w.c.s. - Development #75724

Formulaires: JS permettant une validation à la volée des champs

22 mars 2023 17:29 - Thomas Jund (congés, retour le 29/04)

Statut:	Fermé	**Début:**	22 mars 2023
Priorité:	Normal	**Echéance:**	
Assigné à:	Frédéric Péters	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Non	**Planning:**	Non

Description

Note décidé pendant le camp front :

validation à la sortie des champs (blur)
comportement différent à la correction (c'est-à-dire quand l'usager modifie un champ qui était marqué en erreur) : vérification à la saisie (avec petit délai)

Demandes liées:

Lié à w.c.s. - Development #76632: code serveur pour la validation à la volée	**Fermé**	**14 avril 2023**

Révisions associées

Révision 2d784e00 - 18 avril 2023 14:23 - Thomas Jund (congés, retour le 29/04)

js: add live field validation (#75724)

Historique

#1 - 30 mars 2023 14:56 - Thomas Jund (congés, retour le 29/04)

- Statut changé de Nouveau à Information nécessaire

- Assigné à changé de Thomas Jund (congés, retour le 29/04) à Frédéric Péters

Premiers tests.
Proto qui gère blur uniquement pour le moment, mais permet une première base de discussion sur les choix.
https://codepen.io/Sacripant/pen/xxaeLoa

Je suis parti sur l'utilisation de l'API native des browsers :
https://developer.mozilla.org/en-US/docs/Learn/Forms/Form_validation#validating_forms_using_javascript
Cela permet d'être plus efficace, plus concis en JS, de multiplier les messages d'aides à l'usager (voir champ code postal qui propose 4 messages d'erreurs différents en fonction des erreurs à corriger), fonctionner avec les CSS `:valid`, `:invalid`, etc.

Ce choix nous incite à utiliser les attributs HTML5 de validation, qui ne sont pas utilisés actuellement :

- Attribut `required` pour les champs requis
- Attributs `pattern` pour les validations du format du contenu
- Éventuellement ajouter des attributs minlength maxlength quand c'est justifié (par ex champ code postal)

Liste des attributs et types d'erreurs retournées : https://developer.mozilla.org/en-US/docs/Web/HTML/Constraint_validation

J'ai fait le choix aussi de poser les différents messages d'erreurs dans des balises `template` en sein de chaque div.widget (peut-être un peu redondant, mais safe), et le template du container des erreurs au sein d'un `<template> à début du form` pour permettre l'utilisation d'un gabarit pour les messages.

Restera encore à définir ce que l'on fait pour les validations qui nécessitent une requête vers le serveur (une liste m'aiderait bien).

#2 - 30 mars 2023 15:24 - Frédéric Péters

- Statut changé de Information nécessaire à En cours

- Assigné à changé de Frédéric Péters à Thomas Jund (congés, retour le 29/04)

> Ce choix nous incite à utiliser les attributs HTML5 de validation, qui ne sont pas utilisés actuellement :

On est déjà passé par là et on a du faire marche arrière. #30419. C'était principalement des questions d'affichage (gérées par ton approche, il me semble) mais il y avait aussi des regex incompatibles entre python et javascript (malheureusement à l'époque je n'avais pas pu remettre la main sur celle rencontrée).

> Restera encore à définir ce que l'on fait pour les validations qui nécessitent une requête vers le serveur (une liste m'aiderait bien).

Je pense qu'il faut partir dans l'autre sens : il faut d'abord considérer la validation par requête vers le serveur, qui peut couvrir tous les cas, puis petit à petit, pour ce qui pourrait être fait en local, le faire. Bref pas de liste de ce qui doit être fait sur le serveur, la liste c'est "tout sauf ce qui peut être fait en local".

(inutile de m'attribuer le ticket, je reçois de toute façon une notification).

#3 - 30 mars 2023 15:35 - Frédéric Péters

> (une liste m'aiderait bien).

mais peut-être qu'il serait juste utile de comprendre pourquoi une telle liste aiderait.

#4 - 30 mars 2023 15:38 - Frédéric Péters

Malgré tout, quand même, d'un œil ce qui aujourd'hui a une validation serveur qui ne peut pas du tout être réimplémentée en js :

- numéro de téléphone français
- regex arbitraire (parce que différences entre regex python et regex javascript)
- condition django

(mais je répète que clairement je ne bloquerais pas sur, par exemple, l'implémentation d'une validation javascript des numéros d'IBAN, que pour toutes les validations ça peut être fait côté serveur).

