_name**

(*voir la page* V2 - Les paramètres des API du broker-module-livedata)

Suivant le paramètre **output_format** (*checks_attached_to_father* / *elements_on_same_level*)

- (présentation du format de retour ci-dessus, mais se référer à l'exemple pour le format exact)
- Pour la valeur False, seuls des checks seront présents dans la liste.

checks_attached_to_father:

- **request_statistics** :
  - ...
- **elements_found** :
  - **clusters** :
    - **cluster 1:**
      - **father_uuid** : *text*
      - **father_name** : text
      - ...
      - **checks** :
        - **check_uuid1** : text, **check_name1** : *text*, ...
        - **check_uuid2**: text, **check_name2** : *text*, ...
        - ...
    - ...
  - **hosts** :
    - **host 1:**
      - **father_uuid** : *text*
      - **father_name** : text
      - ...
      - **checks** :
        - **check_uuid1** : text, **check_name1** : *text*, ...
        - **check_uuid2**: text, **check_name2** : *text*, ...
        - ...
    - ...

```
curl -s -S -H 'x-api-token: XYZ' \
-d "output_format=checks_attached_to_father" \
http://broker-module-livedata:50100/api/v2/checks
```

Exemple de sortie attendue :

elements_on_same_level :

- **request_statistics** :
  - ...
- **elements_found :**
  - **nb_element:** *X*
  - **elem1:**
    - **type: check**
    - **check_uuid** : *text*
    - **check_name:** *text*
    - **father_name :** *text*
    - **father_uiid :** *text*
    - ...
  - **elem2:**
    - **type: check**
    - **check_uuid** : *text*
    - **check_name:** *text*
    - **father_name :** *text*
    - **father_uiid :** *text*
    - ...
- ...

```
curl -s -S -H 'x-api-token: XYZ' \
-d "output_format=elements_on_same_level" \
http://broker-module-livedata:50100/api/v2/checks
```

Exemple de sortie attendue :

http://localhost:50100/api/v2/checks/


```
{
  "request_statistics": {
    "nb_elements_total": 9,
    "nb_checks_total": 4,
    "nb_elements_filtered": 4,
    "nb_checks_filtered": 4
  },
  "elements_found": {
    "clusters": [
      {
        "father_name": "datacenter bdx",
        "father_uuid":
"d6921ee8ba1511eba36c0800277faebe",
        "checks": [
          {
            "check_name": "System Uptime",
            "check_uuid":
"d6921ee8ba1511eba36c0800277faebe-
e6daad4cba1511eb95980800277faebe"
          }
        ]
      }
    ],
    "hosts": [
      {
        "father_name": "Bordeaux",
        "father_uuid":
"2c6dcf1aba1611ebaa7d0800277faebe",
        "checks": [
          {
            "check_name": "CPU Stats",
            "check_uuid":
"2c6dcf1aba1611ebaa7d0800277faebe-
c296d75e5ad911e58cc5080027f08538"
          },
          {
            "check_name": "Disks Stats",
            "check_uuid":
"2c6dcf1aba1611ebaa7d0800277faebe-
c29735965ad911e58cc5080027f08538"
          }
        ]
      },
      {
        "father_name": "Nantes",
        "father_uuid":
"76f45d80ba1e11eba2670800277faebe",
        "checks": [
          {
            "check_name": "CPU Stats",
            "check_uuid":
"76f45d80ba1e11eba2670800277faebe-
c296d75e5ad911e58cc5080027f08538"
          }
        ]
      }
    ]
  }
}
```


http://localhost:50100/api/v2/checks/


```
{
  "request_statistics": {
    "nb_elements_total": 9,
    "nb_checks_total": 4,
    "nb_elements_filtered": 4,
    "nb_checks_filtered": 4
  },
  "elements_found": [
    {
      "check_name": "CPU Stats",
      "check_uuid":
"2c6dcf1aba1611ebaa7d0800277faebe-
c296d75e5ad911e58cc5080027f08538",
      "father_name": "Bordeaux",
      "father_uuid":
"2c6dcf1aba1611ebaa7d0800277faebe",
      "type": "check_host"
    },
    {
      "check_name": "Disks Stats",
      "check_uuid":
"2c6dcf1aba1611ebaa7d0800277faebe-
c29735965ad911e58cc5080027f08538",
      "father_name": "Bordeaux",
      "father_uuid":
"2c6dcf1aba1611ebaa7d0800277faebe",
      "type": "check_host"
    },
    {
      "check_name": "System Uptime",
      "check_uuid":
"d6921ee8ba1511eba36c0800277faebe-
e6daad4cba1511eb95980800277faebe",
      "father_name": "datacenter bdx",
      "father_uuid":
"d6921ee8ba1511eba36c0800277faebe",
      "type": "check_cluster"
    },
    {
      "check_name": "CPU Stats",
      "check_uuid":
"76f45d80ba1e11eba2670800277faebe-
c296d75e5ad911e58cc5080027f08538",
      "father_name": "Nantes",
      "father_uuid":
"76f45d80ba1e11eba2670800277faebe",
      "type": "check_host"
    }
  ]
}
```


