Tutoriel d'URL Rewriting (réécriture de liens)

Exemple avec phpBB 2

par Guillaume Rossolini (Tutoriels Web / SEO / PHP) (Blog)

Date de publication : 7 mai 2006

Dernière mise à jour : 5 juillet 2006

Pour diverses raisons (optimisation de site, faciliter la mémorisation des liens, cloaking, *etc.*), il peut être souhaitable de modifier la forme que prennent les liens d'un site Internet, sans pour autant changer toute la structure des pages physiques.

C'est ce que permet la réécriture dynamique de liens, alias URL Rewriting. J'ai mis en place cette technique sur le **Forum Cinéma**.

I - Notes

I-1 - Remerciements

L'équipe Web pour m'avoir relu et pour avoir proposé des améliorations.

I-2 - Avertissements

- [Avertissement] Cet article est inutile pour les sites non dynamiques.
- [Avertissement] Une mauvaise réécriture des adresses rendra tout ou partie de votre site inaccessible.
- [Avertissement] Deux conditions techniques doivent être remplies pour que ce Mod fonctionne correctement avec votre site : votre serveur (hébergeur) doit autoriser l'utilisation du fichier *.htaccess* ainsi que la réécriture des adresses (directive *RewriteEngine*).

I-3 - Rappels

- [Rappel] J'évoque *PHP* mais d'autres langages de script sont possibles.
- [Rappel] Similairement, d'autres serveurs Web qu'*Apache* autorisent ce genre de procédé.
- [Rappel] De même, mon exemple traite de *phpBB* mais il est facile de l'adapter à toute autre forme de site.

I-4 - Bonus

- [Bonus] En écrivant le Mod, j'ai noté un léger bug dans la gestion du surlignage par *phpBB*... J'imagine que vous aviez remarqué cette fonctionnalité de *phpBB* (le surlignage) : lorsque l'on fait une recherche, *phpBB* surligne les mots cherchés dans la page lue. Cependant, si l'on accède directement à une page précise du sujet concerné, le paramètre n'est pas transmis et le surlignage n'est pas activé. J'ai réglé ce petit désagrément.
- [Bonus] Tous les liens des messages (du forum) seront réécrits dynamiquement, de la même manière que les liens qui sont en-dehors des messages.
- [Bonus] J'en ai profité pour ajouter la propriété "*title*" dans les liens qui n'en possédaient pas, ce qui affiche un titre lorsqu'on passe la souris.

I-5 - Configuration

- [Config] Pour changer le texte affiché dans les liens, modifiez à la fois le fichier de langue (*lang_urlrewrite.php*) et le fichier *.htaccess*.
- [Config] Ajouter un fichier de langue dans chacun de vos répertoires de langues vous permettra de réécrire les adresses en fonction du profil du visiteur. Si vous le faites, vous devrez prendre garde à ce que le fichier *.htaccess* (unique, pour sa part) reflète toujours toutes les possibilités (attention aux conflits !).

II - Théorie

II-1 - Introduction

La problématique : les liens courants sont peu pratiques et très peu optimisés

Il y a plusieurs raisons à cela :

- Nous, lecteurs humains, avons du mal à lire ces adresses.
- Elles présentent peu d'informations qui nous intéressent : par exemple, il manque presque systématiquement les mots clefs qui pourraient indiquer le sujet traité par la page liée.

- Les paramètres peuvent être combinés à volonté et comporter autant de valeurs que l'on peut imaginer : les moteurs de recherche n'aiment pas trop s'amuser à enregistrer toutes les possibilités. Les discussions vont bon train quant à la limite (car il y a une limite) du nombre de paramètres que Google référence : trois, six, *etc*.
- De même, il est possible d'intervertir l'ordre des paramètres pour donner une liste presque infinie d'adresses qui sont par ailleurs totalement identiques (si l'on demande plusieurs pages au serveur et que l'on change uniquement l'ordre des paramètres -pas leur valeur-, toutes les pages envoyées en réponse seront 100% identiques).

Le gros de la question est lié au référencement et aux moteurs de recherche : ils ont tendance à laisser de côté les liens qui comportent trop de paramètres (plus de deux, par exemple). Par conséquent, ils ne référencent pas la totalité du site. De même, le manque de mots clefs est un manque à gagner considérable...

La forme des liens pose donc un problème.
La solution est simple : il faut modifier cette apparence. Cependant, techniquement, nous sommes contraints à l'utiliser.

Exemple de lien dynamique
```
http://g-rossolini.developpez.com/tutoriels/seo/url-rewriting/?page=2
```

Une solution trop simple : ajouter un paramètre inutile

Le principe est d'ajouter un paramètre, par exemple nommé "keywords", qui permet d'ajouter une chaîne de caractères servant de mots clefs. Le script ignore ce paramètre puisqu'il est inutile, mais le fait qu'il apparaisse permet aux bots d'analyser les mots clefs de la nouvelle URI.
Tant que l'on a un minimum de contrôle sur son serveur Web, il est préférable de porter son choix sur la solution suivante. Le paramètre inutile ne doit être qu'un dernier recours et utilisé avec précaution, dans la mesure où les moteurs de recherche pourraient l'interpréter comme une tentative de les tromper.

Exemple de paramètre inutile
```
http://g-rossolini.developpez.com/tutoriels/seo/url-rewriting/?page=2&keywords=theorie-de-l-url-rewriting
```

Une meilleure solution : complètement réécrire les liens contenus dans la page Web

C'est ici qu'entre en jeu **la réécriture de liens** (alias **URL Rewriting**), un procédé dynamique servant de compromis. En effet, cette technique modifie l'apparence des liens pour l'utilisateur (vous et moi) sans pour autant modifier la structure du site du côté du serveur Web.

