RFC 6068 : The 'mailto' URI Scheme

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 5 octobre 2010

Date de publication du RFC : Octobre 2010

https://www.bortzmeyer.org/6068.html

Le plan d'URI `mailto:`, qui permettait d'indiquer une adresse de courrier à contacter, était normalisé dans le RFC 2368[1]. Désormais, il est remplacé par ce RFC 6068, qui introduit notamment la possibilité d'internationalisation des adresses indiquées en paramètre de `mailto:`.

La plupart des plans d'URI, comme `http:` ou `ftp:` permettent de désigner une ressource accessible quelque part sur le Web. `mailto:` est un cas à part dans la mesure où il sert plutôt à déclencher une action, l'envoi d'un message. L'usage le plus connu du plan `mailto:` est dans HTML, quand on écrit quelque chose du genre :

```
<p>N'hésitez pas à <a href="mailto:contact@example.org">signaler</a> tout problème que vous verriez dans ces pages. Merci d'avance.</p>
```

et le lecteur verra un lien sur le mot « signaler ». Sélectionnant ce lien, son navigateur lui proposera alors d'envoyer un courrier à `contact@example.org` (ou bien déléguera cette tâche à un MUA).

La syntaxe formelle des URI `mailto:` figure en section 2. Outre l'adresse de destination (comme dans l'exemple ci-dessus), la grammaire permet de spécifier des champs qui seront pré-remplis comme `Subject:` (la grammaire ne met pas de limite sur les noms de ces champs, donc elle peut servir pour tout nouveau champ inventé). Juste après le nom du plan, `mailto` et le deux-points obligatoire, figure l'adresse du destinataire. Elle est évidemment à la syntaxe normalisée dans le RFC 5322, avec quelques restrictions, notamment l'interdiction des commentaires (dans l'adresse `Jean Durand <jean@durand.example>`, « Jean Durand » est le commentaire). La section 6 fournit de nombreux

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc2368.txt`

exemples, que je reprends ici (parfois un peu adaptés). Par exemple, un URI `mailto:` de base peut être `<mailto:chris@example.com>` (en utilisant la convention de l'annexe C du RFC 3986 d'écrire l'URI entre chevrons). Un URI avec sujet peut être `<mailto:infobot@example.com?subject=current-issue>`.

En théorie, n'importe quel en-tête peut être ainsi spécifié et, par exemple, un logiciel qui archive une liste de diffusion sous forme de page Web (comme Mhonarc `<http://www.mhonarc.org/>`) pourrait mettre un `In-Reply-To:` dans les URI, pour garantir qu'une réponse ne casse pas les fils de discussion. Le RFC cite cet exemple :

```
<li><a href="mailto:list@example.org?In-Reply-To=%3C3469A91.D10AF4C@example.com%3E">
Foo larger than Bar?</a> 2010-10-04 08:28</li>
```

C'est très astucieux mais, en 2010, je ne connais aucun archiveur qui fasse cela...

Si on spécifie deux champs dans l'URI `mailto:`, on les sépare par le & : `<mailto:joe@example.com?cc=bob@e`

Bien des caractères qui sont légaux dans une adresse de courrier peuvent poser des problèmes dans un URI. C'est le cas par exemple de %, de /, de ?, etc. Ceux-ci doivent être encodés en « pour-cent », c'est-à-dire que, par exemple, ? devient %3F (la valeur de l'octet en hexadécimal). Ainsi, un URI avec indication du corps du message et un espace dans ce corps peut être `<mailto:infobot@example.com?body=send%20cu`
Un URI pour écrire à `gorby%kremvax@example.com` devrait s'écrire `<mailto:gorby%25kremvax@example.co`
(Cet exemple du RFC fait référence à une vieille blague Usenet.)

Si les caractères à encoder ne sont pas dans ASCII, c'est leur forme UTF-8 qui est encodée pour-cent, octet par octet. Ainsi, `Stéphane` devient `St%C3%A9phane`. Doit-on appliquer le même traitement aux caractères dans la partie « domaine » (à droite du @) de l'adresse, si le domaine est un IDN ? Oui, on peut, en appliquant l'encodage pour-cent à la forme UTF-8. Si l'adresse est `druon@académie-française.fr`, on peut l'écrire `druon@acad%C3%A9mie-fran%C3%A7aise.fr` dans un URI `mailto:` mais il faudra le transformer en Punycode (`druon@xn--acadmie-franaise-npb1a.fr`) au moment de la composition du message, ce qui garantira que cela fonctionne toujours, même avec des applications anciennes. Un exemple plus sophistiqué est donné en section 6.3 avec le mot japonais *"natt[Caractère Unicode non montré* [2] *]"*, qui s'écrit en Unicode U+7D0D U+8C46. Pour écrire à `user@"natto".example.org`, on aura alors un URI `<mailto:user@%E7%B4%8D%E8%B1%86.example.org?subject=Test&body=NATTO>`.

