Passerelle - Development #11497

base adresse, liste des communes via API Géo

22 juin 2016 20:13 - Thomas Noël

Statut:	Fermé	**Début:**	22 juin 2016
Priorité:	Normal	**Echéance:**	
Assigné à:	Valentin Deniaud	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Faire un connecteur vers https://api.gouv.fr/api/geoapi.html pour :

- lister des communes et/ou codes postaux dans wcs, depuis un nom et/ou un code postal
- et trouver d'autres idées d'utilisation

Demandes liées:

Lié à Passerelle - Development #26332: base adresse : inclure un endpoint ave...	**Rejeté**	**12 septembre 2018**
Lié à Passerelle - Development #28018: Faire un connecteur API Géo	**Rejeté**	**15 novembre 2018**

Révisions associées

Révision 5c283e02 - 06 décembre 2019 17:07 - Valentin Deniaud

base_adresse: prefer urljoin to os.path.join (#11497)

Révision bccef547 - 06 décembre 2019 17:07 - Valentin Deniaud

base_adresse: add missing asserts in streets tests (#11497)

Révision 27f3bafe - 06 décembre 2019 17:07 - Valentin Deniaud

base_adresse: move endpoint documentation to decorator (#11497)

Révision 1a4c7c64 - 06 décembre 2019 17:07 - Valentin Deniaud

base_adresse: add API Géo endpoints (#11497)

Namely /cities/, /departments/ and /regions/.

Historique

#1 - 05 juin 2018 15:37 - Thomas Noël

- Tracker changé de Bug à Development

- Sujet changé de connecteur GéoAPI à connecteur API Géo

Trouverait sa place dans base_adresse pour fournir /regions, /departments, /cities à côté de streets

- Sur région: filtre par texte (?q=)
- Sur département: filtre par région (?region_id=) et/ou par texte (?q=)
- Sur villes: filtre par région (?region_id=) ou département (?department_id=) ; et/ou par texte (?q=)

Par défaut renvoyer toutes les infos d'API Géo, sauf les contours.

#2 - 12 septembre 2018 09:45 - Frédéric Péters

- Sujet changé de connecteur API Géo à base adresse, liste des communes via API Géo

#3 - 12 septembre 2018 09:46 - Frédéric Péters

- Lié à Development #26332: base adresse : inclure un endpoint avec la liste des communes ajouté

#4 - 15 novembre 2018 09:06 - Frédéric Péters

- Lié à Development #28018: Faire un connecteur API Géo ajouté

#5 - 26 février 2019 10:22 - Brice Mallet

Pour information, articles en rapport :

- https://www.banquedesterritoires.fr/les-acteurs-de-lopen-data-sinterrogent-sur-les-limites-et-les-risques-des-api
- https://medium.com/@cq94/les-api-cest-bien-en-abuser-ca-craint-b5d1c92b32f2 :
  "Dans certains cas, l'API n'apporte pas de réelle valeur ajoutée [...] c'est typiquement le cas pour des données qui ne changent que très rarement et qui ne sont pas volumineuses. Appeler par exemple une API de géocodage pour obtenir le nom d'une commune (parmi désormais moins de 35000) à partir d'un code postal n'est pas forcément pertinent et cela pose un vrai problème de dépendance forte pour un service à valeur ajoutée au final très faible."

#6 - 26 février 2019 10:24 - Frédéric Péters

(...) cela pour dire que sur ce type de données on a plutôt à faire comme les rues, une conservation locale des données.

#7 - 26 février 2019 14:08 - Thomas Noël

De https://github.com/etalab/api-geo/blob/master/download-sources.sh on tire l'URL des données sources http://etalab-datasets.geo.data.gouv.fr/contours-administratifs/latest/geojson/ — les fichiers "100m" contiennent me semble-t-il les contours des objets "à 100 mètres près"

#8 - 26 février 2019 14:10 - Thomas Noël

Thomas Noël a écrit :

> De https://github.com/etalab/api-geo/blob/master/download-sources.sh on tire l'URL des données sources http://etalab-datasets.geo.data.gouv.fr/contours-administratifs/latest/geojson/ — les fichiers "100m" contiennent me semble-t-il les contours des objets "à 100 mètres près"

Mais y'a pas "la" donnée la plus souvent demandée, les codes postaux. A trouver ailleurs... (soupir)...

