Passerelle - Development #34178

jsonschema : ajout d'un petit DSL de transformation de données

19 juin 2019 17:37 - Emmanuel Cazenave

Statut:	Rejeté	**Début:**	19 juin 2019
Priorité:	Normal	**Echéance:**	
Assigné à:		**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Sur un endpoint qui a beaucoup de paramètres en entrées, un long schéma associé, ça implique souvent de longues lignes ennuyeuses dans le endpoint :

```
demande_number = self._soap_call(
    wsdl='DemandeService', method='insertDemandeByType',
    contactNom=post_data['contact_nom'],
    contactTelephone=post_data['contact_telephone'],
    contactCourriel=post_data['contact_email'],
    contactAdresse=post_data['contact_adresse'], demandeObjet=post_data['demande_objet'],
    demandeLieu=post_data['demande_lieu'],
    demandeDescription=post_data['demande_description'],
    remoteAddress=post_data['remote_adresse'], codeEquipement=post_data['code_equipement'],
    codeServiceDemandeur=post_data['code_service_demandeur'],
    dateSouhaitee=post_data['date_souhaite'], typeDemande=post_data['type_demande']
```

On pourrait rajouter dans le schéma json une propriété translate :

```
INSERT_DEMANDE_COMPLET_BY_TYPE = {
    '$schema': 'http://json-schema.org/draft-03/schema#',
    'type': 'object',
    'properties': {
        'contact_nom': {
            'description': 'Nom du contact',
            'required': True,
            'translate': 'ContactNom'
        },
        'contact_tel': {
            'description': 'Téléphone du contact',
            'type': 'string',
            'translate': 'PhoneNum'
        },
```

Et dans le get_params de passerelle.views.py générer un translated_post_data en plus du post_data, qui mâcherait le travail coté endpoint.

Historique

#1 - 19 juin 2019 18:14 - Benjamin Dauvergne

Il me semble que jsonschema (la lib, la spéc le supporte aussi je suppose) supporte l'extension du schéma de base mais faut voir comment ça marche.

#2 - 19 juin 2019 18:59 - Emmanuel Cazenave

- Fichier 0001-misc-add-parameters-translation-capabilities-34178.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

Pas regardé la spec mais je pensais à quelque chose de sauvage comme ça.

python-jsonschema ne râle pas et quand bien même ce serait facile de lui passer un schéma propre.

#3 - 19 juin 2019 19:05 - Frédéric Péters

rename plutôt que translate.

#4 - 20 juin 2019 12:29 - Emmanuel Cazenave

- Fichier 0001-misc-add-parameters-rename-capabilities-34178.patch ajouté

rename donc.

#5 - 20 juin 2019 14:20 - Thomas Noël

Alors moi je serais un peu plus direct, je renommerai directement les clés dans post_data, ie sans besoin d'un renamed_post_data. Après tout, quand on ajoute un "rename" dans le schéma, c'est qu'on sait ce qu'on veut... non ?

#6 - 20 juin 2019 14:44 - Emmanuel Cazenave

Pensé à ça aussi, mon cœur balance un peu mais penche pour un renamed_post_data.

Genre je vais pas forcément définir un 'rename' sur tous les paramètres,m juste sur ceux que je veux passer tel quel. Mais le rename_post_data je veux pouvoir l'enrichir avec d'autres trucs que j'aurai calculé sur la base du post_data. Et si tout est tout mélangé dans l'unique post_data, ça devient le bordel je pense.

#7 - 20 juin 2019 15:06 - Emmanuel Cazenave

- Assigné à mis à Emmanuel Cazenave

#8 - 20 juin 2019 15:48 - Thomas Noël

- Assigné à Emmanuel Cazenave supprimé

Emmanuel Cazenave a écrit :

> si tout est tout mélangé dans l'unique post_data, ça devient le bordel je pense.

Je comprends, de mon point de vue c'est le rename_post_data qui est un premier bordel, auquel tu imagines ensuite ajouter un transform_post_data et autres ?

Je trouverais plus joli que le schema permette de clairement dire "voilà les données que j'attends, valide les, et voilà comment tu dois me les renvoyer (renommées, transformées, calculées...)" (ce patch étant la partie "renommées")

Non ?

#9 - 20 juin 2019 16:32 - Emmanuel Cazenave

Thomas Noël a écrit :

> Je comprends, de mon point de vue c'est le rename_post_data qui est un premier bordel, auquel tu imagines ensuite ajouter un transform_post_data et autres ?

