

Le Comportement des Navigateurs Internet

par [Mathieu Lemoine](#)

Date de publication : 01/07/2006

Dernière mise à jour : 16/10/2006

Les navigateurs Web, ou Browsers en anglais, font désormais partie des outils de tous les jours dans le paysage informatique. Malheureusement, et bien que ca devrait pourtant leur être indispensable, la plupart des Webmasters ignorent tout ou presque de leurs comportements les plus basiques. Nous allons voir quels sont les mécanismes de bases de leur comportement.

Notes

- Remerciements
- Introduction
- I - La résolution du nom de domaine
 - I-A - Cas général
 - I-B - Points particuliers
- II - La construction de la requête HTTP
 - II-A - Choix de la méthode
 - La Méthode GET
 - La Méthode POST application/x-www-form-urlencoded
 - Les requêtes POST multipart/form-data
 - II-B - Génération des en-têtes
 - Les en-têtes ne dépendant pas de la requête
 - Les en-têtes dépendant de la requête
 - II-C - Cas de la soumission d'un formulaire : récupération des données
 - Sélection des balises
 - L'ordre des données
- III - Envoi de la requête et traitement de la réponse
 - III-A - Envoi de la requête
 - III-B - Traitement de la réponse
 - Les en-têtes Content-type et Content-length
 - L'en-tête Set-Cookie
 - Le corps de la réponse
- IV - Voir aussi

Notes

Remerciements

Merci à [Yogui](#) pour sa relecture.

Introduction

Cet article présente le comportement fondamental des navigateurs Internet d'une manière simple et, normalement, compréhensible pour n'importe qui ayant un minimum de connaissances concernant la Théorie des Réseaux, le protocole HTTP (dans sa version 1.0 ou 1.1), le langage (X)HTML et quelques notions (au moins) au niveau de la gestion des noms de domaines sur l'Internet.

Nous utilisons tous les jours un ou plusieurs navigateurs Web. En ce moment même, pour lire ce tuto, vous en utilisez certainement un (si vous lisez la version en ligne).

Normalement, tout Webmaster devrait avoir au moins une connaissance des actions de bases qui sont effectuées par ces logiciels.

Malheureusement, ce n'est que trop rarement le cas. J'espère que cet article pourra vous éclairer...

I - La résolution du nom de domaine

I-A - Cas général

Afin de récupérer une page Web, il faut se connecter au serveur Web chargé de gérer cette page. Cependant, on ne peut pas se connecter directement à un serveur à partir de son nom de domaine, il faut l'adresse IP du serveur. Ces correspondances entre nom de domaine et adresse IP sont stockées au niveau des serveurs DNS. Les adresses de ces serveurs DNS sont stockées dans le système d'exploitation, le navigateur va donc demander au système d'exploitation l'adresse correspondant au nom de domaine sur lequel est hébergée la page. Ceci s'appelle la résolution du nom de domaine. Quand l'IP est introuvable (nom de domaine pas enregistré, serveur DNS qui ne répond pas, pas de serveur DNS connu du système d'exploitation, etc.), alors la page Web n'est pas accessible et le navigateur indique un message d'erreur caractéristique.



Message d'erreur DNS de Mozilla Firefox

I-B - Points particuliers

- Dans le cadre de la navigation interne à un site Web, il est probable que le navigateur connaisse déjà l'adresse IP associée à un nom de domaine, dans ce cas la demande n'est pas effectuée une nouvelle fois.
- Le protocole DNS (qui est utilisé pour communiquer avec les serveurs DNS) est complexe et met en jeu de nombreux serveurs. Il est parfois nécessaire de contacter de nombreux serveurs DNS avant de pouvoir résoudre complètement un nom de domaine.
- Si l'organisme gérant le site Web est très important, il est possible qu'un système de répartition de charge soit mis en place. Dans ce cas, il est possible que plusieurs IPs soient associées à un même domaine. En général, c'est alors le système d'exploitation qui choisit l'IP à indiquer au navigateur.
- Il est possible de simuler manuellement cette opération manuellement grâce au programme nslookup, qui est disponible sur de nombreuses plateformes et permet de spécifier le serveur DNS à utiliser. Si vous n'en spécifiez pas, c'est le serveur DNS enregistré dans le système d'exploitation qui sera utilisé.

syntaxe de nslookup

```
nslookup {nom.de.domaine} [ {IP.du.serveur.DNS} ]
```

II - La construction de la requête HTTP

II-A - Choix de la méthode

Maintenant que l'on connaît l'adresse IP du serveur, on peut s'y connecter afin d'envoyer la requête HTTP adéquate. Cependant, avant d'envoyer la requête HTTP, il faut la construire.