#10 - 20 novembre 2019 16:34 - Valentin Deniaud

- Assigné à mis à Valentin Deniaud

#11 - 27 novembre 2019 12:14 - Valentin Deniaud

- Fichier 0001-wip-api-geo.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

Je pose un patch wip où il manque surtout les tests et la doc, au cas où quelqu'un ait le temps jeter un œil. N'est présent également que l'endpoint /cities/. Quelques réflexions en passant :

- Dans #28018, Benj demande du JSONP. Il me semble que c'est déprécié, donc je ne m'en préoccupe pas.
- Le connecteur a principalement vocation à être utilisé comme source de donnée JSON dans wcs, pour autocompléter une liste. Partant de là, est-ce qu'au lieu de « renvoyer toutes les infos d'API Géo, sauf les contours », sachant que chaque info génère un form_var_, on ne devrait pas en enlever un maximum ? Les défauts de l'API sont assez sains je trouve[1].
- Le cas d'usage décrit dans #37728 me pose question. 1) l'usager rempli un CP 2) un champs affiche la liste des communes. Outre que je ne sais pas comment construire un tel champs (appel WS du champ 2 qui dépend des données du champ 1 et doit être actualisé), je pense plutôt, et c'est ce que le connecteur permet, qu'on aura simplement un champ CP et que taper le CP donne la ville par autocomplétion (il sera affiché Ville (CP)). De même, un champ ville donne par autocomplétion du nom une liste avec un affichage Ville (CP). Bref, la ville et le code postal dans un seul champ, et demander d'entrer le CP ou la ville est un choix d'UX, mais permettra d'obtenir les deux de toute façon.
- Le problème des codes postaux multiples est résolu en utilisant code_insee.code_postal comme id, et en ajoutant un champ codePostal pour avoir une donnée de formulaire sympa.
- Il n'y a pas de ?region_id ou ?departement_id, car il me semble que c'est quelque chose que l'on spécifiera à la création du connecteur, jamais dans les appels.

[1] Champs inclus par défaut : nom, code insee, codes postaux, code département, code région, population. Exclus : contours, centre, surface, département, région. Limite je serais pour exclure code département et code région, c'est le genre de chose qu'on aura spécifié dans la requête de toute façon, ainsi que population, si le boost fonctionne sans.

#12 - 27 novembre 2019 13:04 - Benjamin Dauvergne

Valentin Deniaud a écrit :

> Je pose un patch wip où il manque surtout les tests et la doc, au cas où quelqu'un ait le temps jeter un œil. N'est présent également que l'endpoint /cities/. Quelques réflexions en passant :
>
> - Dans #28018, Benj demande du JSONP. Il me semble que c'est déprécié, donc je ne m'en préoccupe pas.

JSONP est nécessaire pour l'autocomplétion mais il est vrai qu'on sait convertir du JSON en JSONP directement dans w.c.s. désormais ; juste se

dire que navigo->w.c.s.->passerelle ça chargera plus que navigo->passerelle, n'ayant pas mesuré ce n'est pas un point bloquant.

- Le connecteur a principalement vocation à être utilisé comme source de donnée JSON dans wcs, pour autocompléter une liste. Partant de là, est-ce qu'au lieu de « renvoyer toutes les infos d'API Géo, sauf les contours », sachant que chaque info génère un form_var_, on ne devrait pas en enlever un maximum ? Les défauts de l'API sont assez sains je trouve[1].

Qui peut le plus peut le moins, pourquoi ne pas tout renvoyer tant qu'on en sait pas plus ?

- Le cas d'usage décrit dans #37728 me pose question. 1) l'usager rempli un CP 2) un champs affiche la liste des communes. Outre que je ne sais pas comment construire un tel champs (appel WS du champ 2 qui dépend des données du champ 1 et doit être actualisé), je pense plutôt, et c'est ce que le connecteur permet, qu'on aura simplement un champ CP et que taper le CP donne la ville par autocomplétion (il sera affiché Ville (CP)). De même, un champ ville donne par autocomplétion du nom une liste avec un affichage Ville (CP). Bref, la ville et le code postal dans un seul champ, et demander d'entrer le CP ou la ville est un choix d'UX, mais permettra d'obtenir les deux de toute façon.

Si ce n'est pas plus cher de savoir tout faire je dirai de ne pas se préoccuper des UX pour la construction du connecteur.

- Le problème des codes postaux multiples est résolu en utilisant code_insee.code_postal comme id, et en ajoutant un champ codePostal pour avoir une donnée de formulaire sympa.

Est-ce que c'est vraiment impossible d'avoir un même code_insee.code_postal pour un nom de commune différent ? J'ai l'impression qu'à la campagne ça arrive avec des hameaux qui sont des lieux-dits, mais il faudrait vérifier dans une liste de tous les codes postaux/code insee.

Et donc j'ai vérifié sur https://www.data.gouv.fr/fr/datasets/base-officielle-des-codes-postaux/ et non ça n'existe pas, il y a juste beaucoup de doublons (insee,postal,commune) avec les variations sur les champs Libelle_acheminement et Ligne_5 (je ne sais pas à quoi ça sert, je crois que justement ça concerne les lieux-dits).

- Il n'y a pas de ?region_id ou ?departement_id, car il me semble que c'est quelque chose que l'on spécifiera à la création du connecteur, jamais dans les appels.

Ce serait plus sympa d'avoir tout dans un seul connecteur, les connecteurs où on indique le département c'est parce qu'on met en cache les fichiers complets, qu'il sont découpés par département et qu'on a pas envie de charger tout le pays (pour les rues) sur chaque instance; mais ce n'est pas forcément très intelligent, on ferait mieux de mutualiser un connecteur pour tous les clients.

