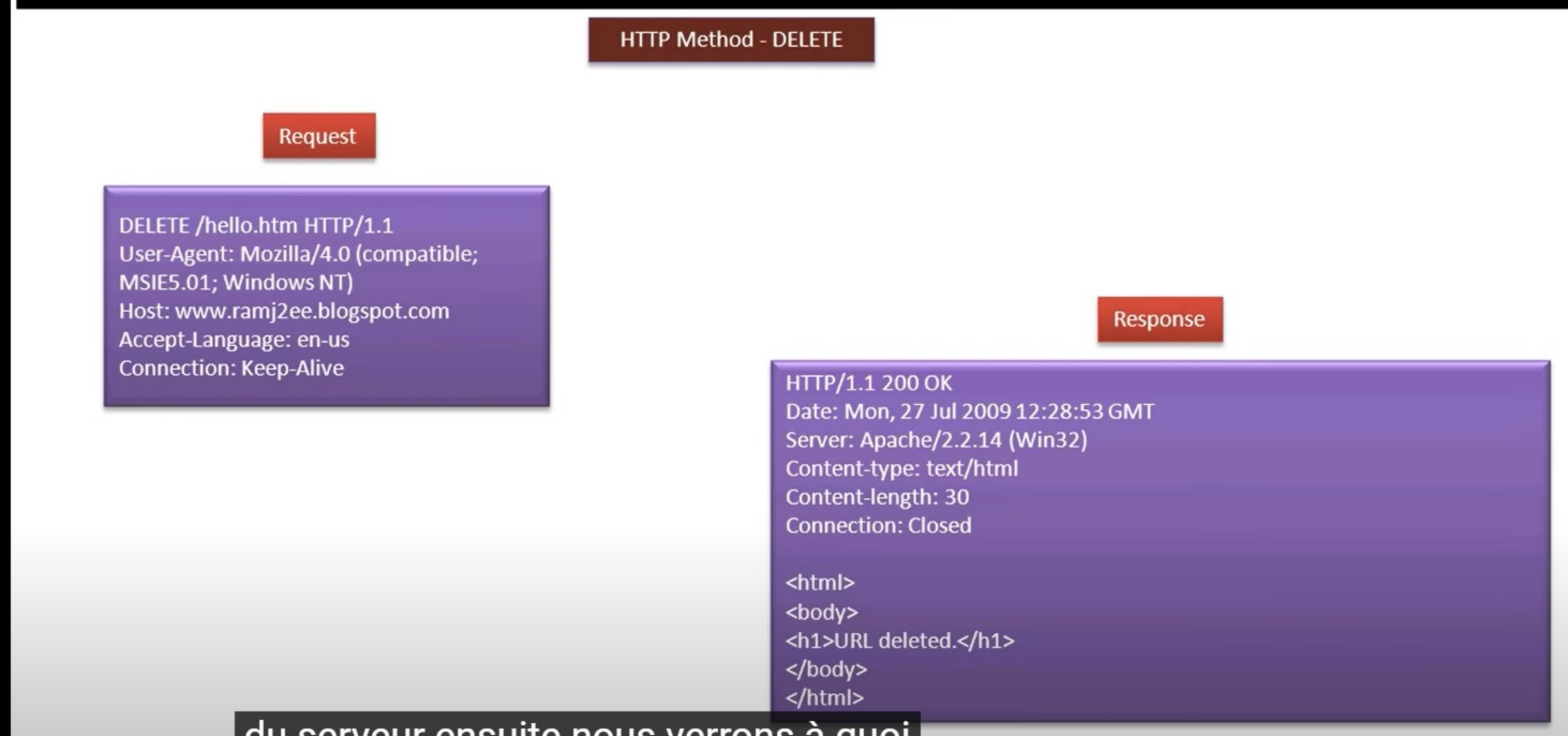
HTTP Method - DELETE
Request
DELETE /hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.ramj2ee.blogspot.com
Accept-Language: en-us
Connection: Keep-Alive
Response
HTTP/1.1 200 OK
Date: Mon, 27 Jul 2009 12:28:53 GMT
Server: Apache/2.2.14 (Win32)
Content-type: text/html
Content-length: 30
Connection: Closed
<html>
<body>
<h1>URL deleted.</h1>
</body>
</html>
du serveur ensuite nous verrons à quoi

Il s'agit d'un instantané archivé du traqueur de bogues bugzilla public du W3C, mis hors service en avril 2019. Veuillez consulter la page d'accueil pour plus de détails.