Je sens qu'un exemple rendra l'explication bien plus claire :

- Imaginons qu'un sujet de notre forum soit accessible par l'URL sujet-3678,tuto-l-url-rewriting-reecriture-de-liens.htm
- En réalité, le lien que *phpBB* reçoit et analyse est : viewtopic.php?t=3678

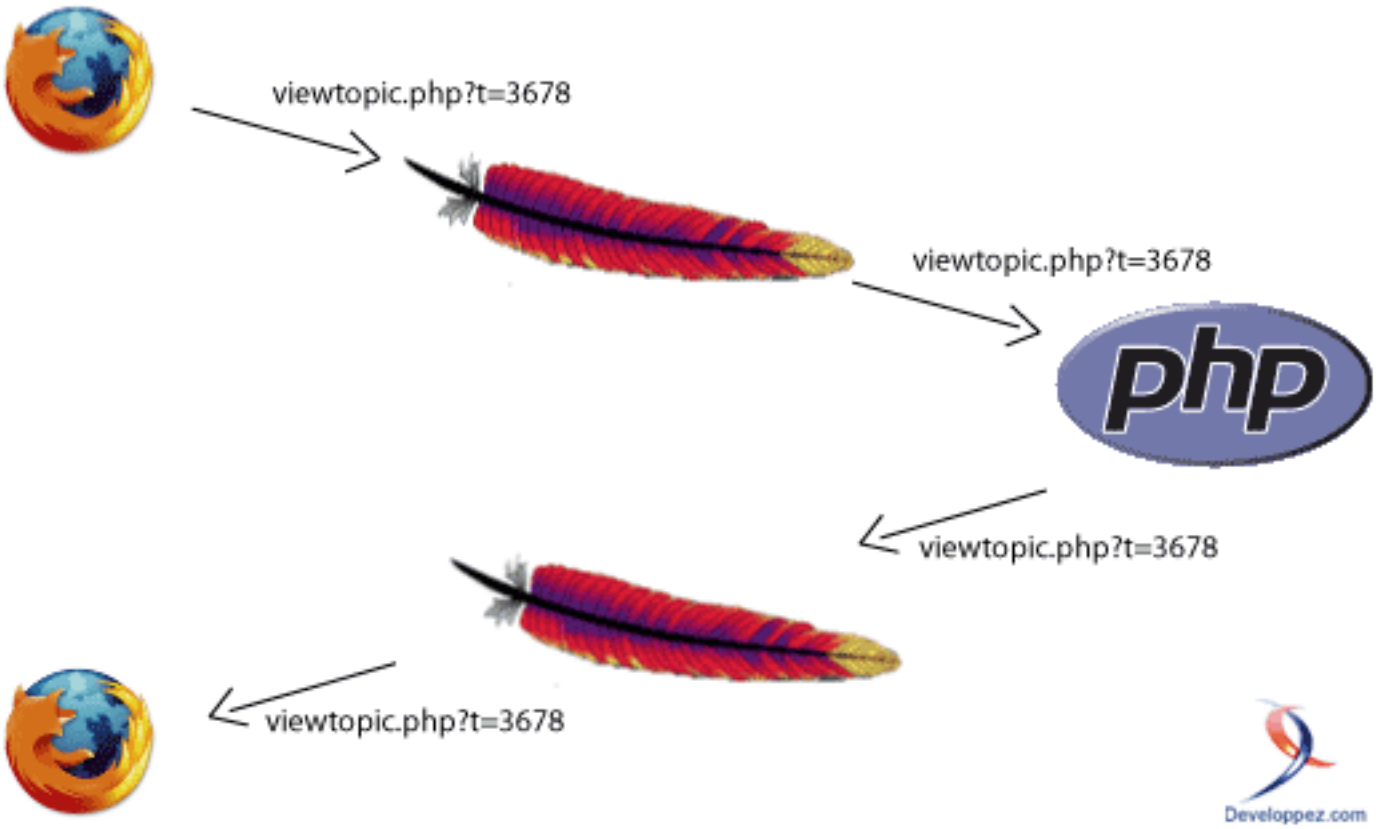
Plus techniquement, il s'agit d'intervenir avant d'envoyer la page demandée par le client, c'est-à-dire un peu avant la fin de l'exécution du fichier de *pied de page*. Nous envoyons au navigateur les *URLs* (les liens) sous une certaine forme modifiée puis, lorsque le client clique sur ces liens, nous intervenons à nouveau avant *PHP*. Ainsi, nous traduisons l'adresse que le client a demandée dans un format que le serveur Web et *PHP* pourront interpréter.

Exemple de lien réécrit
```
http://g-rossolini.developpez.com/tutoriels/seo/url-rewriting/page-2-theorie-de-l-url-rewriting.html
```

Un exemple de transaction : demande / réception d'une page Web

Voici comment se passe une demande de page classique :

1 Le navigateur demande la page viewtopic.php?t=3678
2 Le serveur Web ne sait pas traiter la page, il l'envoie donc à *PHP*
3 *PHP* traite le script et renvoie le résultat au serveur Web
4 Le serveur Web transmet ce résultat au navigateur