> [1] Champs inclus par défaut : nom, code insee, codes postaux, code département, code région, population. Exclus : contours, centre, surface, département, région. Limite je serais pour exclure code département et code région, c'est le genre de chose qu'on aura spécifié dans la requête de toute façon, ainsi que population, si le boost fonctionne sans.

#13 - 27 novembre 2019 13:58 - Valentin Deniaud

- Statut changé de Solution proposée à En cours

Ça roule merci, en route vers un vrai patch.

#14 - 27 novembre 2019 21:32 - Benjamin Dauvergne

Je ne trouve pas où dans la doc tu as trouvé l'usage du paramètre boost.

PS: là https://github.com/etalab/api-geo/blob/92f7a8bf9e966f0e03904ccd225028201329620a/lib/communes.js ?

#15 - 28 novembre 2019 09:51 - Valentin Deniaud

Non je n'ai pas trouvé la doc, Thomas m'avait soufflé à l'oreille que ça marchait comme ça, j'ai supposé que c'était classique et confirmé à la main (recherche de « Paris », Paris arrive bien en tête de liste et pas Parisot). Visiblement tu as déniché le bon bout de code !

#16 - 28 novembre 2019 10:21 - Thomas Noël

As usual j'ai rien inventé (snif), c'est au fin fond de la doc, mais le secret c'est qu'il y a plusieurs docs... Donc dans https://geo.api.gouv.fr/decoupage-administratif/communes « L'option boost=population vous permet de prioriser les résultats avec une plus grande population. »

#17 - 28 novembre 2019 12:57 - Benjamin Dauvergne

Thomas Noël a écrit :

> As usual j'ai rien inventé (snif), c'est au fin fond de la doc, mais le secret c'est qu'il y a plusieurs docs... Donc dans https://geo.api.gouv.fr/decoupage-administratif/communes « L'option boost=population vous permet de prioriser les résultats avec une plus grande population. »

C'est vraiment formidable ces documentations distribués.

#18 - 28 novembre 2019 14:33 - Valentin Deniaud

- Fichier 0002-base_adresse-add-API-G-o-endpoints-11497.patch ajouté

- Fichier 0001-base_adresse-move-endpoint-documentation-to-decorato.patch ajouté

- Statut changé de En cours à Solution proposée

Voili voilou, le seul point à réfléchir était le coup des codes postaux multiples, sinon c'est globalement un proxy vers l'API Géo.

Deux remarques :

- Le premier patch enlève la documentation des endpoints hardcodée dans le template pour la laisser s'autogénérer proprement, mais du coup ça casse la valeur d'exemple de /streets/ qui faisait appel à self. Worth it imo, mais si il y a un moyen de s'en sortir mieux je prends.
- Il est permis de faire un appel à /cities/ sans paramètre, du coup c'est très long et sans usage pertinent, est-ce qu'il vaut mieux l'empêcher en retournant une erreur ?

#19 - 28 novembre 2019 15:55 - Thomas Noël

À mon tour...

- sur la recherche q= il faut être plus intelligent. On veut permettre à l'usager de taper un code postal : il faut donc, si la personne a tapé "14000", remonter les villes qui ont ce code postal.
- je sais pas si c'est possible, mais si on pouvait remonter toutes les villes dont le code postal "commence par" (genre 14 ou 140) ça serait super chouette. Comme apigéo ne semble pas savoir le faire, on peut laisser cette feature pour plus tard, mais ça serait bien à l'avenir. Idée : quand on détecte au moins 2 chiffres de suite (et moins de 5), chercher dans le département XX concerné toutes les villes, et ne renvoyer que celles sont un code postal commence effectivement par les chiffres concernés.
- J'aurai utilise "department" au lieu de "county", parce que county est vraiment pas un mot habituel je trouve (enfin pour moi).
- Juste pour le jour où quelqu'un le lire vraiment, dans api_description :

  "endpoints source data from API Geo." -> "endpoints source data from French API Geo."

- Ce bout de code va raiser si id ne contient pas de point :

  ```
      @code, zipcode = id.split('.')@
  ```

  Il faudrait plutôt alors renvoyer un 400 (raise APIError...)
- Ajoute aussi de quoi surcharger boost et fields
- ne laisse pas de "**kwargs" trainer dans les endpoiints si tu ne t'en sert pas. Ca permettra de gratuitement sortir une 400 qui quelqu'un essaye d'envoyer un truc pas prévu.
- '.'.join((city['code'], zipcode)) ouaip hé, quand même, hein, bon : moi je trouve plus clair un bête et méchant '%s.%s' % (city['code'], zipcode)
- new_city['text'] = '%s (%s)' % (city['nom'], zipcode) : en France on préfèrera la notation « "%s %s" % (zipcode, city['nom']) »
- attention tu utilise le même nom "zipcode" dans ta boucle et comme argument de fonction, ça rend pas le code super facile à lire (même si ça marche, c'est pas très joli et pas recommandé)
- dans le endpoint counties : pourquoi garder un argument "code" alors qu'on a "id" qui fait tout aussi bien ?
- même remarque sur le format de sortie, plutôt "61 Orne"

J'arrête ici :)

#20 - 28 novembre 2019 16:02 - Thomas Noël

Oublié de répondre :

> - Le premier patch enlève la documentation des endpoints hardcodée dans le template pour la laisser s'autogénérer proprement, mais du coup ça casse la valeur d'exemple de /streets/ qui faisait appel à self. Worth it imo, mais si il y a un moyen de s'en sortir mieux je prends.