Bogue 10671
Résumé : envisagez d'ajouter la prise en charge de PUT et DELETE en tant que méthodes de formulaire

Statut : RÉSOLU WONTFIX

Alias: Aucun

Produit: GT HTML
Composant: Spécification LC1 HTML5 (afficher d'autres bogues)
Version: non spécifié
Matériel: Tout Tout

J'ai de l'importance : Amélioration P3
Jalon cible : ---
Cessionnaire: donateur
Contact AQ : Liste d'archives HTML WG Bugzilla

URL : http://dev.w3.org/html5/spec/associat...
Tableau blanc :
Mots clés: NE, problème de suivi

Doublons (2) : ~~12647~~ ~~22695~~ (voir comme liste de bogues)
Dépend de:
Blocs :

Signalé: 2010-09-21 15:19 UTC par Julian Reschke
Modifié: 2013-07-17 13:49 UTC (Histoire)
Liste CC : 19 utilisateurs

andrew.gibson
annevk
brunoaiss
dragon d'argile
cmhjones

Voir également:

Pièces jointes

Julien Reschke 2010-09-21 15:19:02 UTC — La description

```
Il semble que nous ne comprenions pas actuellement comment PUT et DELETE seront utiles
pour les formulaires HTML.

Pour DELETE, il est en effet facile de créer une requête utile. Cependant, les
implémentations de serveur répondent généralement avec 200 et un corps de réponse minimal
("supprimé") ou 204 (pas de contenu). Il n'est donc pas clair comment cela peut être
utilisé dans une application Web.

Pour PUT, il semble qu'il n'y ait pas de cas d'utilisation réel tant que la page Web n'a
pas un contrôle total sur la charge utile et peut également définir le type de contenu.

Veuillez envisager de supprimer cette fonctionnalité jusqu'à ce que vous compreniez mieux
à quoi elle sert.
```

Anne 2010-09-24 09:21:42 UTC — Commentaire 1

```
Je pense que cela a du sens. La principale raison pour laquelle nous les avons ajoutés
était à cause de XForms et cela n'a jamais vraiment été une bonne raison.
```

Ian 'Hixie' Hickson 2010-09-30 07:58:31 UTC — Commentaire 2

```
RÉPONSE DE LA RÉDACTION : ceci est une réponse de la rédaction à votre commentaire. Si
vous êtes satisfait de cette réponse, veuillez changer l'état de ce bogue en FERMÉ. Si
vous avez des informations supplémentaires et souhaitez que l'éditeur reconsidère,
veuillez rouvrir ce bogue. Si vous souhaitez faire remonter le problème au groupe de
travail HTML complet, veuillez ajouter le mot-clé TrackerRequest à ce bogue et suggérer un
titre et un texte pour le problème de suivi ; ou vous pouvez créer vous-même un problème
de suivi, si vous en êtes capable. Pour plus de détails, consultez ce document :
    http://dev.w3.org/html5/decision-policy/decision-policy.html

Statut : Accepté
Description du changement : voir les différences ci-dessous
Justification : D'accord avec les commentaires du journaliste.
```

contributeur 2010-09-30 07:59:10 UTC — Commentaire 3

```
Enregistré en tant que révision WHATWG r5566.
Commentaire d'enregistrement : Mettez method=DELETE et method=PUT au repos.
http://html5.org/tools/web-apps-tracker?from=5565&to=5566
```

Mike Amundsen 2010-10-06 20:18:19 UTC — Commentaire 4

```
L'exécution de PUT et DELETE pour modifier les ressources sur le serveur d'origine est
simple pour les navigateurs Web modernes utilisant l'objet XmlHttpRequest. Pour les
interactions de navigateur non scriptées, ce n'est pas si simple. En règle générale, les
développeurs et les frameworks finissent par créer des solutions de contournement qui
imitent l'interaction HTTP PUT/DELETE + Etag en utilisant un "POST FORM" couplé à un code
côté serveur spécialisé pour trier le cas particulier et agir comme si la méthode HTTP
appropriée était utilisée. dans la requête :

<!-- solution de contournement typique pour prendre en charge PUT/DELETE -->
<form method="poster" action="...">
  <input type="hidden" name="override_method" value="put" />
  <input type="hidden" name="hashtag" value="..." />
  ...
</form>

Ce modèle est requis si souvent que plusieurs frameworks/bibliothèques Web couramment
utilisés[1][2][3][4] ont créé une solution de contournement « intégrée ». Notez que
certaines bibliothèques incluent la prise en charge d'une valeur de concurrence et
d'autres non.

Autres considérations:
- L'utilisation de POST comme tunnel au lieu d'utiliser PUT/DELETE peut entraîner des
erreurs de mise en cache (par exemple, les réponses POST peuvent être mises en cache[5],
les réponses PUT ne le sont pas[6], les réponses DELETE ne le sont pas[7])
- L'utilisation d'une méthode non idempotente (POST) pour effectuer une opération
idempotente (PUT/DELETE) complique la récupération en raison de pannes de réseau (par
exemple "Est-il sûr de répéter cette action ?").
- Le tunneling conduit à une "visibilité" réduite du protocole d'application sur le
réseau, ce qui complique l'audit des demandes, la sécurité au niveau des autorisations,
etc.
- L'ajout de la prise en charge de PUT/DELETE (avec l'inclusion des en-têtes If-Match/If-
Unmodified-Since) peut réduire les occurrences du problème de concurrence "Lost Update".

[1] http://guides.rubyonrails.org/form_helpers.html#how-do-forms-with-put-or-delete-
methods-work
[2] http://book.cakephp.org/view/183 /Creating-Forms#options-type-184
[3] http://djangosnippets.org/snippets/174/
[4] http://geekswithblogs.net/michelotti/archive/2010/01/08/implementing-a- delete-link-
with-mvc-2-and-httpmethodoverride.aspx
[5] http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.5
[6] http://www. w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.6
[7] http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html#sec9.7
```