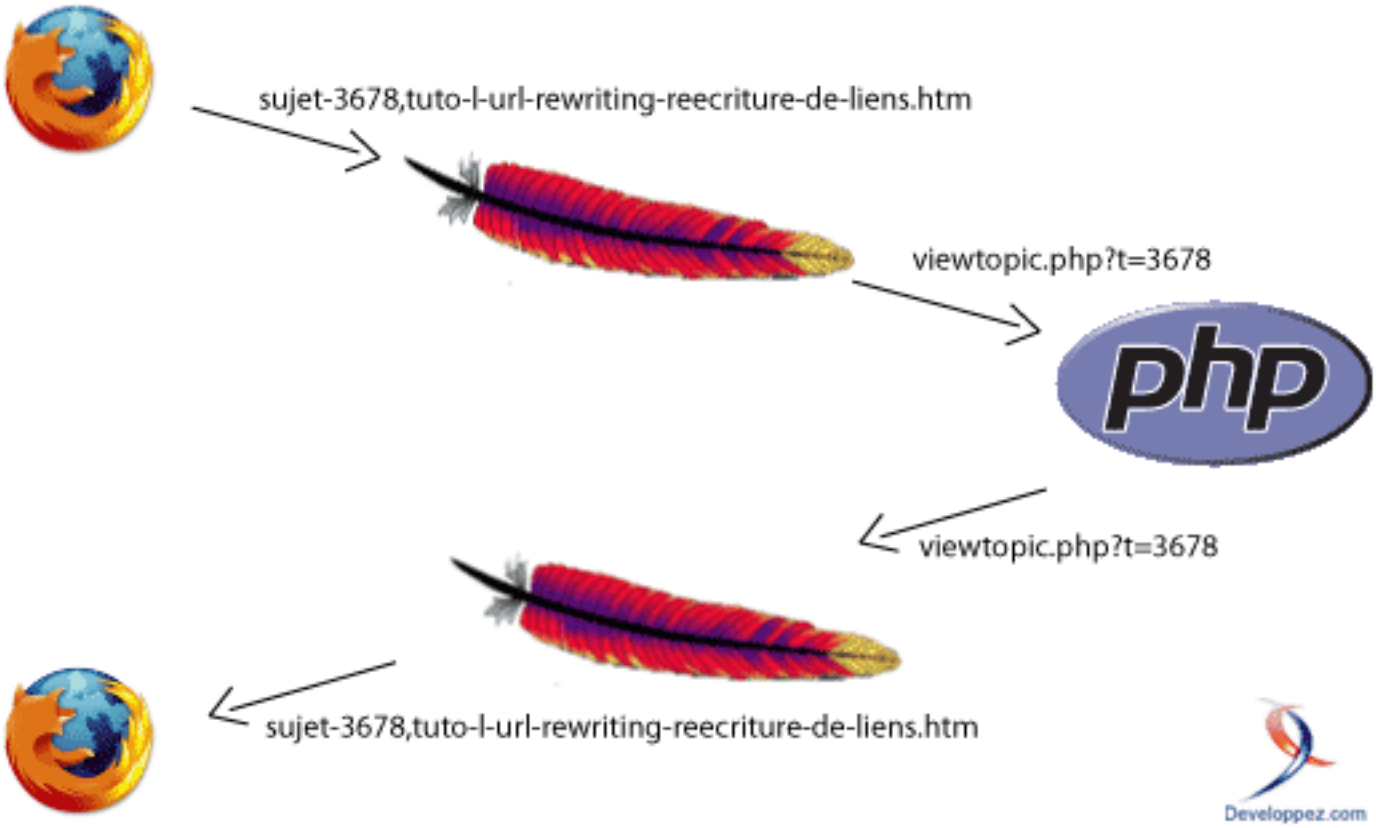


Exemple de transaction classique : demande / réception d'une page Web

Il est clair ici que le serveur Web se contente de transmettre les requêtes de pages à *PHP* et les réponses au navigateur. Quel fainéant...

Maintenant, voici ce que permet l'URL Rewriting :

1 Le navigateur demande la page sujet-3678,tuto-l-url-rewriting-reecriture-de-liens.htm
2 Le serveur Web se rend compte qu'**il doit effectuer une opération** : il remplace l'adresse demandée par une autre afin que *PHP* puisse s'en occuper correctement ==> viewtopic.php?t=3678
3 *PHP* traite le script et renvoie le résultat au serveur Web
4 Le serveur Web transmet ce résultat au navigateur



Exemple de transaction d'URL Rewriting : demande / réception d'une page Web

L'*URL Rewriting* intercale un niveau (d'abstraction) avant *PHP*.
Il est clair que le serveur Web intervient davantage dans la transaction, il ne se contente plus de simplement servir de courtier.
Concrètement, l'*URL Rewriting* permet d'avoir des liens en français quelle que soit la technologie du site que nous utilisons (*DotClear*, *phpBB*, fait maison, *etc*.) et sans toucher à sa structure.
Lorsque nous cliquons sur un lien, un dispositif que nous avons mis en place du côté du serveur modifie le lien sujet-3678,tuto-l-url-rewriting-reecriture-de-liens.htm (qui fait référence à un fichier fictif) en viewtopic.php?t=3678 (qui fait référence à un fichier bien réel) avant que *PHP* commence à traiter la demande.

En résumé

Je viens de mettre en évidence les deux étapes nécessaires pour mettre en place la réécriture de liens : d'une part, modifier les pages en sortie du serveur Web vers le client (elles ne doivent plus être de la forme viewtopic.php?t=3678 mais sujet-3678,tuto-l-url-rewriting-reecriture-de-liens.htm) et, d'autre part, modifier les demandes du client au serveur Web (c'est la procédure inverse).
Dès lors, une question fondamentale surgit : quelle forme doivent avoir mes nouveaux liens ?
Si nous n'avons pas la réponse, nous n'irons pas bien loin... Il est temps de se creuser la tête pour trouver les solutions les plus adéquates !

II-2 - Le brainstorming

Je vais décrire la méthodologie que nous avons suivie, sur le *Forum Cinéma*, pour définir la forme de nos nouvelles *URLs*.
Quand j'ai commencé ce tutoriel, je ne pensais pas m'étendre sur le *brainstorming* car je me disais que ce serait pénible à lire... En fin de compte, ce sera pénible à lire mais je vais le faire néanmoins : bien choisir sa réécriture est exactement ce qui donne toute son efficacité à la technique entière !
Attention, ça va être long... Je ne vous en voudrais pas de sauter cette description !