Même les champs supplémentaires indiqués dans l'URI peuvent avoir besoin d'être encodés pour-cent, car leur nom peut contenir des caractères ASCII mais interdits dans un URI. C'est encore plus vrai pour les valeurs des champs, qui ont des chances de contenir des caractères non-ASCII. Ainsi, demander du café (cf. RFC 2324) par courrier se fera avec un URI `<mailto:user@example.org?subject=caf%C3%A9>`.
`body` pose un cas particulier car ce n'est pas un vrai champ de l'en-tête, c'est un pseudo-champ qui identifie le corps du message. Il peut être en UTF-8 (et donc encodé pour-cent), mais on ne doit pas le stocker en Quoted-Printable ou autres encodages utilisés dans le courrier. De toute façon, ce pseudo-champ est conçu pour de courts messages, pas pour du multimédia compliqué.

Des surencodages peuvent ensuite avoir besoin de s'appliquer, selon le contexte. Ainsi, si un URI `mailto:` comprend un & (utilisé comme délimiteur de paramètres), il ne posera pas de problème en

2. Car trop difficile à faire afficher par LaTeX

texte seul mais, en XML, il devra être écrit `&` (regardez le source XML ou HTML de cet article pour des exemples...).

Après ces détails de syntaxe, le mode d'emploi des URI `mailto:` (section 3). Les normes sur les URI spécifient leur syntaxe mais en général ne donnent guère de détails sur leur résolution : un URI est avant tout un identificateur. Pour `mailto:`, l'usage typique est d'envoyer un courrier à l'adresse donnée en paramètre, en incluant les champs supplémentaires donnés dans l'URI.

L'encodage des URI, sujet compliqué, justifie une section entière, la numéro 5. Le RFC 3986 impose un surencodage de certains caractères, que l'on peut trouver dans une adresse, dans un en-tête, ou dans le corps du message. Un exemple typique est l'espace, qui doit être écrit %20. Quant aux fins de ligne, elles **doivent** suivre la règle de l'Internet et être composées de deux caractères, %0D%0A. Tout logiciel qui produit des URI `mailto:` doit donc faire attention à encoder ces caractères « spéciaux ». Par exemple, dans un formulaire HTML, il est courant de représenter l'espace par un +. Mais cela peut créer une confusion avec les vrais +, qui peuvent se retrouver dans un URI `mailto:`, y compris dans l'adresse `<https://www.bortzmeyer.org/arreter-d-interdire-des-adresses-legales.html>` comme dans l'exemple `bill+ietf@example.org` que cite le RFC. Bref, l'espace doit être représenté avec %20 et le + par lui-même ou par %2B.

À noter que ce RFC n'inclut pas la possibilité d'utiliser des adresses de courrier complètement internatonalisées `<https://www.bortzmeyer.org/courrier-entierement-internationalise.html>`, la spécification de celles-ci étant encore marquée « expérimental ». Elle est devenue norme complète en 2012 avec le RFC 6530.

Les changements depuis le RFC 2368 sont rassemblés en section 9 : le principal est évidemment la possibilité d'inclure de l'UTF-8, encodé en pour-cent, possibilité qui ajoute l'internationalisation aux URI `mailto:`. Le nouveau plan `mailto:` est désormais enregistré dans le registre des plans `<https://www.iana.org/assignments/uri-schemes.html>`, tel que documenté en section 8.

Comme l'utilisation normale d'un URI `mailto:` est une **action** (envoyer un message), il faut soigner la sécurité, à laquelle la section 7 est consacrée. L'utilisateur n'est pas forcément conscient de ce qu'il fait (d'autant plus que le texte du lien peut être trompeur, par exemple `<a href="mailto:victim@example.com">Envoye un message à attacker@example.net</a>`). Le RFC insiste donc que le logiciel ne doit **pas** envoyer de message en réaction à un `mailto:` sans l'accord explicite et informé (affichage du contenu complet, et des en-têtes) de l'utilisateur. Autrement, l'utilisateur risque, sans s'en rendre compte, d'envoyer des messages qui lui vaudront des ennuis (par exemple parce qu'ils contiennent des menaces).

D'autre part, les URI `mailto:`, étant structurés, se prêtent bien à l'analyse et donc à la récupération automatique. Comme cette récolte a en général pour but l'envoi de spam, il peut être préférable de ne pas utiliser d'URI `mailto:` (c'est le cas de ce blog pour l'adresse indiquée en bas de chaque page).

Autre problème, la section 4 met en garde contre les en-têtes « dangereux ». Une liste indicative se trouve en section 3, incluant par exemple les en-têtes de routage du courrier comme `Apparently-To:`. `From:` est aussi dangereux (car il permet de faire envoyer un message qui ment sur sa source). Le RFC cite ensuite quelques en-têtes « sûrs » : `Subject:`, `Keywords:` et le pseudo en-tête `body`. Si un URI `mailto:` contient des en-têtes dangereux, le navigateur ne doit pas envoyer le message, ou alors il doit le « censurer » en ne tenant pas compte des dits en-têtes. De toute façon, en pratique, le créateur d'un URI `mailto:` ne peut pas espérer que les en-têtes autres que `Subject:` et `body` soient compris par le navigateur.