Julien Reschke 2010-10-07 07:34:07 UTC — Commentaire 5

```
(En réponse au commentaire #4 )

Cela semble être un commentaire général sur la qualité de PUT et DELETE par rapport à tout
faire avec POST. Je suis d'accord avec ça.

La question délicate est de savoir comment * utiliser * PUT et DELETE avec des formulaires
HTML. Le bogue a été soulevé parce que je pense que la spécification (telle qu'elle était
à l'époque) n'était pas assez spécifique pour que cela fonctionne, et donc une adoption
précoce (comme dans FF4) rendrait très difficile de faire la bonne chose plus tard.
```

En ce qui concerne la mise en cache: je crois comprendre que la mise en cache est la même pour PUT, POST, DELETE et de nombreuses autres méthodes. Vous voudrez peut-être jeter un œil à HTTPbis, partie 6 (http://greenbytes.de/tech/webdav/draft-ietf-httpbis-p6-cache-11.html)

Mike Amundsen 2010-10-07 12:37:27 UTC Commentaire 6

(En réponse au commentaire #5)
> (En réponse au commentaire #4)
>
> Cela semble être un commentaire général sur la qualité de PUT et DELETE vs
> tout faire avec POST. Je suis d'accord avec ça.

J'ai posté ici sur la base d'une suggestion "d'expliquer en détail..." mon exemple de cas d'utilisation de PUT (http://krijnhoetmer.nl/irc-logs/whatwg/20101006#l-662).

>
> La question délicate est de savoir comment *utiliser* PUT et DELETE avec les formulaires

Je pense que les utilisations de PUT/DELETE dans les quatre frameworks que j'ai cités sont claires, n'est-ce pas ?

> Le bug a été signalé car je pense que la spécification (telle qu'elle était à l'époque) n
> suffisamment spécifique pour que cela fonctionne, et donc adoption précoce (comme dans FF
> rendrait très difficile de faire la bonne chose plus tard.

J'ai compris qu'il avait été supprimé "jusqu'à ce qu'il y ait une meilleure compréhension à quoi ça sert." Ai-je mal interprété votre remarque dans la description du bogue ?

>
> En ce qui concerne la mise en cache : je crois comprendre que la mise en cache est la mêm
> pour PUT, POST, DELETE et bien d'autres méthodes. Vous voudrez peut-être jeter un œil à
> HTTPbis, partie 6
> (http://greenbytes.de/tech/webdav/draft-ietf-httpbis-p6-cache-11.html)

Je connais ce matériel. Il ne m'est pas clair (d'après votre commentaire ici) comment le contenu de la partie 6 affecte mes remarques sur la mise en cache de POST par HTTP 1.1 par rapport à PUT et DELETE. Plus précisément, je ne vois aucun changement dans la partie 2 (http://greenbytes.de/tech/webdav/draft-ietf-httpbis-p2-semantics-11.html) qui indique un changement dans la mise en cache de POST, PUT ou SUPPRIMER. Pouvez-vous m'aider à m'assurer que je comprends votre point ici ?

Julien Reschke 2010-10-07 13:08:21 UTC Commentaire 7

(En réponse au commentaire #6)
> J'ai posté ici sur la base d'une suggestion pour "expliquer en détail..." mon exemple de
> cas d'utilisation pour PUT (http://krijnhoetmer.nl/irc-logs/whatwg/20101006#l-662).

Acquittement. Et juste pour référence; J'ai ouvert ce bogue parce que l'implémentation (maintenant désactivée/supprimée) dans FF4 avait deux bogues, dont l'un concernait le calcul d'un mauvais URI cible, l'autre concernant la gestion des redirections.

Le premier était un bogue par rapport à ce que disait HTML5, le second n'est pas spécifié dans HTML5, donc le code a simplement réutilisé le comportement XMLHttpRequest (qui, je pense, est également cassé). Cela m'a rendu nerveux à l'idée que cela soit fait * correctement *.

> > La question délicate est de savoir comment * utiliser * PUT et DELETE avec les formulai
>
> Je pense que les utilisations de PUT/DELETE dans les quatre frameworks que j'ai cités son
> correct ?

Autant que je sache, ceux-ci utilisent POST pour tunnel PUT/DELETE. Il n'y a rien de mal à cela en principe.

Peut-être que le désaccord est basé sur d'où nous venons? J'utilise des serveurs qui prennent en charge PUT et DELETE tout le temps. Ces serveurs envoient 200/201/204 avec un message d'état minimal lorsque tout va bien. Comment sont-ils censés être utilisés à partir d'un formulaire HTML ? Peut-être qu'ils ne sont pas censés le faire ?