- Les requêtes GET
- Les requêtes POST application/x-www-form-urlencoded
- Les requêtes POST multipart/form-data

Dans le cadre de certaines optimisations liées au cache, il est possible que le navigateur puisse exceptionnellement utiliser des requêtes HEAD. Toutefois, cette alternative est laissée à la discrétion du navigateur, sous réserve que le serveur accepte la méthode HEAD (et donc la version 1.1 du protocole HTTP). La construction de la requête se fait en plusieurs temps.

La Méthode GET

Les requêtes de type GET sont les plus fréquemment employées.

- Une URL est entrée directement dans la barre d'adresse
- On clique sur un lien
- On charge un document annexe à la page (frame, iframe, feuille de style, image, icône, fichier de script, etc.)
- On soumet un formulaire utilisant la méthode GET
- La réponse HTTP précédente a retourné un code status impliquant une redirection simple

La Méthode POST application/x-www-form-urlencoded

Les requêtes de type POST application/x-www-form-urlencoded sont utilisées à la soumission d'un formulaire dont la propriété enctype a la valeur application/x-www-form-urlencoded (c'est la valeur par défaut).

Le format d'encodage des données est le même que pour les paramètres GET.

Les requêtes POST multipart/form-data

Ces requêtes sont utilisées à la soumission d'un formulaire dont la propriété enctype est à multipart/form-data. Le format est plus complexe que pour les POST application/x-www-form-urlencoded, car elles doivent permettre d'envoyer des fichiers en plus des données.

Il y a tout d'abord un élément essentiel, nommé le boundary. C'est une chaîne de caractères qui permet de servir de séparateur, son format et sa valeur sont laissés à la discrétion du navigateur. Il est spécifié à l'aide d'une en-tête dans la requête HTTP. Ensuite, chaque donnée de formulaire est indiquée de cette façon :

```
Content-Disposition: form-data; name=" {NOM} "[CRLF]
[CRLF]
{VALEUR}
```

Tandis que les fichiers sont indiqués selon un format presque similaire :

```
Content-Disposition: form-data; name="{NOM}"; filename="{NOM DU FICHER}"[CRLF]
Content-Type: {TYPE MIME}[CRLF]
[CRLF]
{CONTENT}
```

Le *[CRLF]* correspond à un retour à la ligne "\r\n". Le NOM et le NOM DU FICHER ne doivent pas contenir de guillemet (") ni de retour à la ligne (\r ou \n).

Les données sont séparées par le boundary, qui est seul sur une ligne. De plus, l'ensemble des données est lui-même indiqué entre deux lignes ne contenant que le boundary.

II-B - Génération des en-têtes

Les en-têtes de la requête sont générées de plusieurs façons selon les en-têtes et la configuration.

Les en-têtes ne dépendant pas de la requête

Ces en-têtes sont envoyées par le navigateur à chaque requête, et ne dépendent pas de la dite requête.

- **User-Agent**

Cette en-tête indique le nom et la version du navigateur. Cela peut servir pour faire par exemple des statistiques sur les visiteurs, mais également à générer des pages optimisées pour le navigateur. Certains navigateurs (Opera par exemple) permettent de modifier leur signature afin de se "faire passer pour un autre navigateur".

- **Accept, Accept-language, Accept-charset et Accept-encoding** Ces en-têtes permettent d'indiquer les types MIME, langues, jeu de caractères et encodages supportés par le navigateur, ils peuvent également être classés par préférence.

Les en-têtes dépendant de la requête

- **Host**

Cette en-tête permet d'indiquer le nom de domaine sur lequel s'effectue la requête, puisqu'en HTTP/1.1, un serveur peut héberger plusieurs noms de domaines.

- **Referer**

Cette en-tête permet d'indiquer la page chargée précédemment, celle qui a conduit à effectuer la requête courante. Certains navigateurs permettent de désactiver cette en-tête.

- **Cookie**

Si le navigateur est configuré pour les supporter, il va lister ici les cookies disponibles pour la page (un exemplaire de l'en-tête par cookie).

II-C - Cas de la soumission d'un formulaire : récupération des données

Quand un formulaire est soumis, les données associées sont récupérées puis envoyées au serveur dans le format approprié. Voici comment sont sélectionnées et regroupées les données.