Pas grave, mais met plutôt 75000 en exemple, et zou.

> - Il est permis de faire un appel à /cities/ sans paramètre, du coup c'est très long et sans usage pertinent, est-ce qu'il vaut mieux l'empêcher en retournant une erreur ?

Nope, laisse filer, 1 seconde c'est pas si long. Quand à l'usage : on ne sait jamais, avec les fous qui nous entourent.

#21 - 28 novembre 2019 16:18 - Valentin Deniaud

- Statut changé de Solution proposée à En cours

Thomas Noël a écrit :

- J'aurai utilise "department" au lieu de "county", parce que county est vraiment pas un mot habituel je trouve (enfin pour moi).

Je suis d'accord que c'est un mot chelou, mais il n'y a rien à faire, department ne veut pas du tout dire département (et county est déjà présent dans le code).

- dans le endpoint counties : pourquoi garder un argument "code" alors qu'on a "id" qui fait tout aussi bien ?

J'avais dans l'idée qu'id était un peu un paramètre à usage interne, qui la plupart du temps ne représente rien, donc par soucis de consistance qu'il valait mieux éviter de le faire représenter quelque chose quand ça nous arrange. Et puis à propos de consistance, on a toujours un paramètre "code" qui veut dire code INSEE pour les trois endpoints, ça rend assez clair je trouve.

> Pas grave, mais met plutôt 75000 en exemple, et zou.

Le raisonnement derrière le 00000 : les seules rues chargées sont celles du code postal préalablement défini, donc mettre un code invalide qui crie « changez moi ! » c'est possiblement mieux que de mettre un code valide dont on sait qu'il ne marchera pas, « tiens il y a rien qui s'affiche ça doit être cassé ».

> Nope, laisse filer, 1 seconde c'est pas si long.

1 seconde le retour de l'api, xx secondes et un lag de toute la page si ça se retrouve dans le html...

> J'arrête ici :)

Merci, je m'y remets !

#22 - 28 novembre 2019 16:21 - Benjamin Dauvergne

Thomas Noël a écrit :

> À mon tour...
>
> - sur la recherche q= il faut être plus intelligent. On veut permettre à l'usager de taper un code postal : il faut donc, si la personne a tapé "14000", remonter les villes qui ont ce code postal.

On se ferait pas moins chier à oublier complètement API Geo et à s'en servir juste comme source de donnée qu'on met en cache puis qu'on gère filtre/indexe comme on a envie ? En plus ça ajoute un point unique de défaillance dont on se passerait bien (pour rebondir sur le commentaire de Brice plus haut).

- J'aurai utilise "department" au lieu de "county", parce que county est vraiment pas un mot habituel je trouve (enfin pour moi).

Comment ça ? et le comté de Hazzard ?

- Ce bout de code va raiser si id ne contient pas de point :
  [...]
  Il faudrait plutôt alors renvoyer un 400 (raise APIError...)

Oui et .split('.', 1) au cas où il y a plusieurs points.

- '.'.join((city['code'], zipcode)) ouaip hé, quand même, hein, bon : moi je trouve plus clair un bête et méchant '%s.%s' % (city['code'], zipcode)

idem.

- new_city['text'] = '%s (%s)' % (city['nom'], zipcode) : en France on préfèrera la notation « "%s %s" % (zipcode, city['nom']) »

idem.

- dans le endpoint counties : pourquoi garder un argument "code" alors qu'on a "id" qui fait tout aussi bien ?

On peut se permettre un endpoint departements, faut pas avoir peur du français dans ces cas là, on est quand même en pleine franchouillardise là.

#23 - 28 novembre 2019 16:53 - Thomas Noël

Valentin Deniaud a écrit :

> Thomas Noël a écrit :
>
> - J'aurai utilise "department" au lieu de "county", parce que county est vraiment pas un mot habituel je trouve (enfin pour moi).
>
> Je suis d'accord que c'est un mot chelou, mais il n'y a rien à faire, department ne veut pas du tout dire département (et county est déjà présent dans le code).

Je me basais sur https://en.wikipedia.org/wiki/Department_(country_subdivision) alors arrêter de dire que je dis des conneries non mais ho.

> - dans le endpoint counties : pourquoi garder un argument "code" alors qu'on a "id" qui fait tout aussi bien ?
>
> J'avais dans l'idée qu'id était un peu un paramètre à usage interne, qui la plupart du temps ne représente rien, donc par soucis de consistance qu'il valait mieux éviter de le faire représenter quelque chose quand ça nous arrange. Et puis à propos de consistance, on a toujours un paramètre "code" qui veut dire code INSEE pour les trois endpoints, ça rend assez clair je trouve.