> > Le bogue a été soulevé parce que je pense que la spécification (telle qu'elle était à l
> > suffisamment spécifique pour que cela fonctionne, et donc adoption précoce (comme dans
> > rendrait très difficile de faire la bonne chose plus tard.
>
> J'ai compris qu'il a été supprimé "jusqu'à ce qu'il y ait une meilleure compréhension
> à quoi ça sert." Ai-je mal interprété votre remarque dans le bug
> descriptif ?

Je ne pense pas.

Je pense que ce qu'il faut, c'est une histoire sur la façon dont PUT et DELETE vont être utilisés dans la pratique. En particulier, si les serveurs qui prennent déjà en charge PUT et DELETE doivent être modifiés afin que cela puisse être utilisé à partir de formulaires HTML ; l'envoi de la requête est simple, mais ce qui est moins clair, c'est quels codes de réponse sont pris en charge et comment ils affectent l'application Web.

> Je connais ce matériel. Ce n'est pas clair pour moi (d'après votre commentaire ici)
> comment le contenu de la partie 6 affecte mes remarques sur la mise en cache de POST par
> 1.1 contre PUT et DELETE. Plus précisément, je ne vois aucun changement dans la partie 2
> (http://greenbytes.de/tech/webdav/draft-ietf-httpbis-p2-semantics-11.html) que
> indiquer un changement dans la capacité de cache de POST, PUT ou DELETE. Pouvez-vous m'ai
> assurez-vous que je comprends votre point ici?

Travaux en cours...

Il ne devrait pas y avoir de cas particuliers sauf pour GET/HEAD. Voir < http://trac.tools.ietf.org/wg/httpbis/trac/ticket/139 > (nous devrons peut-être rouvrir ce problème, veuillez suivre la liste de diffusion HTTP WG si vous pensez qu'il faut plus être terminé).

Mike Amundsen 2010-10-07 14:49:07 UTC Commentaire 8

FWIW, je sens un peu de changement de contexte ici dans les raisons de l'abandon de cette fonctionnalité :

- "l'implémentation dans FF4 avait deux bugs" (Compris)
- "Il n'y a rien de mal... en principe" (je ne partage pas cet avis<g>)
- "ce qu'il faut, c'est une histoire sur la façon dont PUT et DELETE vont être utilisés dans la pratique." (Fourni)
- "...les serveurs envoient 200/201/204 avec un message d'état minimal lorsque tout va bien. Comment sont-ils censés être utilisés à partir d'un formulaire HTML ?" (Le même que lors de leur retour par la POSTE)
- "...si les serveurs qui supportent déjà PUT et DELETE doivent être modifiés..." (je ne vois pas pourquoi c'est un problème et j'aimerais voir un cas résolu où c'est un problème potentiel).

Il semble que la liste de bogues ne soit pas le lieu de discussion sur ces éléments et je n'ai pas la possibilité de publier sur html-public afin de poursuivre cela. Si vous souhaitez le poursuivre, vous pouvez me contacter sur IRC ou directement par e-mail (mamund -AT- yahoo -DOT- com).

Merci pour votre temps.
MCA

(En réponse au commentaire #7)
> (En réponse au commentaire #6)
> > J'ai posté ici sur la base d'une suggestion pour "expliquer en détail..." mon exemple d
> > cas d'utilisation pour PUT (http://krijnhoetmer.nl/irc-logs/whatwg/20101006#l-662).
>
> Acq. Et juste pour référence; J'ai ouvert ce bogue parce que le
> L'implémentation (maintenant désactivée/supprimée) dans FF4 avait deux bogues, dont l'un
> à propos d'une mauvaise URI cible en cours de calcul, l'autre par rapport à la gestion