J'imagine qu'à un moment on aura besoin de faire des truc comme ça :

```
@endpoint(
     perm='can_access',
     post={
         'description': 'Insert action comment',
         'request_body': {
             'schema': {
                 'application/json': UN_SCHEMA
             }
         }
     }
)
def mon_endpoint(request, post_data, renamed_post_data):
    if post_data['field1'] = 'XXX' and post_data['field2'] == 'YY':
        renamed_post_data['field3'] = 'ZZ'
        self._soap_call(..., renamed_post_data)
```

Avec dans le schéma, pas de renommage défini pour field1 et field2, parce que ce ne sont pas des champs attendus par le webservice métier. Et donc je voudrais ne pas avoir à me soucier de supprimer field1 et field2 de post_data, ce que je devrais faire si on renomme directement dans post_data.

Je voudrais pouvoir balancer renamed_post_data tel quel, ou éventuellement avec des trucs que je lui aurais rajouté.

#10 - 20 juin 2019 16:44 - Thomas Noël

Emmanuel Cazenave a écrit :

> Je voudrais pouvoir balancer renamed_post_data tel quel, ou éventuellement avec des trucs que je lui aurais rajouté.

Ouaip c'est ce que j'avais compris et sans doute que ça colle dans certains cas, mais avoir un truc qui s'appelle "renamed_post_data" et qu'en fait on tripote pour ajouter des choses qui n'ont rien à avoir avec du renommage, je sais pas, je trouve ça moche.

En fait tout se passe comme si "renamed_post_data" était le résultat d'un traitement de "post_data". Avec ce patch, il sera le résultat des renommages, plus tard il pourrait aussi contenir quelques petits calculs déclarés dans le @endpoint, que sais-je... Faudrait donc juste lui trouver un autre nom et je serai heureux, mais comme d'hab j'ai pas d'idée... (validated_post_data, parsed_post_data, processed_data, ... ?)

#11 - 20 juin 2019 19:00 - Emmanuel Cazenave

- Fichier 0001-misc-add-parameters-rename-capabilities-34178.patch ajouté

transformed_post_data

#12 - 20 juin 2019 19:30 - Emmanuel Cazenave

Pour une utilisation concrète, aller voir http://git.entrouvert.org/passerelle.git/tree/passerelle/apps/atal/models.py?h=wip/34175-atal.

C'est bien pratique, je trouve ça uber cool.

#13 - 20 juin 2019 21:05 - Frédéric Péters

Que ça reste post_data, et si jamais d'une manière ou d'une autre il y a intérêt à retomber sur ce qui a réellement été transmis, qu'on stocke ça dans orig_post_data, genre.

> Et donc je voudrais ne pas avoir à me soucier de supprimer field1 et field2 de post_data, [...]

Ajouter la prise en charge d'une clé "transform", qui puisse être une chaine de caractère, "ignore", qui fera que ça soit zappé. En se disant aussi que ça pourra évoluer ensuite pour prendre une fonction.

#14 - 21 juin 2019 10:48 - Emmanuel Cazenave

Frédéric Péters a écrit :

> Que ça reste post_data, et si jamais d'une manière ou d'une autre il y a intérêt à retomber sur ce qui a réellement été transmis, qu'on stocke ça dans orig_post_data, genre.

orig_post_data passé directement au endpoint ou dans post_data lui même ?

#15 - 21 juin 2019 11:04 - Frédéric Péters

Pas dans post_data, sur l'idée que tu veux le transférer tel quel. Mon idée était de juste poser ça sur request mais ça ne me semble pas un gain énorme, le connecteur qui aurait à y accéder, il peut très bien juste faire un json.loads(request.body).

```
-                d['post_data'] = data
+                request.orig_post_data = copy.deepcopy(data)
+                ... ton patch ...
```

mais sur la partie "ton patch", avoir le comportement par défaut comme étant de garder tous les attributs, donc je pense,

```
+                for p_name, p_value in json_schema.get('properties', {}).items():
+                    p_new_name = p_value.get('rename')
+                    if p_new_name and p_name in data:
+                        transformed_post_data[p_new_name] = data[p_name]
```

suivre ça de :

```
elif not p_new_name:
    transformed_post_data[p_name] = data[p_name]
```

Alternativement, inverser et itérer sur le contenu de data, et modifier le dictionnaire "in place" (c'était mon idée initiale, c'est pour ça qu'il y a un copy dans mon premier bout de diff).

#16 - 24 juin 2019 15:23 - Emmanuel Cazenave

- Fichier 0001-misc-add-parameters-transformation-capabilities-3417.patch ajouté

Bilan des courses : on aurait un request.orig_post_data (parce que tout ça pour se retrouver à faire des json.load dans un endpoint ça me chagrine), et ci dessous un exemple du mini DSL qui est supporté :

```
def foo(x):
    return x.upper()

def bar(x, y):
    return x + y

schema = {
    '$schema': 'http://json-schema.org/draft-03/schema#',
    'type': 'object',
    'properties': {
        'contact_nom': {
            'type': 'string',
            'required': True,
            'transform': ('rename', 'ContactNom')
        },
        'contact_tel': {
            'type': 'string',
            'transform': 'ignore'
        },
        'contact_first_name': {
            'type': 'string',
            'transform': foo
        }
        'contact_email': {
            'type': 'string',
            'transform': (bar, '-postfix')
        }
    }
}
```

#17 - 24 juin 2019 15:55 - Benjamin Dauvergne

À mon avis ça démontre l'inutilité de ce ticket, le renommage aurait plus à voir avec le SOAP/REST en sortie non ?