Il faut connaître quelques petites règles d'optimisation (SEO) avant de commencer

- Commençons par les séparateurs de mots. En effet, il est fondamental de séparer les mots (dans l'*URL*) afin que les moteurs de recherche puissent les confronter à leur dictionnaire et/ou aux termes de la recherche. De nombreux séparateurs sont valables, peu sont recommandés. Nous utiliserons le trait d'union "-" de préférence au slash "/" (peu pratique à gérer, techniquement parlant) et au caractère souligné "_" (déconseillé et peu lisible).
- Vous noterez que ce tutoriel est hébergé sur "*g-rossolini.developpez.com*" : j'ai séparé mon prénom (*Guillaume*) de mon nom de famille (*Rossolini*), de manière à ce que les moteurs de recherche puissent faire correspondre le nom du domaine à une recherche sur mon nom de famille. Sans le trait d'union, ce nom de domaine serait bien mal optimisé car le mot obtenu aurait peu de rapport avec moi. C'est l'une des règles que nous appliquerons.
- Il sera parfois nécessaire d'utiliser un deuxième séparateur : ce sera la virgule ",".
- Il est préférable d'utiliser uniquement les lettres de l'alphabet anglais (aucun accent) et les chiffres arabes. Cela facilite la lecture ainsi que le travail du moteur de recherche, tout en évitant divers problèmes techniques.
- Il faut absolument éviter d'obtenir plusieurs adresses différentes qui redirigent vers la même page (contenus identiques) : cela s'appelle *duplicate content* et cela pénalise largement le référencement de nos pages.
- Il ne faut pas surcharger l'URL en mettant trop de mots clefs.

Rappelons que ce tutoriel a été rédigé en écrivant du code spécifique à *phpBB2*. Je vais donc illustrer mes propos à l'aide de sa structure... Bien évidemment, la démarche reste similaire quel que soit le site dont on souhaite réécrire les *URLs*.
Il faut répertorier tous les fichiers présents sur le serveur. Ou plutôt : répertorier tous les scripts pour lesquels nous souhaitons appliquer la réécriture de liens. Il s'agit de tous les points d'entrée du site. Pour chacun d'eux, il nous faut déterminer quels paramètres de type *GET* ils peuvent accepter et selon quelles combinaisons. Il faut être particulièrement minutieux, sans quoi il restera des liens non réécrits par la suite !
Puisque l'ordre des paramètres dans l'*URL* peut changer, j'ai trouvé plus simple de déterminer d'abord le nombre de *paramètres utiles (non vides)* pour pouvoir ensuite déterminer la liste des combinaisons.

Voici les templates d'URLs que nous utilisons :

- faq.php : 0 ou 1 paramètre
  Pas de surprise, c'est une mise en jambe facile. Nous aurons évidemment deux formes de lien pour ce script : avec ou sans le paramètre.
  De plus, il se trouve que le seul paramètre est "*mode*" et qu'il n'a qu'une seule valeur possible, ce qui simplifie encore les choses.
  Voici ce que nous avons adopté : *foire-aux-questions*, *bbcode*
- groupcp.php : 0 ou 1 paramètre
  Celui-ci est facile également. Le seul paramètre est "*g*", à savoir l'identifiant numérique du groupe souhaité. S'il est présent, il suffit de consulter la base de données et d'y récupérer le nom de ce groupe afin de l'ajouter à notre *URL*.
  Voici ce que nous avons adopté : *groupes-d-utilisateurs*, *groupe-d-utilisateurs-%d-%s*
- index.php : 0 ou 1 paramètre
  Finalement, c'est assez pratique : ce script accepte au maximum un paramètre à la fois, mais ce n'est pas toujours le même. Il peut s'agir de "*c*" (l'identifiant numérique d'une catégorie de forums) ou bien de '*mark*' (pour marquer tous les forums comme lus)
  Voici ce que nous avons adopté : *index*, *categorie-%d-%s*, *marquer-tous-les-forums-comme-lus*
- login.php : 0 ou 1 paramètre
  Le seul paramètre utile est "*logout*" avec la valeur "*true*".
  Il y a une particularité : il faut ajouter l'identifiant de session lors de la construction du lien de déconnexion, sans quoi *phpBB* n'effacera ni les cookies ni la session.
  Voici ce que nous avons adopté : *se-deconnecter-%s*, *se-connecter*.
- memberlist.php : 0 ou 3 paramètres
  Soit il n'y a pas de paramètre, soit il y a à la fois "*mode*", "*start*" et "*order*".
  Nous avons pris le parti de ne pas gérer *ASC* et *DESC* de la même manière.
  Voici ce que nous avons adopté : *liste-des-membres*, *liste-des-membres-par-%s-a-partir-de-%d*, *liste-des-membres-par-%s-a-partir-de-%d-dans-l-ordre-descendant*
- posting.php : 1 ou 2 paramètres

Voilà qui commence à devenir intéressant ^^
Dans le cas d'un paramètre unique, il s'agit de "*mode*" avec la valeur "*smilies*"
Dans le cas de deux paramètres, il s'agit de "*mode*" ayant plusieurs valeurs possibles et combiné avec "*f*" (identifiant numérique d'un forum), "*t*" (identifiant numérique d'un sujet) ou "*p*" (identifiant numérique d'un message).
Gestion d'erreurs : il arrive souvent qu'un message ne possède pas de titre. Dans de tels cas, nous avons opté pour utiliser le titre du sujet en remplacement.
Voici ce que nous avons adopté : *nouveau-sujet-dans-le-forum-%d-%s*, *repondre-au-sujet-%d-%s*, *citer-le-message-%d-%s*, *editer-le-message-%d-%s*