#5 - 30 mars 2023 16:19 - Thomas Jund (congés, retour le 29/04)

> On est déjà passé par là et on a du faire marche arrière. #30419. C'était principalement des questions d'affichage (gérées par ton approche, il me semble) mais il y avait aussi des regex incompatibles entre python et javascript (malheureusement à l'époque je n'avais pas pu remettre la main sur celle rencontrée).

Oui en effet, dans ma proposition, il n'y a aucun problème d'affichage. On gère l'affichage comme bon nous semble (cf proto).
Pour les regex,

> Je pense qu'il faut partir dans l'autre sens : il faut d'abord considérer la validation par requête vers le serveur, qui peut couvrir tous les cas, puis petit à petit, pour ce qui pourrait être fait en local, le faire. Bref pas de liste de ce qui doit être fait sur le serveur, la liste c'est "tout sauf ce qui peut être fait en local".

Dommage de charger le serveur par plein de micro-requêtes qui pourraient majoritairement être évitées. Mais OK.

Proposition :

Passer une url de validation via un attribut data : `data-validation-url="https://form-url/validation/field_name"`
L'idéal serait que le serveur retourne un objet compatible / équivalement à celui de l'API JS (en gardant uniquement le pertinant) :
Sachant que `valueMissing` peut être géré côté browser, qu'on n'utilise pas à ma connaissance ni les ranges, ni les steps. on se retrouve avec un json réduit
On conserve l'idée des messages d'erreurs dans l'html et le serveur retourne simplement

```
{
-  patternMismatch: true,
  tooLong: false,
-  tooShort: false,
-  typeMismatch: false,
-  valid: false
}
```

donc

- Si pas de value dans le champ : validation native via attribut required.
- Si value et pas de `data-validation-url`: validation native à travers les atrtibuts html.
- Si value et `data-validation-url`, demande de validation au serveur.

#6 - 30 mars 2023 18:53 - Frédéric Péters

> Je pense qu'il faut partir dans l'autre sens : il faut d'abord considérer la validation par requête vers le serveur, qui peut couvrir tous les cas, puis petit à petit, pour ce qui pourrait être fait en local, le faire. Bref pas de liste de ce qui doit être fait sur le serveur, la liste c'est "tout sauf ce qui peut être fait en local".

Dommage de charger le serveur par plein de micro-requêtes qui pourraient majoritairement être évitées. Mais OK.

Il s'agit biend'éviter ça mais "petit à petit", pour ne pas bloquer les développements ici sur une réimplémentation javascript de la validation des IBAN, et autres.

> Sachant que `valueMissing` peut être géré côté browser (...)

Partiellement seulement, pas pour les blocs de champs je crois. (mais on peut exclure ça pour le moment)

> On conserve l'idée des messages d'erreurs dans l'html et le serveur retourne simplement :

Ça me semble bizarre d'avoir une structure qui empile toutes les possibilités puis avoir dans le code un boucle pour trouver celle qui a été choisie,

```
let errorType
for (const key in validityState) {
  if (validityState[key]) {
    errorType = key
    break
  }
}
```

autant renvoyer directement le type d'erreur, en suivant le format des messages de Publik, ça passerait avec : {'err': 1, 'err_code': 'pattern_mismatch'}, et sur base de l'err_code tu peux trouver le message correspondant ? Je ne capte pas dans le code proposé où viendrait l'interrogation du serveur, donc pas vraiment où il y aurait un gain à suivre la structure de validityState.

#7 - 31 mars 2023 18:45 - Frédéric Péters

J'avais commencé à regarder pour poser de manière statique les différentes erreurs possibles dans des balises <template> mais ça ne va pas être possible pour tout, la réponse du serveur doit pouvoir contenir un message personnalisé, et à partir de là, autant tout le temps juste fournir le message.

https://git.entrouvert.org/entrouvert/wcs/commit/b99b04c19f3aca75ab21eaae2a664644389678b9

#8 - 03 avril 2023 10:24 - Thomas Jund (congés, retour le 29/04)

> Ça me semble bizarre d'avoir une structure qui empile toutes les possibilités puis avoir dans le code un boucle pour trouver celle qui a été choisie,

Oui, dans le proto, je retourne pour le moment la première erreur que je trouve. Mais si toutes les erreurs sont exposées, c'est bien pour afficher potentiellement plusieurs messages d'erreurs en cas d'erreurs multiples, ce cas de figure est plutôt rare mais peut arriver théoriquement, par exemple avec une validation pattern + min-length comme avec le code postal par exemple.