Retour du code 400

Paramètres POST incorrects

Paramètre inconnu

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "parametre_inconnu=is_status_:true" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: POST parameter [ parametre_inconnu ] is
unknown
```

```
ERROR 400: POST parameter [ parametre_inconnu ] is unknown
```

Messages d'erreurs des filtres (filterX)

(*voir la page V2 - Les paramètres des API du broker-module-livedata*)

Filtre inexistant

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "filter01=is_status_:true" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: filtering[0]: invalid field name [
is_status_ ]
```

```
ERROR 400: filtering[0]: invalid field name [ is_status_ ]
```

Filtre incomplet

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "filter0=next_check" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: filtering[0]: missing value for field [
next_check ]
```

```
ERROR 400: filtering[0]: missing value for field [ next_check ]
```

Filtre incorrect

Valeur incorrecte pour ce type de filtre

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "filter0=status:9" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: filtering[0]: field [ status ] => wrong
value ['9']
```

```
ERROR 400: filtering[0]: field [ status ] => wrong value ['9']
```

Opérateur incorrect

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "filter0=next_check:avant" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: filtering[0]: field [ next_check ]
unknown date constraint [ avant ]
```

```
ERROR 400: filtering[0]: field [ next_check ] unknown date constraint
[ avant ]
```

Argument incorrect

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "filter0=next_check:in-less-than|hier" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: filtering[0]: field [ next_check ] => [
in-less-than ] => invalid literal for int() with
base 10: 'hier'
```

```
ERROR 400: filtering[0]: field [ next_check ] => [ in-less-than ] =>
invalid literal for int() with base 10: 'hier'
```

Messages d'erreurs liés aux paramètres de tri (sort)

(*voir la page* V2 - Les paramètres des API du broker-module-livedata)

Propriété inconnu

```
ERROR 400: sort: invalid field name [ host ]
```

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "sort=host" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: sort: invalid field name [ host ]
```

Ordre de tri incorrect

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "sort=father_name:big" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: sort: invalid sort direction [ big ]
for field [ father_name ]
```

ERROR 400: sort: invalid sort direction [big] for field [father_name]

Messages d'erreurs liés au paramètre de format du résultat (output_format)

(*voir la page* V2 - Les paramètres des API du broker-module-livedata)

Valeur incorrecte

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "output_format=vrai" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: output_format: invalid value [ vrai ]
```

ERROR 400: output_format: invalid value [vrai]

Messages d'erreurs lors du paramétrage des propriétés présentes dans la sortie (output_field)

(*voir la page* V2 - Les paramètres des API du broker-module-livedata)

Propriété de sortie inexistante

```
$ curl -s -S -H "x-api-token: XYZ" \
-d "output_field=is_status_" \
http://broker-module-livedata:50100/api/v2/checks
ERROR 400: output_field: invalid field name [
is_status_ ]
```

ERROR 400: output_field: invalid field name [is_status_]

Erreurs communes lors de l'envoi de la requête

Messages d'erreurs liés au protocole HTTPS

Le certificat SSL a été refusé

```
$ curl -s -S -H "x-api-token: XYZ" \
https://broker-module-livedata:50100/api/v2/inventory
curl: (60) SSL certificate problem: unable to get
local issuer certificate
…
```

Cela peut se produire si le certificat du serveur est auto-signé.

Pour contourner la validité de l'émetteur du certificat, il faut utiliser l'option **--insecure** (*option courte : **-k***).

```
curl --insecure -s -S -H "x-api-token: XYZ" \
https://broker-module-livedata:50100/api/v2
/inventory
```

Avec curl v7.29.0 :

```
curl: (60) Peer's Certificate issuer is not recognized.

…
```

Avec curl v7.60 et supérieur :

```
curl: (60) SSL certificate problem: unable to get local issuer certificate

…
```

Requête HTTP sur un serveur en HTTPS

```
$ curl -s -S -H "x-api-token: XYZ" \
http://broker-module-livedata:50100/api/v2/inventory
The client IP_SERVEUR:PORT_CLIENT sent a plain HTTP
request,
but this server only speaks HTTPS on this port.
```

```
The client IP_SERVEUR:PORT_CLIENT sent a plain HTTP request,
but this server only speaks HTTPS on this port.
```

Requête HTTPS sur un serveur en HTTP

```
$ curl -s -S -H "x-api-token: XYZ" \
https://broker-module-livedata:50100/api/v2/inventory
curl: (35) error:1408F10B:SSL routines:
ssl3_get_record:wrong version number
```

Avec curl v7.29.0 :

```
curl: (35) SSL received a record that exceeded the maximum
permissible length.
```

Avec curl v7.60 et supérieur :

```
curl: (35) error:1408F10B:SSL routines:ssl3_get_record:wrong version
number
```