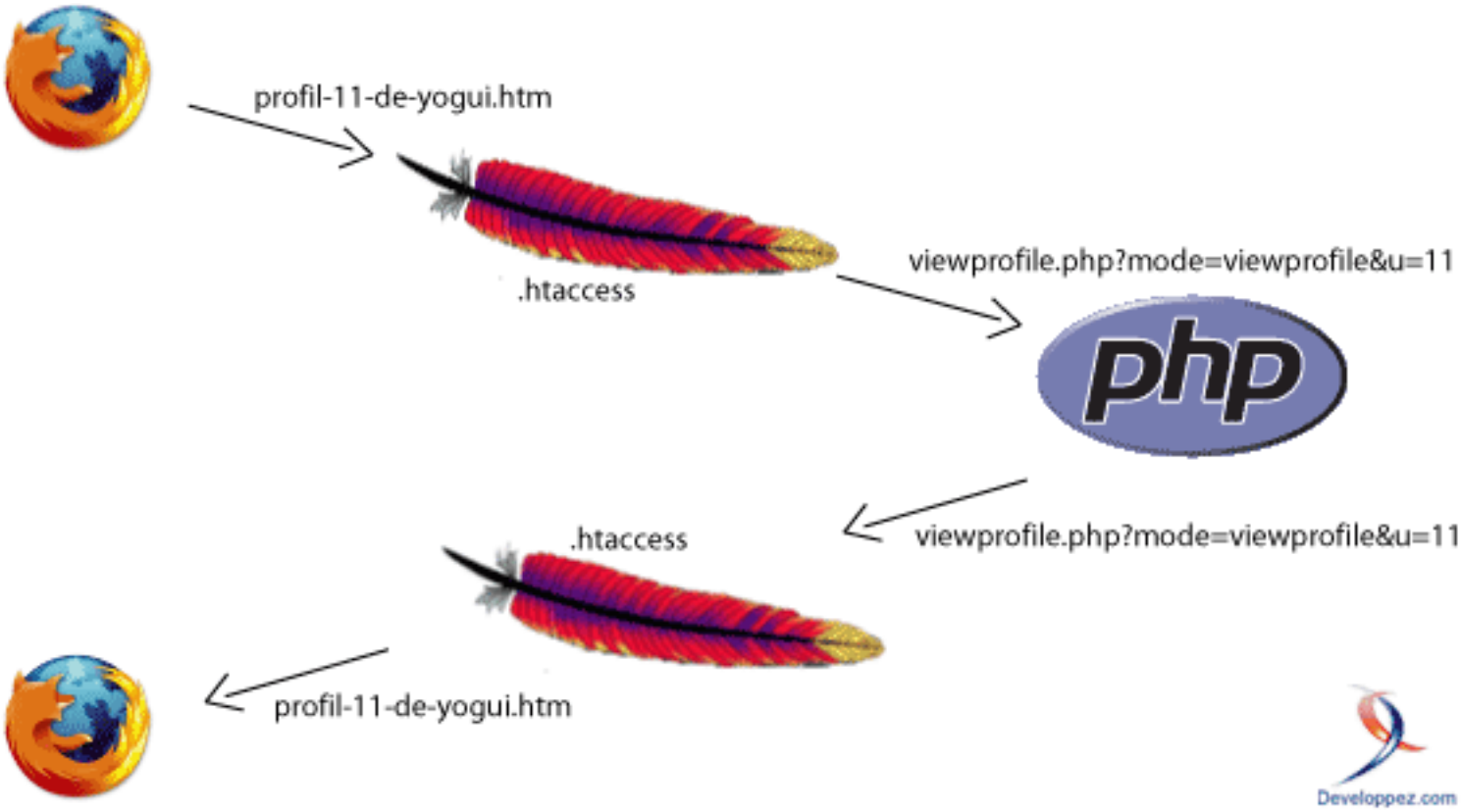
- privmsg.php : de 1 à 3 paramètres
  Un seul paramètre : "*mode*" avec la valeur "*post*" ou bien '*folder*' avec différentes valeurs possibles.
  Voici ce que nous avons adopté : *envoyer-un-message-prive*, *messages-prives-recus*, *messages-prives-en-partance*, *messages-prives-envoyes*, *messages-prives-sauvegardes*.
  Deux paramètres : "*mode*" avec la valeur "*post*" + "*u*" avec l'identifiant numérique du destinataire, ou bien "*mode*" avec diverses valeurs possibles et "*p*" avec l'identifiant numérique du message concerné.
  Voici ce que nous avons adopté : *envoyer-un-message-prive-a-%d-%s*, *repondre-au-message-prive-%d-%s*, *citer-le-message-prive-%d-%s*, *editer-le-message-prive-%d-%s*
  Trois paramètres : "*folder*" avec diverses valeurs possibles + une combinaison de "*start*", "*p*" et "*mode*".
  Je vais abréger le calvaire car vous avez saisi l'idée générale. De plus, tout ce que vous pourriez désirer savoir à se sujet se trouve dans les archives à télécharger.
- viewtopic.php : de 1 à 3 paramètres
  Je voulais terminer par le plus fun de tous... Sans rentrer dans des détails excessifs, je trouve intéressant de se pencher sur ce script car c'est probablement celui qui pose le plus de problèmes.
  Il peut y avoir une combinaison de divers paramètres : "*p*", "*t*", "*watch*", "*unwatch*", "*postorder*", "*view*" et "*highlight*". De plus, il est possible que deux combinaisons différentes mènent à des pages en *duplicate content*. N'ayons crainte, utilisons l'*URL Rewriting* pour y remédier ! Dans le même ordre d'idée, *phpBB* a tendance à afficher des paramètres vides (pagination, ordre d'affichage des messages, *etc.*) et je ne sais pas si les moteurs de recherche les prennent en compte : si c'est le cas, cela génère encore des pages en *duplicate content*. Heureusement que nous filtrons les paramètres au départ du script d'*URL Rewriting*...
  Je me suis rendu compte que les paramètres "*p*" (identifiant numérique d'un message) et "*t*" (identifiant numérique d'un sujet) affichent parfois la même page. Ce n'est qu'une histoire de pagination. Ainsi, j'ai préféré utiliser systématiquement l'identifiant et le titre du sujet (plutôt que ceux du message) afin d'éviter d'avoir des pages en *duplicate content*. L'ancre (le dièse "#" à la fin de l'*URL*) est disponible quel que soit le cas, cela ne pose donc aucun souci.
  Afin de me simplifier la vie, j'ai utilisé la virgule pour séparer le titre (information facultative) des autres paramètres (absolument nécessaires). Il est également possible de se creuser la tête et de trouver d'autres manières de faire (surtout si vous souhaitez conserver le trait d'union) mais, là, j'ai eu la flemme.