Ca roule.

> Pas grave, mais met plutôt 75000 en exemple, et zou.
>
> Le raisonnement derrière le 00000 : les seules rues chargées sont celles du code postal préalablement défini, donc mettre un code invalide qui crie « changez moi ! » c'est possiblement mieux que de mettre un code valide dont on sait qu'il ne marchera pas, « tiens il y a rien qui s'affiche ça doit être cassé ».

En fait tu peux ne pas mettre d'exemple, ça sera encore mieux (ça permettra de voir toutes les rues en cache), mais en revanche ajouter page_limit=30 sur l'exemple (et les autres paramètres, pour qu'ils soient un peu documentés et pas cachés comme actuellement)

> Nope, laisse filer, 1 seconde c'est pas si long.
>
> 1 seconde le retour de l'api, xx secondes et un lag de toute la page si ça se retrouve dans le html...

True. Mais bon, entre ça et planter, je préfère le lag.

#24 - 28 novembre 2019 16:57 - Thomas Noël

Benjamin Dauvergne a écrit :

> - sur la recherche q= il faut être plus intelligent. On veut permettre à l'usager de taper un code postal : il faut donc, si la personne a tapé "14000", remonter les villes qui ont ce code postal.
>
> On se ferait pas moins chier à oublier complètement API Geo et à s'en servir juste comme source de donnée qu'on met en cache puis qu'on gère filtre/indexe comme on a envie ? En plus ça ajoute un point unique de défaillance dont on se passerait bien (pour rebondir sur le commentaire de Brice plus haut).

C'est vrai que ça donnerait peut-être un code plus joli à lire, en plus. Avoir trois tables ville/departement/region, avec une synchro à la streets, et zou. Il faudrait éventuellement gérer le q= avec un truc genre trigramme, pour être solide ; même si dans un premier temps on peut faire de la recherche LIKE/unaccent toute bête.

A voir si tu as envie d'aller dans ce sens, Valentin.

#26 - 03 décembre 2019 11:11 - Valentin Deniaud

Une question me vient : est-ce que c'est envisageable de ne renvoyer que id et text lors d'un appel avec ?q=, et toutes les données seulement avec un appel avec ?id= ? Il y a pas photo dans la rapidité de l'autocomplétion, et vraisemblablement w.c.s. n'utilise que id et text, donc pas de différence pour un gain de confort usager réel.
L'inconvénient c'est que si on imagine que l'API du connecteur a vocation à être utilisée hors autocomplétion w.c.s., on pourrait aimer avoir tout tout de suite en recherchant sur ?q=. Mais à ce moment là, un paramètre ?full=true à spécifier en plus, et on s'en sort.

#27 - 03 décembre 2019 11:35 - Frédéric Péters

> ... est-ce que c'est envisageable ...

On pourra vouloir, pour une recherche ?q=..., dans w.c.s. lors de l'autocomplétion avoir également accès aux différents morceaux, donc je dirais de tout le temps tout renvoyer. (et j'écris ça sans savoir ce que "tout" représente comme données, mais j'imagine qu'on n'a pas de photos des façades, et que du coup ça reste quand même léger).

#28 - 03 décembre 2019 11:56 - Valentin Deniaud

> ça reste quand même léger

Oui, la différence entre les deux c'est « rapide » ou « instantané » (quoique quand on entre juste un chiffre, les (1/10 * 30 000) données des communes en moyenne qui arrivent font passer de rapide à lent, ou de instantané à rapide).

Mais est-ce que la bonne manière de faire ça ne serait pas plutôt de patcher w.c.s. pour qu'il appelle ?id= à la sélection du choix ? Ça permettrait de faire marcher les champs conditionnels, en plus. Ou alors tu imagines un cas où on a besoin des données de tous les choix dans la liste qui s'affiche, rien à faire à ce moment là (sauf à gérer un nouveau paramètre ?full comme je l'écrivais).

#29 - 03 décembre 2019 12:02 - Frédéric Péters

> pour qu'il appelle ?id= à la sélection du choix

Je pense qu'il fait déjà ça, et on fait des champs conditionnels basés sur les attributs supplémentaires reçus, mais dans la perspective des évolutions "adresse", il pourrait justement y avoir volonté de pouvoir traiter le tout directement à la sélection javascript, sans refaire un roundtrip http vers w.c.s. (puis lui vers passerelle, etc.). (une autre idée évoquée à un moment était que w.c.s. puisse se passer d'appel à l'endpoint ?id=, en capturant à la volée les détails fournis dans une réponse à ?q=). Voilà, pour ces raisons, je serais pour un retour intégral dans ?q=.