> redirige.
>
> Le premier était un bogue par rapport à ce que disait HTML5, le second ne l'est pas
> spécifié dans HTML5, donc le code vient de réutiliser le comportement XMLHttpRequest (que
> pense que c'est cassé aussi). Cela m'a rendu nerveux à l'idée que cela se fasse
> *droit*.
>
> > > La question délicate est de savoir comment * utiliser * PUT et DELETE avec les formul
> > >
> > > Je pense que les utilisations de PUT/DELETE dans les quatre frameworks que j'ai cités s
> > > correct ?
> >
> > Autant que je sache, ceux-ci utilisent POST pour tunnel PUT/DELETE. Il n'y a rien
> > tort à ce sujet en principe.
> >
> > Peut-être que le désaccord est basé sur l'endroit d'où nous venons ? J'utilise des serveu
> > supportez PUT et DELETE tout le temps. Ces serveurs envoient 200/201/204 avec un
> > message d'état minimal lorsque tout va bien. Comment sont-ils censés être utilisés
> > depuis un formulaire HTML ? Peut-être qu'ils ne sont pas censés le faire ?
> >
> > > > Le bogue a été soulevé parce que je pense que la spécification (telle qu'elle était à
> > > > suffisamment spécifique pour que cela fonctionne, et donc adoption précoce (comme dan
> > > > rendrait très difficile de faire la bonne chose plus tard.
> > > >
> > > J'ai compris qu'il a été supprimé "jusqu'à ce qu'il y ait une meilleure compréhension
> > > à quoi ça sert." Ai-je mal interprété votre remarque dans le bogue
> > > descriptif ?
> >
> > Je ne pense pas.
> >
> > Je pense que ce qu'il faut, c'est une histoire sur la façon dont PUT et DELETE vont être
> > en pratique. En particulier, si les serveurs prenant déjà en charge PUT et DELETE
> > doivent être modifiés pour pouvoir être utilisés à partir de formulaires HTML ; l'envoi d
> > simple, mais ce qui est moins clair, c'est quels codes de réponse sont pris en charge et
> > affecter l'application Web.
> >
> > > Je connais ce matériel. Ce n'est pas clair pour moi (d'après votre commentaire ici)
> > > comment le contenu de la partie 6 affecte mes remarques sur la mise en cache de POST pa
> > > 1.1 contre PUT et DELETE. Plus précisément, je ne vois aucun changement dans la partie
> > > (http://greenbytes.de/tech/webdav/draft-ietf-httpbis-p2-semantics-11.html) que
> > > indique un changement dans la capacité de cache de POST, PUT ou DELETE. Pouvez-vous m'a
> > > assurez-vous que je comprends votre point ici ?
> >
> > Travaux en cours...
> >
> > Il ne devrait pas y avoir de cas particuliers sauf pour GET/HEAD. Voir
> > < http://trac.tools.ietf.org/wg/httpbis/trac/ticket/139 > (nous devrons peut-être rouvrir
> > ce problème, veuillez suivre la liste de diffusion HTTP WG si vous pensez plus
> > à faire).

Anne 2010-10-07 14:54:00 UTC Commentaire 9

Ma principale préoccupation est l'incohérence avec GET/POST en ce qui concerne les
demandes d'origine croisée en plus de celles-ci étant beaucoup moins utilisées que
GET/POST. S'ils ne nécessitaient pas de traitement spécial, le choix serait facile, mais
il n'est pas clair maintenant si la complexité supplémentaire est justifiée.

Donc, à mon avis, nous devrions nous concentrer sur cette fonctionnalité pour le moment
jusqu'à ce qu'il soit plus clair ce qui est exactement nécessaire.

Cameron Jones 2011-03-31 13:53:16 UTC Commentaire 10

(En réponse au commentaire #9)
> Ma principale préoccupation est l'incohérence avec GET/POST en ce qui concerne l'origine
> les requêtes en plus de celles-ci étant beaucoup moins utilisées que GET/POST. S'ils ne l
> nécessite une manipulation particulière le choix serait facile, mais maintenant ce n'est
> si la complexité supplémentaire est justifiée.

>
> Donc, à mon avis, nous devrions jeter un coup d'œil sur cette fonctionnalité pour l'insta
> ce qu'il faut exactement.

Je ne suis pas un expert des requêtes d'origine croisée ou de la complexité de la mise en
œuvre, mais je risquerais que celles-ci utilisent beaucoup moins que PUT et DELETE.

en réponse à une question ouverte sur public-html-comments, ce bogue a été examiné et doit
être rouvert en fonction des commentaires fournis dans le fil :

http://lists.w3.org/Archives/Public/public-html-comments/2011Mar/0006.html

Pour paraphraser l'exigence fondamentale de prise en charge de ces méthodes sur les
formulaires :
 afin qu'un véritable humain puisse interagir nativement avec un système Restful en
utilisant du HTML simple, sans ajax ou d'autres solutions de contournement.

Le formulaire est le seul élément d'interface utilisateur capable d'initier des requêtes
HTTP personnalisées, sans que l'éventail complet des méthodes ne soit pris en charge,
aucun autre moyen n'est disponible pour interagir avec les services conformes à HTTP.

Marc 2011-04-05 11:26:09 UTC Commentaire 11

J'écris ça rapidement sur le dessus de ma tête...

J'y ai toujours pensé comme ça avec les méthodes des formulaires HTML - récapitulatif
rapide :

POST : les données du formulaire sont collectées (telles que définies par HTML), encodées
dans le "enctype" donné et soumises en tant que corps de la requête à l'URL dans
l'attribut "action".

GET : cette méthode, telle que définie par HTTP, ne doit pas avoir de corps de requête (je
pense qu'il est conseillé aux serveurs de supprimer tout corps, si je me souviens bien),
donc les données du formulaire seront soumises en tant que chaîne de requête de l'URL
donnée dans l'attribut "action". Le format d'encodage "par défaut" est "application/x-www-
form-urlencoded".

Donc, fondamentalement, _GET_ est le cas particulier, car la manière naturelle d'envoyer
des données à un serveur serait le corps de la requête. Ce qui signifie essentiellement
pour les méthodes PUT et DELETE :

PUT : pratiquement identique à POST, du point de vue du formulaire.

DELETE : tout comme GET, cette méthode ne doit pas avoir de corps de requête. Doit être
géré de la même manière que GET (regrouper les données du formulaire dans la chaîne de
requête).