Pour faire simple, j'ai utilisé la syntaxe de la fonction *sprintf()* : *%d* représente un nombre (généralement un identifiant numérique) et *%s* représente une chaîne (généralement un titre).

II-3 - Le processus de réécriture

Première partie : traduire les requêtes du client

Il s'agit ici de trouver un moyen d'intercepter les requêtes du client avant qu'elles parviennent à *PHP*. Ce moyen s'appelle "*.htaccess*" si l'on a un serveur *Apache* ou bien "*HTTPHandlerFactory*" si l'on utilise le moteur *ASP .NET*. Je me concentrerai ici sur la solution d'*Apache*.
Il s'agit d'un fichier nommé "*.htaccess*", un simple fichier texte qui va donner des ordres à *Apache*. Parmi ces ordres, plus communément appelés "directives", figureront les changements à effectuer.



Rappel du schéma vu plus tôt : une transaction d'URL Rewriting

Voici ce que doit contenir le fichier .htaccess pour nos besoins :

Fichier : .htaccess
```
RewriteEngine on
RewriteRule ^profil-([0-9]+).* /phpBB2/profile.php?mode=viewprofile&u=$1 [L]
RewriteRule ^editer-profil.* /phpBB2/profile.php?mode=editprofile [L]
RewriteRule ^sujet-([0-9]+).* /phpBB2/viewtopic.php?t=$1 [L]
etc.
```

L'option **L** en fin de ligne indique à *Apache* qu'il peut s'arrêter de comparer dès qu'il trouve le résultat. Pour plus de détails, consulter le fichier complet inclus dans l'archive à télécharger.
La directive *RewriteEngine* est fondamentale car elle active le moteur de réécriture. Les directives suivantes, *RewriteRule*, sont généralement très nombreuses.
Le principe des directives *RewriteRule* est d'utiliser les *expressions régulières*. Je n'ai pas l'intention de rentrer dans les détails sur les *regexp* ici car le chapitre est trop long et car ce n'est pas l'objet de mon article.

Pour nos besoins, il est suffisant de savoir que :

- **[0-9]+** fait référence à un ou plusieurs chiffres à la suite
- **[a-z]+** fait référence à une ou plusieurs lettres à la suite
- **[a-z0-9]+** fait référence à un ou plusieurs caractères (lettres et chiffres confondus)
- **.*** fait référence à n'importe quelle suite de caractères (quantité nulle jusqu'à l'infini)
- Les "**lettres**" font référence à l'alphabet anglais (aucun accent)
- Les **parenthèses** permettent de "capturer" ce qui se trouve à l'intérieur afin de le rappeler dans la suite de la directive : le contenu du premier couple de parenthèses sera appelé par **$1**, le second par **$2** *etc*.
- Lorsque plusieurs règles gèrent des chaînes similaires, il faut toujours mettre de la règle la plus précise à la plus générale. Exemple :

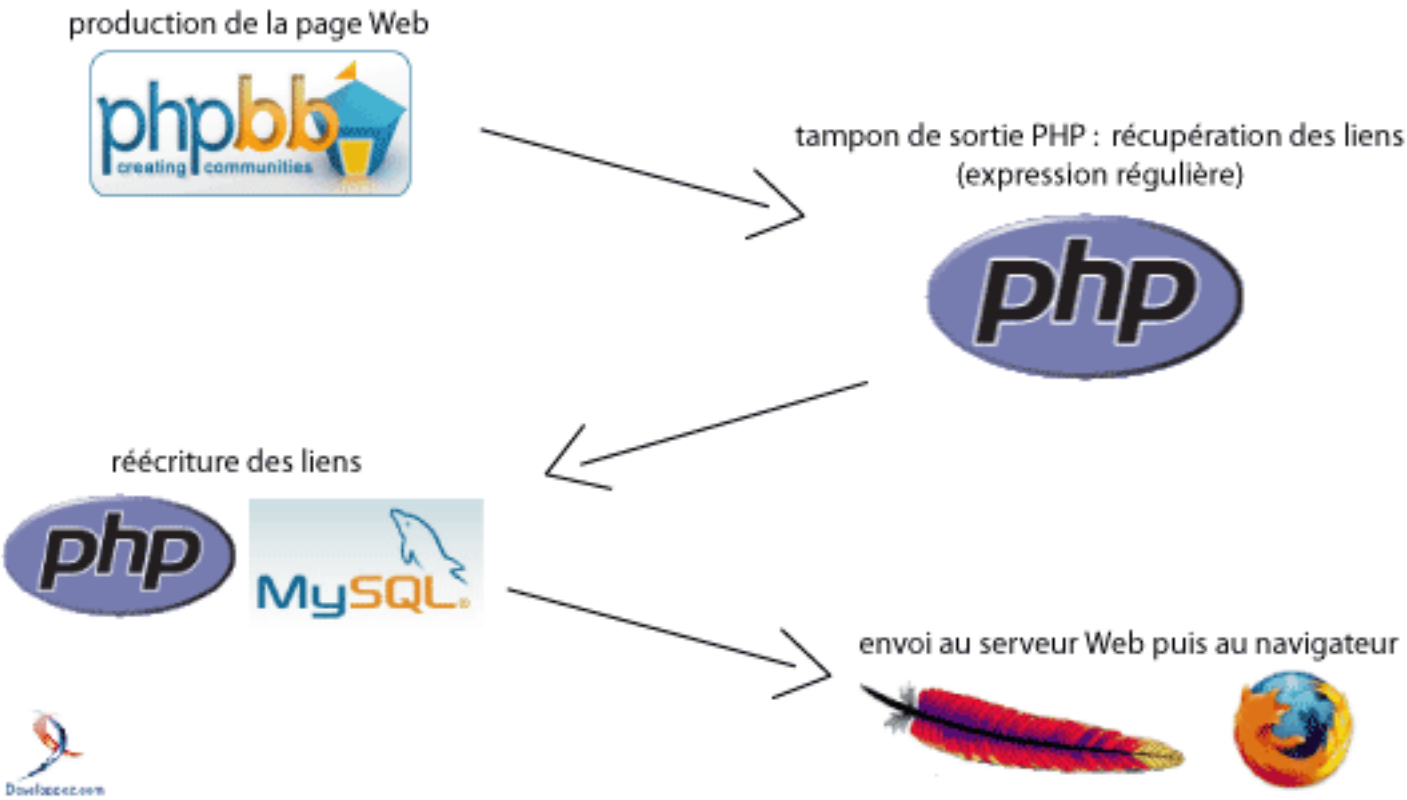
Fichier : .htaccess
```
RewriteRule ^chercher-([a-z]+).* /phpBB2/search.php?mode=$1 [L]
RewriteRule ^chercher.* /phpBB2/search.php [L]
```