#30 - 03 décembre 2019 14:58 - Valentin Deniaud

Valentin Deniaud a écrit :

> Il y a pas photo dans la rapidité de l'autocomplétion

(tout était de ma faute, un select_related bien placé résout mon problème de lenteur, qui n'avait rien à voir avec la quantité de données envoyées, désolé pour le bruit)

#31 - 03 décembre 2019 16:37 - Valentin Deniaud

- Fichier 0002-base_adresse-add-API-G-o-endpoints-11497.patch ajouté

- Statut changé de En cours à Solution proposée

Voilà la version avec mise en cache, c'est plutôt très différent de l'autre. J'ai pris en compte vos remarques là où elles avaient encore du sens.

Pas encore de trigramme et il manque le test de la méthode qui met à jour les données.

#32 - 03 décembre 2019 17:24 - Benjamin Dauvergne

Valentin Deniaud a écrit :

> Pas encore de trigramme et il manque le test de la méthode qui met à jour les données.

À quoi sert le filtrage des régions, départements, code postaux sur la mise à jour ? On veut tout non ? (je répète on est sur 36000 lignes, y a moins d'3Mo de données là, bien sûr ce n'est pas cool de recopier ça pour tous nos clients et ce serait mieux de l'avoir qu'une seule fois mais au pire pour 200 clients on parle de 600 Mo sur disque).

#33 - 04 décembre 2019 00:00 - Thomas Noël

Benjamin Dauvergne a écrit :

> Valentin Deniaud a écrit :
>
> > Pas encore de trigramme et il manque le test de la méthode qui met à jour les données.

> À quoi sert le filtrage des régions, départements, code postaux sur la mise à jour ? On veut tout non ?

Oui oui Valentin, on prend tout. Un filtrage sera éventuellement fait lors des requêtes, par exemple pour limiter la recherche à un département (c'est le seul cas que je vois à cet instant, pour les métropoles on verra plus tard si on trouve une idée pertinente).

#34 - 04 décembre 2019 10:51 - Valentin Deniaud

Benjamin Dauvergne a écrit :

> À quoi sert le filtrage des régions, départements, code postaux sur la mise à jour ?

Mmmh, je ne vois pas à quoi tu fais référence.
En revanche je ne récupère pas 'surface' ni 'centre', un peu par flemme et parce que c'est inutile par rapport aux cas d'usages dont j'ai eu vent, que ça se rajoute facilement dans le cas improbable où le besoin surgit, mais ok je peux faire le taf maintenant (avec 'centre' dans un JSONField). Et si on veut aussi stocker les contours, c'est déjà plus chiant, l'API renvoie des BadRequest si on demande tout d'un coup (je suppose que la réponse dépasse une taille limite).

#35 - 04 décembre 2019 11:13 - Benjamin Dauvergne

Valentin Deniaud a écrit :

> Benjamin Dauvergne a écrit :
>
> > À quoi sert le filtrage des régions, départements, code postaux sur la mise à jour ?
>
> Mmmh, je ne vois pas à quoi tu fais référence.

Ok j'ai cru que self.department limitait aussi l'import, je ne suis pas certain qu'on ait besoin d'un filtrage de base même sur le endpoint mais d'autres dirong.

> En revanche je ne récupère pas 'surface' ni 'centre', un peu par flemme et parce que c'est inutile par rapport aux cas d'usages dont j'ai eu vent, que ça se rajoute facilement dans le cas improbable où le besoin surgit, mais ok je peux faire le taf maintenant (avec 'centre' dans un JSONField). Et si on veut aussi stocker les contours, c'est déjà plus chiant, l'API renvoie des BadRequest si on demande tout d'un coup (je suppose que la réponse dépasse une taille limite).

Oui ça ok, on pourra voir une autre fois si on en a besoin (enfin centre peut-être que ça pourrait servir rapidement pour faire un boost/tri par proximité plutôt que par population mais on verra à ce moment là).

Pour moi c'est bon alors, je laisse Thomas relire une dernière fois.

#36 - 04 décembre 2019 11:27 - Thomas Noël

Faut juste virer self.region et self.departement, ça n'est pas utile.

En revanche, documente le paramètre "id". Pour la doc de "q", disons plutôt "Search text in name or postal code or ..."

Ensuite lors de la synchro des données, prévoir quand même de faire du delete, car le jour où une ville sera supprimée, il faudra qu'on le sache.

Pour les id, je suis pas partisan d'utiliser ceux de la table, qui ne seront pas fixes d'une instance à l'autre. Plutot partir sur les code fournis par l'api géo, sachant que pour les villes ça serait insee+code_postal.

Prévoit aussi que l'API ne soit dispo 24*7 et gère les possibilités de soucis lors des requests.get, y compris un JSON valable mais vide... (on a déjà vu la BANO remonter des listes vides pendant 24h, ça serait pas cool que ça arrive aussi sur l'api géo, mais j'ai la confiance à zéro sur les api de l'état).