C'est pas plus clair de faire un DSL que pour ça et de ne pas mélanger avec le schéma ?

```
soap_body = {
  'ContactNom': {'@ref': 'contact_nom'},
  ...
}
call_soap(**transform(soap_body, request.post_data))
```

#18 - 25 juin 2019 10:53 - Emmanuel Cazenave

- Sujet changé de jsonschema : traduction de paramètres à jsonschema : ajout d'un petit DSL de transformation de données

- Statut changé de Solution proposée à Nouveau

L'intention première étant de s'éviter des longues lignes de code bateau, je ne verrai pas le bénéfice à troquer ça contre une longue déclaration d'un deuxième schéma.

Cela étant, ce ticket étant totalement non crucial, je met en pause, je reprendrai si le besoin se refait sentir.

#19 - 25 juin 2019 12:01 - Benjamin Dauvergne

Emmanuel Cazenave a écrit :

> L'intention première étant de s'éviter des longues lignes de code bateau, je ne verrai pas le bénéfice à troquer ça contre une longue déclaration d'un deuxième schéma.
>
> Cela étant, ce ticket étant totalement non crucial, je met en pause, je reprendrai si le besoin se refait sentir.

J'ai plusieurs objections :

- faut pas mélanger entrée et sortie, surtout si on a des fois plusieurs sorties (un appel REST/HTTP-RPC -> plusieurs appels SOAP)
- étaler un mapping au milieu d'un autre, ça disperse juste l'information et la rend moins lisible, si le jsonschema est long et qu'on a des transforms parsemé dans le tas et pas dans le même ordre que le schéma XSD du WSDL ça n'apporte rien, juste des nœuds au cerveau quand on relit
- la longueur du code n'a pas d'importance pour moi la clarté de ce qui se passe et l'aide au codage est plus importante (genre si ça pouvait

indiquer qu'on mappe un booléen JSON sur un xs:int en sortie et que donc y a un souci, ce serait grand, ou que soit la clé en sortie soit la clé en entrée n'existent pas, avec un inspect de l'opération SOAP par exemple ou que sais-je, au moins ça amènerait un peu d'assurance que ce qu'on fait a un sens)

- je ne vois pas trop en quoi ça

```
'contat_nom': {
...
   'transform': ('rename', 'ContactNom'),
...
'contact_telephone':
...
   'transform': ('rename', 'ContactTelephone'),
```

c'est plus court que ça

```
demande_number = self._soap_call(
    wsdl='DemandeService', method='insertDemandeByType',
    contactNom=post_data['contact_nom'],
    contactTelephone=post_data['contact_telephone'],
    contactCourriel=post_data['contact_email'],
    contactAdresse=post_data['contact_adresse'], demandeObjet=post_data['demande_objet'],
    demandeLieu=post_data['demande_lieu'],
    demandeDescription=post_data['demande_description'],
    remoteAddress=post_data['remote_adresse'], codeEquipement=post_data['code_equipement'],
    codeServiceDemandeur=post_data['code_service_demandeur'],
    dateSouhaitee=post_data['date_souhaite'], typeDemande=post_data['type_demande']
```

je trouve plus court, plus implicite et plus local un :

```
self.build_soap_call(
   wsdl='DemandeService',
   method='insertDemandType').with_post_data(
   request.post_data).map(
   contactNom='contact_nom',
   contactTelephone='contact_telephone').call()
```

en utilisant un pattern builder (\o/ Gang-of-four) ou avec moins d'indentation,

```
soap_call = self.build_soap_call(wsdl='DemandeService', method='insertDemandType')
soap_call.from_post_data(request.post_data)
soap_call.map(contactNom='contact_nom')
result = soap_call.call()
```

#20 - 05 mars 2024 18:00 - Emmanuel Cazenave

- Statut changé de Nouveau à Rejeté

Oublions.

Fichiers

0001-misc-add-parameters-translation-capabilities-34178.patch	4,72 ko	19 juin 2019	Emmanuel Cazenave
0001-misc-add-parameters-rename-capabilities-34178.patch	4,58 ko	20 juin 2019	Emmanuel Cazenave
0001-misc-add-parameters-rename-capabilities-34178.patch	4,61 ko	20 juin 2019	Emmanuel Cazenave
0001-misc-add-parameters-transformation-capabilities-3417.patch	6,17 ko	24 juin 2019	Emmanuel Cazenave