Une fois ce fichier placé à la racine du forum, il s'occupera de traduire les adresses que nous lui enverrons plus tard en adresses classiques de *phpBB*.
Si une adresse réécrite (donc fictive) ne figure pas dans la liste des règles du fichier *.htaccess*, alors elle ne sera pas traduite, résultant en une erreur *404* ("page non trouvée").

Cette étape est terminée : passons maintenant à la suivante.

Seconde partie : modifier les pages envoyées au client

Ici, c'est déjà plus fun.
En fait, c'est simple à imaginer. La complexité vient de la grande quantité de tests à faire...



Étapes de la réécriture

Il y a deux manières de modifier les liens : *dynamiquement* et *statiquement*.

- **Statiquement**
  Les premières versions de l'*URL Rewriting* que j'ai mises en place sur le *Forum Cinéma* sont de la méthode statique, celle que je trouve trop peu pratique à mettre en place pour être viable... Il m'a fallu un moment pour intégrer tous les concepts et pour me motiver à refaire tout ça à ma sauce !
  La méthode statique est la plus facile et la plus longue, voire la plus frustrante à mettre en place. C'est aussi la moins informatisée et, par conséquent, la plus complexe à maintenir. Il s'agit de parcourir le code source du site complet et de modifier chaque lien à la main. C'est fastidieux et on en oublie toujours :/
  En plus, c'est aussi long à chaque fois que l'on effectue l'installation du Mod...
  L'inconvénient principal est que les mises à jour sont une vieille galère ! On oublie toujours un ancien lien quelque part, ou bien une mise à jour nous restaure un lien à l'ancienne...
- **Dynamiquement (que nous allons utiliser)**
  Cette méthode est largement plus simple à mettre en place mais également largement plus complexe à concevoir : pas de souci, je m'en suis chargé pour vous.
  Il faut penser à tous les cas de figure, chose qu'il aurait fallu faire avec la méthode statique de toute manière : ils peuvent varier d'un forum *phpBB* à l'autre, suivant les Mods installés et/ou l'humeur de l'administrateur.

L'idée est de laisser *phpBB* produire l'intégralité de son code *HTML*, puis d'intervenir en remplaçant d'un seul coup la totalité des liens de la page.
Pour y parvenir, nous allons utiliser la mise en tampon du code *HTML* (utilisation d'un "buffer"). En effet, *PHP* met à disposition quelques fonctions qui permettent de manipuler le contenu destiné au navigateur, avant de l'envoyer au navigateur. C'est exactement comme si nous mettions dans une variable tout le *HTML* produit par *PHP* : rien n'est envoyé au navigateur tant que nous ne faisons pas echo de cette variable.
Je le répète : je ne compte pas expliquer ici en détail la procédure de développement, simplement donner des points de repère.
L'idée générale est d'appliquer une recherche à base d'*expression régulière* à la totalité du code *HTML* qui est sur le point d'être envoyé au client : cela nous permet de récupérer les liens, de les modifier et enfin d'envoyer au client le code *HTML* modifié.

Voici l'*expression régulière* que j'utilise :

Expression régulière
```
#<a(.+)href="([^"]+)"([^>]*)>(.*)</a>#Usi
```

Elle comporte quatre parties entre parenthèses, dites capturées (ce sont celles qui nous intéressent) :

1 d'éventuelles *options* comme **class="..."** et **style:"..."**
2 l'adresse complète de la cible du lien
3 d'autres éventuelles *options* comme **class="..."** et **style:"..."**
4 le texte, l'ancre du lien (ce que l'on voit affiché dans la page Web)

Ici, les **#** sont les délimiteurs de l'*expression régulière*.
Seul le deuxième couple de parenthèses est vraiment important. Les trois autres sont là uniquement pour conserver des paramètres existants, rien de vraiment fondamental. En fait, pour extraire d'autres types de liens à l'aide de mon script final, il y a principalement besoin du 2° couple de parenthèses (logiquement, le 1° est également nécessaire afin de conserver l'ordre).