Et aussi, je pense pas qu'il soit bien d'utiliser os.path.join pour créer des URLs, même si ça a été fait comme ça ailleurs, utiliser plutôt urlparse.urljoin

#37 - 04 décembre 2019 16:28 - Valentin Deniaud

- Fichier 0004-base_adresse-add-API-G-o-endpoints-11497.patch ajouté

- Fichier 0003-base_adresse-move-endpoint-documentation-to-decorato.patch ajouté

- Fichier 0002-base_adresse-add-missing-asserts-in-streets-tests-11.patch ajouté

- Fichier 0001-base_adresse-prefer-urljoin-to-os.path.join-11497.patch ajouté

Remarques appliquées. Ne manque plus que l'utilisation des trigrammes.

#38 - 05 décembre 2019 11:01 - Valentin Deniaud

Pour les trigrammes, j'ai beau avoir ajouté postgres dans les installed_apps et activé l'extension dans un shell psql comme dit ici https://docs.djangoproject.com/en/1.11/ref/contrib/postgres/lookups/#trigram-similarity, je n'arrive pas à avoir une requête qui fonctionne :

```
>>> StreetModel.objects.filter(name__trigram_similar='hop')
ProgrammingError: operator does not exist: character varying % unknown
LINE 1: ...eetmodel" WHERE "base_adresse_streetmodel"."name" % 'hop' OR...
                                                             ^
HINT:  No operator matches the given name and argument types. You might need to add explicit type casts.
```

Pourtant c'est exactement l'exemple de la doc, et mes recherches sur cette erreur n'ont rien renvoyé de pertinent. Ça sera pour une prochaine fois, donc.

#39 - 06 décembre 2019 01:32 - Thomas Noël

- 0001 ne passera pas seul, il manque l'import de urljoin.
- 0002 hé hé hé, of course
- 0003 il manque la description de "id" pour streets.

0004 donc.

Utiliser le six de Django.

Pour clarifier, au lieu de "City code" tu peux carrément dire "INSEE code"

Dans cities, tu as mis les "if code/ if region_code/ if departement_code" en dehors du "if id". Soit tu les mets dans le else, soit tu ne mets pas le "if q" dans le else... (la dernière solution a ma préférence, ie tous les paramètres sont des filtres et voili voilà). Même chose pour les autres endpoints regions et departements.

Dans départements, le "if not 2 <= len(id) <=3" n'est pas utile, filter va faire son travail de toute façon. Même chose pour le "id = int(id)" dans regions.

unicodedata.normalize('NFKD', q).encode('ascii', 'ignore').lower() : tu peux utiliser le nouveau passerelle.utils.conversion::simplify à la place (arrivé hier dans master, bisous Nicolas)

Dans get_api_geo_endpoint tu regardes le json même si le code de retour n'est pas 200, il faut pas. Aussi, en cas de result json correct mais vide (if not result), il faut méchamment cracher pour que ça alerte la terre entière (comme pour les rues, un sale « raise Exception('api geo returns empty json') », ça ne doit vraiment jamais arriver, il faut que l'exception remonte au plus haut niveau des autorités de ce pays)

Dans update_api_geo_data, au cas où les download prennent du temps ou crashent, les faire tous les trois d'abord, puis seulement commencer à compter (start_update = timezone.now()) et faire les update_or_create/delete.

Laisse le code de RegionModel en simple CharField... on sait jamais, la colonisation peut reprendre.

Aussi, les name à 100 caractères et unaccented_name à 150 : met 150 pour les deux.

Dans le to_json de CityModel, renvoie toujours le même dictionnaire, et ajoute les noms des objets liés, ça gagnera un poil de temps :

```
        data = {
           'text': str(self),
           'id': '%s.%s' % (self.code, self.zipcode),
           'code': self.code,
           'name': self.name,
           'zipcode': self.zipcode,
           'population': self.population,
           'department_code': self.department.code if self.departement else None,
           'department_name': self.department.name if self.departement else None,
           'region_code': self.region.code if self.region else None,
           'region_name': self.region.name if self.region else None,
        }
```

Enfin dans le template, au même niveau que l'infonotice

```
{% trans "Street data is not available yet, it should soon be downloaded." %}
```

ajoute une infonotice identique sur API Géo si l'URL est configurée mais qu'il n'y a pas encore de ville dans la base.

Bref, ça peut paraître long mais ce ne sont que de petits détails. Ça sent bon le ack au prochain coup !

#40 - 06 décembre 2019 12:50 - Valentin Deniaud

- Fichier 0004-base_adresse-add-API-G-o-endpoints-11497.patch ajouté

- Fichier 0003-base_adresse-move-endpoint-documentation-to-decorato.patch ajouté

- Fichier 0002-base_adresse-add-missing-asserts-in-streets-tests-11.patch ajouté

- Fichier 0001-base_adresse-prefer-urljoin-to-os.path.join-11497.patch ajouté

Hop !

> Utiliser le six de Django.

Fait dans 0001.

#41 - 06 décembre 2019 14:58 - Thomas Noël

Un pépin ici :

```
+         try:
+             response = self.requests.get(urljoin(self.api_geo_url, endpoint))
+         except ConnectionError as e:
+             error = e
+         if response.status_code == 200:
```

En cas de ConnectionError il ne faut pas continuer, le if response.status_code va crasher. Tu peux chercher à ajouter un test (avec side_effect=ConnectionError...)