> la réponse du serveur doit pouvoir contenir un message personnalisé

Je ne suis pas sûr de comprendre, Il n'est pas possible de générer la personnalisation au chargement de la page, parce que le message est potentionnement composé de contenu de champs ({{ form_var_* }} ?)
C'est un peu bizzare, cela signifie que le champ à remplir doit forcement l'être après un autre (qui n'est pas forcement obligatoire, ni conditionné), sinon le message d'erreur sera incomplet, ou il y a formement une condition d'affichage qui lie ces champs ?

#9 - 03 avril 2023 11:10 - Frédéric Péters

> Je ne suis pas sûr de comprendre, Il n'est pas possible de générer la personnalisation au chargement de la page, parce que le message est potentionnement composé de contenu de champs ({{ form_var_* }} ?)

Il n'y a pas de possibilité (facile) d'une liste exhaustive des erreurs potentielles côté serveur, et comme il s'agit de tests qui devront nécessairement avoir lieu côté serveur, il me semble nettement plus évident sur ce moment d'également fournir le message associé. Et à partir du moment où sur les validations côté serveur on a des cas avec le message fourni au moment de l'interrogation, on doit pouvoir le gérer, et à ce compte autant fournir dans tous les cas le message précis.

> C'est un peu bizzare, cela signifie que le champ à remplir doit forcement l'être après un autre

Mais ce champ pourrait être d'une page précédente, ou (surtout) l'information pourrait venir du champ lui-même (je n'ai pas d'exemple à donner parce que justement point précédent liste exhaustive compliquée à établir).

~~

J'ai posé https://git.entrouvert.org/entrouvert/wcs/commits/branch/wip/75724-error-template-markup avec une série de commits brouillons/expérimentations autour de tout ça.

Il y a dedans l'ajout de <template> avec les textes d'erreur, les codes d'erreur, puis aussi

- https://git.entrouvert.org/entrouvert/wcs/commit/c78f71d819714cee24489e35b06ef3e288722a79 qui ajoute un attribut pour les validations "complexes", pour permettre côté js de savoir qu'il faut une validation iban, par exemple,
- https://git.entrouvert.org/entrouvert/wcs/commit/df17bed730f444f6e38742e191c43c98840643da qui ajoute data-check-condition-url, avec l'URL d'une API locale, et un bouton moche pour voir ça en action.

#10 - 04 avril 2023 10:48 - Frédéric Péters

Dans la branche aussi maintenant un <template> avec le message d'erreur de validation personnalisé. Et le <template id="form_error_tpl"> présent dans codepen.io.

#11 - 12 avril 2023 14:52 - Thomas Jund (congés, retour le 29/04)

- Projet changé de Intégrations graphiques Publik à w.c.s.

#12 - 13 avril 2023 15:47 - Robot Gitea

- Statut changé de En cours à Solution proposée

Thomas Jund (tjund) a ouvert une pull request sur Gitea concernant cette demande :

- URL : https://git.entrouvert.org/entrouvert/wcs/pulls/240
- Titre : Js: add live field validation (#75724)
- Modifications : https://git.entrouvert.org/entrouvert/wcs/pulls/240/files

#13 - 13 avril 2023 15:49 - Thomas Jund (congés, retour le 29/04)

J'ai poussé une première version du JS.
À tester.

#14 - 14 avril 2023 10:07 - Frédéric Péters

- Lié à Development #76632: code serveur pour la validation à la volée ajouté

#15 - 17 avril 2023 12:25 - Robot Gitea

- Statut changé de Solution proposée à En cours

- Assigné à changé de Thomas Jund (congés, retour le 29/04) à Frédéric Péters

Frédéric Péters (fpeters) a ouvert une pull request sur Gitea concernant cette demande :

- URL : https://git.entrouvert.org/entrouvert/wcs/pulls/244
- Titre : WIP: validation js revue et à revoir
- Modifications : https://git.entrouvert.org/entrouvert/wcs/pulls/244/files

#16 - 18 avril 2023 14:13 - Robot Gitea

- Statut changé de En cours à Solution proposée

#17 - 18 avril 2023 14:23 - Robot Gitea

- Statut changé de Solution proposée à Solution validée

Lauréline Guérin (lguerin) a approuvé une pull request sur Gitea concernant cette demande :

- URL : https://git.entrouvert.org/entrouvert/wcs/pulls/244

#18 - 18 avril 2023 14:29 - Robot Gitea

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

Frédéric Péters (fpeters) a mergé une pull request sur Gitea concernant cette demande :

- URL : https://git.entrouvert.org/entrouvert/wcs/pulls/244
- Titre : validation js revue et à revoir
- Modifications : https://git.entrouvert.org/entrouvert/wcs/pulls/244/files

#19 - 20 avril 2023 20:14 - Transition automatique

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

#20 - 25 juin 2023 04:42 - Transition automatique

Automatic expiration