Voici mes idées de brainstorming pour enrichir les formulaires HTML :

Étant donné que HTTP définit quelques méthodes standard, mais que l'ajout de méthodes
personnalisées est absolument autorisé, je pense que les formulaires HTML ne doivent pas
restreindre les méthodes dans l'attribut "method". Les méthodes inconnues (pas dans la
spécification HTTP d'origine) doivent être gérées de la même manière que POST ou PUT. Ce
serait par exemple. permettre à la méthode PATCH définie dans la RFC 5789 de fonctionner
(mais qui a probablement besoin d'un nouveau "enctype" de toute façon).

Nous pourrions également ajouter un attribut qui définit "où" les données du formulaire
sont envoyées, quelque chose comme

<form where="query-string" method="FOO"> (les méthodes GET/DELETE impliquent "query-
string")
<form where="body" method="BAR"> (les méthodes POST/PUT impliquent "body")

et peut-être même

<form where="header" method="BAR"> (pour envoyer des données de formulaire sous forme
d'en-têtes HTTP supplémentaires à la place ; je n'ai pas pensé aux implications de
sécurité avec cela, juste une idée)

Cela permettrait également d'envoyer des méthodes personnalisées via une chaîne de requête
(nous pourrions également appeler l'attribut "via", "position" ou quelque chose).

Je considère également DELETE comme un cas particulier, qui pourrait être traité d'une
autre manière. Étant donné qu'une requête GET déclenchée par l'utilisateur en HTML est

également effectuée en cliquant sur un lien hypertexte (une fonctionnalité qui peut être émulée à l'aide d'un formulaire GET HTML), et que les requêtes DELETE pointeront probablement principalement vers une ressource spécifique où il n'est pas logique d'ajouter un chaîne de requête (par exemple http://example.com/users/42), nous pourrions peut-être étendre les éléments button/input :

<button type="delete" url=" http://example.com/users/42 ">Supprimer l'utilisateur Douglas</button>

Un tel élément n'a pas besoin d'être contenu dans un élément de formulaire et envoie simplement la demande DELETE à l'URL spécifiée. Les clients peuvent par défaut afficher une boîte de dialogue de confirmation avant de soumettre une requête DELETE.

D'autres choses auxquelles j'ai pensé sont comment cela pourrait être fait, par exemple. envoyer un fichier (disons une "image/png") à une URL avec son type MIME correct en utilisant des formulaires en HTML (sans l'envelopper dans un corps "multipart/form-data"), ce qui pourrait être intéressant pour PUT. Peut-être un attribut comme "only":

<form method="PUT" action="..." enctype="image/png" only="foobar">
<input type="file" name="foobar" accept="image/png" />
<button type="submit">Télécharger</button>
</form>

"enctype"/"accept" pourrait être omis pour permettre l'envoi de données formatées arbitrairement (et laisser le client faire de son mieux pour deviner le type MIME du fichier téléchargé).

Avec cela, nous pourrions commencer à créer des interfaces dans des formulaires HTML pour une interaction facile avec les services Web RESTful.

Une dernière chose à propos de la réponse du serveur : répondre avec un statut "200 OK" est parfaitement valable pour les requêtes PUT et DELETE. BTW, la dernière fois que j'ai vérifié, Firefox respecte le statut "204 No Content" tel que défini par HTTP, en envoyant le formulaire et en ne réinitialisant pas la vue du document. Je connais l'événement "onsubmit" déclenché par les navigateurs à utiliser avec JavaScript, mais existe-t-il également quelque chose comme un événement "onsubmitcomplete" ? Il serait intéressant de savoir comment d'autres navigateurs gèrent ces états de réponse ou des états de réponse similaires. Les navigateurs doivent certainement en quelque sorte informer l'utilisateur sur "204 No Content" en affichant une boîte de notification asynchrone.

Paolo Perrotta 2011-04-09 02:10:48 UTC Commentaire 12

> Pour paraphraser l'exigence fondamentale d'appuyer ces méthodes sur
> formulaires :
> pour qu'un vrai humain puisse interagir nativement avec un système Restful en utilisant
> HTML simple, pas d'ajax ou d'autres solutions de contournement.

En tant que développeur, je pense que c'est une raison suffisante pour avoir PUT/DELETE dans les formulaires. Cette fonctionnalité serait utile pour la documentation, le débogage, le prototypage, les tests manuels et les tests automatisés. Les ramifications pour le domaine des services Web pourraient aller assez loin. L'auto-documentation en particulier est perçue comme un point faible des services REST par rapport aux services SOAP.

Je vois également un autre avantage de PUT/DELETE dans les formulaires : ils permettent au navigateur de savoir que vous faites quelque chose d'idempotent. Lorsque vous émettez un POST via un formulaire, puis que vous appuyez sur le bouton "Précédent" (ou que vous appuyez sur "Actualiser" avant que la réponse ne soit parvenue), le navigateur n'a d'autre choix que de vous montrer une boîte de dialogue de confirmation effrayante, qui rend régulièrement perplexe les non- utilisateurs techniques. Avec PUT/DELETE, les navigateurs peuvent choisir de simplement soumettre à nouveau, ou au moins de vous montrer une boîte de dialogue plus informative. Ajax n'aiderait pas là-bas, car cela se passe derrière le dos du navigateur.