Aussi, il faut pouvoir couper le mécanisme d'update, par exemple sur les instances qui n'ont pas accès à Internet. Pour cela, simplement ajouter un "if not self.api_geo_url: return" dans update_api_geo_data. Il suffira de vider l'URL dans la config du connecteur pour que la mise à jour soit coupée.

Et la condition sur le message dans le template deviendra {% if object.api_geo_url and not object.cities_exist %}...

#42 - 06 décembre 2019 15:37 - Valentin Deniaud

- Fichier 0004-base_adresse-add-API-G-o-endpoints-11497.patch ajouté

J'en ai profité pour changer ConnectionError en RequestError.

#43 - 06 décembre 2019 16:17 - Thomas Noël

Ah, je l'avais écrit plus haut, j'aimerai bien qu'on ai dans le to_json de City :

```
  'department_name': self.department.name if self.departement else None,
  'region_name': self.region.name if self.region else None,
```

Cas d'usage : un simple champ form_var_ville qui permettra de choisir sa ville dans la France entière, avec ça on aura en cadeau form_var_ville_departement_name et _region_name, sans avoir besoin d'aller chercher quel est le nom de departement_code 61.

Promis, après, je acke :)

#44 - 06 décembre 2019 16:24 - Valentin Deniaud

- Fichier 0004-base_adresse-add-API-G-o-endpoints-11497.patch ajouté

En effet j'avais lu trop vite ton diff.

#45 - 06 décembre 2019 17:00 - Thomas Noël

- Statut changé de Solution proposée à Solution validée

Attention, ton "commit -a" a dû ajouter les traductions. Retire-les du commit, et pushe !

#46 - 06 décembre 2019 17:08 - Valentin Deniaud

Ah oui oups, à propos, je les pousse quand ?

#47 - 06 décembre 2019 17:09 - Valentin Deniaud

- Statut changé de Solution validée à Résolu (à déployer)

```
commit 1a4c7c649be00232d471559ebcd420b5b25882e2
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Mon Nov 25 18:00:33 2019 +0100

    base_adresse: add API Géo endpoints (#11497)
```

```
    Namely /cities/, /departments/ and /regions/.

commit 27f3bafed5c434d7a3d9091077cbf4e88c61a32f
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Nov 27 17:01:07 2019 +0100

    base_adresse: move endpoint documentation to decorator (#11497)

commit bccef547537f0a69306a2784c9c93462639a1efd
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Dec 4 15:07:39 2019 +0100

    base_adresse: add missing asserts in streets tests (#11497)

commit 5c283e0277fbf3fad7f40c7f753f12c085a2021c
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Dec 4 14:10:51 2019 +0100

    base_adresse: prefer urljoin to os.path.join (#11497)
```

#48 - 06 décembre 2019 17:17 - Thomas Noël

On écrit et on pousse les traductions au moment de la release

(c'est un fichier commun, on n'y travaille pas chacun dans son coin sinon ça conflicterait de partout)

#49 - 07 décembre 2019 18:15 - Frédéric Péters

- Statut changé de Résolu (à déployer) à Solution déployée

Fichiers

0001-wip-api-geo.patch	3,31 ko	27 novembre 2019	Valentin Deniaud
0001-base_adresse-move-endpoint-documentation-to-decorato.patch	3,94 ko	28 novembre 2019	Valentin Deniaud
0002-base_adresse-add-API-G-o-endpoints-11497.patch	18,1 ko	28 novembre 2019	Valentin Deniaud
0002-base_adresse-add-API-G-o-endpoints-11497.patch	26,8 ko	03 décembre 2019	Valentin Deniaud
0003-base_adresse-move-endpoint-documentation-to-decorato.patch	4,19 ko	04 décembre 2019	Valentin Deniaud
0004-base_adresse-add-API-G-o-endpoints-11497.patch	34,1 ko	04 décembre 2019	Valentin Deniaud
0002-base_adresse-add-missing-asserts-in-streets-tests-11.patch	1,16 ko	04 décembre 2019	Valentin Deniaud
0001-base_adresse-prefer-urljoin-to-os.path.join-11497.patch	1,31 ko	04 décembre 2019	Valentin Deniaud
0003-base_adresse-move-endpoint-documentation-to-decorato.patch	4,26 ko	06 décembre 2019	Valentin Deniaud
0004-base_adresse-add-API-G-o-endpoints-11497.patch	35,6 ko	06 décembre 2019	Valentin Deniaud
0002-base_adresse-add-missing-asserts-in-streets-tests-11.patch	1,16 ko	06 décembre 2019	Valentin Deniaud
0001-base_adresse-prefer-urljoin-to-os.path.join-11497.patch	1,77 ko	06 décembre 2019	Valentin Deniaud
0004-base_adresse-add-API-G-o-endpoints-11497.patch	36 ko	06 décembre 2019	Valentin Deniaud
0004-base_adresse-add-API-G-o-endpoints-11497.patch	57,1 ko	06 décembre 2019	Valentin Deniaud