Sélection des balises

- input
- select
- textarea

Celles dont la propriété *disabled* (désactivé) est spécifiée sont ignorées. Chaque balise est ensuite traitée selon le cas. Le nom est à chaque fois l'attribut name : s'il est vide, alors une chaîne vide est envoyée en tant que nom.

- **Les input button**

Ils ne sont pas destinés à être envoyés au serveur, mais uniquement à déclencher une action côté client.

Ils sont ignorés.

- **Les input submit et image**

Si la soumission a été déclenchée par le clic sur un élément de ce type, il est alors communiqué au serveur, sinon il est ignoré. Si l'input ne possède pas d'attribut value, alors la valeur est laissée à l'appréciation du navigateur, sinon c'est elle qui est spécifiée. Pour les input image, les coordonnées du clique (par rapport au coin inférieur gauche) sont communiquées grâce à deux données supplémentaires dont les noms sont la valeur du name, suivi d'un underscore (_), suivi d'un x minuscule (coordonnée horizontale) ou d'un y minuscule (coordonnée verticale).

- **Les input checkbox et radio**

Seuls ceux qui sont cochés sont envoyés, s'il n'y a pas de valeur spécifiée (attribut value), alors le défaut est laissé à l'appréciation du navigateur (en général, c'est "on" qui est utilisé).

- **Les inputs text, hidden, password et les textarea**

Ces quatre types d'éléments sont traités de la même façon, la valeur est simplement celle entrée par l'utilisateur ou spécifiée dans l'attribut value (pour les inputs) ou entre les balises du textarea.

- **Les select**

Pour les select (multiple ou non), la (ou les) valeur(s) est (sont) l'attribut value de l' (des) option sélectionnée(s), ou à défaut, le contenu des balises.

Dans le cas d'un select multiple, chaque valeur est envoyée avec le même nom. Si aucune valeur n'est

sélectionnée, alors il est ignoré.

- **Les input file**

Les input file sont traités différemment selon la méthode de soumission. S'il s'agit d'un POST multipart/form-data, alors les données sont extraites comme indiqué dans la description de ce type de requêtes.

Sinon, la valeur est simplement le nom du fichier (sans le chemin d'accès).

L'ordre des données

L'ordre est propre au navigateur. Cependant, aucune donnée ne doit être supprimée, même en cas de collision de nom, ou de nom absent.

III - Envoi de la requête et traitement de la réponse

III-A - Envoi de la requête

Une fois la requête construite, le navigateur se connecte au serveur et envoie la requête.

III-B - Traitement de la réponse

Le premier élément de la réponse qui est reçue est le code status et les en-têtes de la réponse.

Dans tous les cas, le navigateur doit rester conforme au protocole HTTP et signaler tout code d'erreur (4xx et 5xx).

Pour les en-têtes, certaines peuvent entraîner quelques modifications du comportement des navigateurs.

Les en-têtes Content-type et Content-length

Ces en-têtes indiquent le type de ressource renvoyée. Dans le cas d'un type MIME non reconnu ou absent, c'est le navigateur qui choisit la valeur par défaut... En général, c'est plain/text qui est utilisé.

Quand Content-length est spécifié, alors la taille du corps de la réponse doit correspondre. Tout ce qui est au-delà doit être ignoré par le navigateur.

L'en-tête Set-Cookie

Cette en-tête permet au serveur de spécifier des cookies à stocker. Si sa configuration l'autorise, alors le navigateur doit mettre ses cookies à jour à la réception de cette en-tête.

Le corps de la réponse

Le navigateur récupère le corps de la réponse, si nécessaire, le parse, et le charge comme il convient (selon que ce soit une page web, une image, une feuille de style, etc.).

Dans le cas d'une page HTML (par exemple), chaque document annexe donnera lieu à un nouvel échange HTTP avec le serveur.

Certains types de données, tels que *plain/text* ou *application/octet-stream*, permettent une lecture continue des données. Ainsi, on peut se servir du protocole HTTP pour visualiser en temps réel (ou presque) la trace d'une application (par exemple). Dans ces cas, le navigateur **peut** éventuellement afficher le corps de la réponse au fur et à mesure de sa réception plutôt que d'attendre la fin de la réception...

IV - Voir aussi

[Le Protocole HTTP](#) : Voici un article qui vous permettra d'en apprendre plus sur le protocole HTTP ou, en tous cas, d'acquérir les connaissances nécessaires pour pouvoir aborder cet article.