Juste mes 2 centimes.

Cameron Jones 2011-04-11 17:45:22 UTC Commentaire 13

Proposition supplémentaire soumise :

http://lists.w3.org/Archives/Public/public-html/2011Apr/0259.html

Cette proposition introduit la manipulation des en-têtes HTTP via des champs de formulaire, la construction d'en-têtes d'authentification HTTP et la prise en charge de la méthode HTTP CONNECT.

Anne 2011-06-24 10:49:04 UTC Commentaire 14

*** ~~Le bogue 12647~~ a été marqué comme doublon de ce bogue. ***

Michael[tm] Smith 2011-08-04 05:14:42 UTC Commentaire 15

composant de déplacement de masse vers LC1

Ian 'Hixie' Hickson 2011-12-02 16:45:03 UTC Commentaire 16

RÉPONSE DE LA RÉDACTION : ceci est une réponse de la rédaction à votre commentaire. Si vous êtes satisfait de cette réponse, veuillez changer l'état de ce bogue en FERMÉ. Si vous avez des informations supplémentaires et souhaitez que l'éditeur reconsidère, veuillez rouvrir ce bogue. Si vous souhaitez faire remonter le problème au groupe de travail HTML complet, veuillez ajouter le mot-clé TrackerRequest à ce bogue et suggérer un titre et un texte pour le problème de suivi ; ou vous pouvez créer vous-même un problème de suivi, si vous en êtes capable. Pour plus de détails, consultez ce document :
http://dev.w3.org/html5/decision-policy/decision-policy.html

Statut : Refusé
Description du changement : aucun changement de spécification
Justification : PUT en tant que méthode de formulaire n'a aucun sens, vous ne voudriez pas PUT une charge utile de formulaire. DELETE n'a de sens que s'il n'y a pas de charge utile, donc cela n'a pas beaucoup de sens non plus avec les formulaires.

Julien Reschke 2011-12-02 18:06:02 UTC Commentaire 17

> Justification : PUT en tant que méthode de formulaire n'a aucun sens, vous ne voudriez pa
> charge utile. (...)

Pourquoi pas?

> (...) DELETE n'a de sens que s'il n'y a pas de charge utile, donc ça n'a pas de sens
> beaucoup de sens avec les formes non plus.

Cela n'a aucun sens, sinon GET ne fonctionnerait pas non plus à partir de formulaires.

Cameron Jones 2011-12-03 11:05:52 UTC Commentaire 18

> Justification : PUT en tant que méthode de formulaire n'a aucun sens, vous ne voudriez pa
> charge utile. DELETE n'a de sens que s'il n'y a pas de charge utile, donc ça ne fait pas
> beaucoup de sens avec les formes non plus.

au lieu d'avoir à justifier pourquoi PUT est valide, vous devriez expliquer en quoi la spécification http "n'a pas beaucoup de sens". une spécification est soit correcte, soit incorrecte, compte tenu du nombre de services existants utilisant PUT et elle est, à tout le moins, prouvée correcte dans ce cas par une utilisation sur le terrain.

DELETE nécessite le même support que GET pour permettre la construction paramétrique de l'URI de la requête. Par exemple:

<form action=" http://example.org/user " method="delete" if-match="*">
  <input name="hat-size" type="text" value="" />
  <type d'entrée="soumettre" />
</form>

Pour que l'utilisateur SUPPRIME les chapeaux suivants :

SUPPRIMER /user?hat-size=small HTTP/1.1
Hébergeur : www.example.org
Si-correspondance : "*"

beaucoup de travail collaboratif s'est déroulé depuis la réouverture de ce rapport de bogue, y compris l'introduction de nouvelles informations et le développement des complexités de mise en œuvre et des exigences du cas.

ce travail doit être engagé et référencé sinon la résolution de ce rapport de bogue est prématurée et mal informée.

Cameron Jones 2012-01-12 15:21:31 UTC Commentaire 19

Grâce à une discussion plus approfondie sur public-html-comments[1], je demande que ce bogue soit escaladé dans le processus de problème du tracker. Je recommande que les deux problèmes identifiés suivants soient créés en tant que problèmes distincts :

Problème 1 :
Améliorer la génération de requêtes http à partir de formulaires

Les formulaires doivent être capables de définir et de construire des représentations de requête http complètes pour toutes les méthodes afin que le code HTML puisse interagir avec les services http conformes.

Problème 2 :
Définir le comportement de gestion des réponses http de l'agent utilisateur

La gestion par l'agent utilisateur des réponses http n'est actuellement pas définie, ce qui a conduit à des comportements divergents[2] du fait d'une interprétation indépendante de la spécification http. Le comportement attendu doit être défini dans html pour permettre aux serveurs de renvoyer des réponses correctes selon la spécification http avec la certitude qu'elles seront traitées de manière uniforme entre les agents.

[1] http://lists.w3.org/Archives/Public/public-html-comments/2011Dec/0003.html
[2] http://www.cmhjones.com/browser-http-test-matrix.html

Julien Reschke 2012-01-14 09:48:33 UTC Commentaire 20

Ouvert < https://www.w3.org/html/wg/tracker/issues/195 > et < https://www.w3.org/html/wg/tracker/issues/196 >.

Tom Wardrop2012-03-10 12:09:20 UTC Commentaire 21

Tout d'abord, je suis surpris du rejet apparemment injustifié de cette demande d'amélioration. Ian, il serait utile que vous donniez des précisions sur votre poste actuel.

Je suis personnellement favorable à l'ajout de la prise en charge des méthodes PUT et DELETE à la spécification HTML. En fait, j'aimerais personnellement voir la prise en charge de toutes les méthodes de requête HTTP. HTML est inextricablement lié à HTTP. HTML est l'interface humaine de HTTP. On peut donc automatiquement se demander pourquoi HTML ne prend pas en charge toutes les méthodes pertinentes dans la spécification HTTP. Pourquoi les machines peuvent-elles METTRE et SUPPRIMER des ressources, mais pas les humains ?

Les formulaires HTML jouent le rôle très important de permettre à un utilisateur d'envoyer des données à un serveur HTTP - des données pertinentes pour l'action en cours. L'action effectuée à la suite de la soumission n'est pas déterminée par l'utilisateur, mais par la méthode des formulaires, telle que définie par le concepteur du formulaire. Si vous supposez que le concepteur du formulaire n'avait aucun contrôle sur le serveur HTTP, alors comment le concepteur du formulaire permet-il à l'utilisateur du formulaire de SUPPRIMER ou de METTRE une ressource ? Il devrait être possible de placer un formulaire HTML devant n'importe quel serveur HTTP prenant en charge GET, POST, PUT et DELETE, et que ce formulaire puisse effectuer n'importe laquelle de ces actions sans avoir besoin de code côté serveur qui tombe en dehors du HTTP spécification.

C'est uniquement à cause des langages de programmation côté serveur et des programmeurs qui ont pris des dispositions spéciales, que nous avons réussi à nous passer du support PUT et DELETE en HTML. Si quelqu'un a un serveur HTTP statique/traditionnel qui dépend des méthodes HTTP disponibles pour répondre aux demandes, alors HTML ne permet pas une interaction complète avec les ressources sur ce serveur, et je pense que cela indique un problème fondamental avec la spécification HTML.

Avec la popularité de REST, les architectes d'applications veulent fournir une interface unique pour les machines et les humains. Les demandes ne devraient pas avoir à déterminer qui ou quoi a fait la demande. La méthode HTTP doit indiquer au serveur quelle action doit être effectuée et les en-têtes doivent communiquer ce que le client attend en réponse. La spécification HTML actuelle ne permet pas une telle interface unifiée, du moins pas sans les solutions de contournement employées par les frameworks d'application. En tant qu'interface humaine pour HTTP, HTML a échoué.

Il est contradictoire que, bien que HTML se donne beaucoup de mal pour assurer le balisage sémantique, il n'a jusqu'à présent fait aucun effort pour garantir les requêtes HTTP sémantiques.

Si ce n'est pas une raison suffisante pour inclure au moins les méthodes PUT et DELETE dans la spécification HTML, alors il faut expliquer pourquoi. Des détails d'implémentation mineurs, tels que la façon dont le client doit se comporter dans certaines conditions, ne doivent pas empêcher de déterminer la validité ou la pertinence de cette amélioration suggérée pour la spécification HTML. Bien que je ne sois pas tout à fait sûr pourquoi ou comment PUT et DELETE introduiraient des problèmes d'implémentation client non déjà résolus par la prise en charge de la méthode POST.

Il est évident que ce rapport de bogue n'a pas encore trouvé de résolution, j'ai donc changé le statut en RÉOUVERT.

Julien Reschke 2012-03-10 12:16:35 UTC Commentaire 22

(En réponse au commentaire #21)
> ...
> Il est évident que ce rapport de bogue n'a pas encore trouvé de solution, j'ai donc
> a changé le statut en RÉOUVERT.

Tom, voir commentaire 20 . Ce problème a été escaladé, il est donc correct que l'état ici soit "RÉSOLU". (En cas de doute, voir le document de politique de décision du groupe de travail).

Une remarque générale : je ne pense pas qu'il y ait une forte opposition à cela en général ; la chose délicate est de savoir jusqu'où aller et d'obtenir la bonne interaction avec HTTP et si HTML5 est (étant post Last Call) est la bonne version à laquelle l'appliquer.

Tom Wardrop 2012-03-10 23:59:12 UTC Commentaire 23

J'ai en quelque sorte manqué ces commentaires, mes excuses. La proposition que vous avez faite semble bien pensée. J'ai d'abord pensé que c'était trop complexe, mais la justification était solide.

Pierre Thierry 2013-07-17 13:49:51 UTC Commentaire 24

*** ~~Le bogue 22695~~ a été marqué comme un doublon de ce bogue. ***