S'ensuivent les modificateurs :

- **U** demande à *PHP* de trouver autant de résultats que possible (sans quoi le **.+** au centre de l'expression pourrait inclure des balises *HTML*, ce qui nous donnerait une page finale avec un contenu complètement ésotérique - obliger *PHP* à ne pas être "*avare*" à l'aide du modificateur *Ungreedy* permet d'éviter ces éventuels désagréments).
- **s** demande au *point* d'inclure les sauts de ligne.
- **i** force l'expression à ne pas tenir compte de la *casse* (minuscules et majuscules).

Voilà, nous avons notre *expression régulière* qui nous permettra d'obtenir un joli tableau *PHP* des liens à modifier.
Il reste le gros du travail, à savoir énumérer tous les cas possibles de combinaisons de paramètres dans l'*URL*...
J'ai dit plus haut que cela pouvait aller à l'infini. C'est partiellement vrai : dans le cas de *phpBB* (comme de la quasi totalité des sites), il y a une limite au nombre de paramètres combinés à chaque page. En outre, nous ne nous occuperons pas toujours du contenu de ces paramètres, ce qui réduit encore la charge de travail.
Il reste néanmoins une belle quantité de situations à déterminer !
Le travail ici est similaire à ce qui a été fait pour le fichier *.htaccess*, seulement à l'envers : il faut constuire la chaîne sujet-3678,tuto-l-url-rewriting-reecriture-de-liens.htm à partir de viewtopic.php?t=3678. On peut également imaginer d'ajouter des informations comme le nom de l'auteur du message, la date... Puisque le nom final sera considéré comme un nom de fichier tout ce qu'il y a de plus normal (j'entends par là que cela ne se verra pas qu'il puisse y avoir des paramètres), nous pouvons ajouter des informations sans avoir peur que le moteur de recherche abandonne, au contraire : cela ajoute des mots clés dont il a tendance à être friand !
Attention hein, je vous vois venir... N'ajoutez pas le message complet au lien : cela n'aurait pas d'intérêt et cela finirait par vous coûter cher en bande passante, sans parler de l'impossibilité de poster un tel lien quelque part sans déformer affreusement la mise en page ! De plus il y a une limite...
Voici comment j'ai procédé.
J'ai pris individuellement chaque fichier de *phpBB* (*profile*, *viewtopic*, *etc.*) et je me suis demandé quels paramètres *GET* chacun d'eux admet.

Ensuite, c'est du cas par cas. La seule opération commune à tous les fichiers est de récupérer un titre cohérent en fonction du paramètre principal.
Dans le code, j'utilise des *switch* imbriqués. Oui, je sais : bouh, pas beau. Il m'arrive même d'en utiliser 3 niveaux (plus des *if/else*)... J'ai trouvé ça à la fois plus rapide à exécuter et plus clair à relire que de mettre uniquement des *if/else*.
Comme dans le reste du code source d'un forum *phpBB*, j'utilise la fonction *sprintf()* et un fichier de langue (*lang_urlrewrite.php*). Cela permet de définir des traductions pour ce Mod et, ainsi, d'adapter automatiquement la réécriture de liens selon les préférences définies dans le profil de chaque membre (un membre allemand pourra voir des *URLs* en allemand) !
Il ne reste plus qu'à mettre la touche finale en convertissant les mots clefs en caractères simples (les lettres de l'alphabet anglais, les chiffres et le trait d'union) avant de les ajouter au lien.
Note : j'ai encore utilisé les *expressions régulières* dans cette fonction.
Consommer chaud.

En résumé

Nous avons modifié toutes les adresses affichées dans toutes les pages Web produites par un forum *phpBB*.

Voici les problèmes posés par les liens classiques tels qu'ils étaient à l'origine :

- **Les paramètres des liens sont un problème simplement par leur présence**
  Solution apportée : les paramètres sont toujours présents mais cachés au sein du lien. Le moteur de recherche n'a pas vraiment de moyen de se rendre compte de leur présence et le lecteur humain a moins de difficultés à les lire et à les reconnaître, voire à les ignorer.
- **Il arrive que *phpBB* introduise des paramètres vides dans les liens**
  Solution apportée : uniquement les paramètres utiles sont introduits dans les nouveaux liens. Cela permet d'éviter le problème du *duplicate content*.
- **Il manque des mots clefs**
  Solution apportée : nous avons ajouté au texte du lien le sujet (ou le titre, le nom d'utilisateur, *etc.*) de la page liée.
- **L'ordre des paramètres n'est pas très pratique**
  Solution apportée : cela n'est plus un problème, dans la mesure où les paramètres sont cachés. Nous les avons également réorganisés afin qu'ils soient toujours présentés dans le même ordre pour un même type de lien. Dans une certaine mesure, cela facilite encore la lecture.
- **Certains moteurs de recherche ont des règles draconiennes à propos du *duplicate content***
  Solution apportée : les doublons ont été supprimés (i.e. viewtopic.php?p=502#502 est remplacé par viewtopic.php?t=43#502 puis réécrit).
  Rappel : le **#** n'indique pas un paramètre. Il est destiné au navigateur de l'internaute car il lui permet de positionner verticalement l'affichage de la page.

III - Exemples avec Apache & PHP

Ce code est minimaliste. Je n'ai mis aucune mesure de sécurité afin de ne pas nous détourner du sujet. De même, je n'ai prêté aucune attention aux tables créées dans la base de donnée : ce n'est pas l'objet de ce tutoriel. Pour un cas réel, il conviendrait d'appliquer votre stratégie de sécurité habituelle et de structurer correctement les données.
Dans tous les exemples, le fichier *.htaccess* redirige vers un répertoire que j'ai nommé "*urlr*" et que j'ai placé à la racine de mon serveur Web : à vous de modifier cette valeur pour qu'elle corresponde à votre configuration.

Ces exemples ont été testés avec succès sur ma machine de développement : j'utilise *EasyPHP 1.8*, c'est-à-dire *Apache 1.3.33* avec *PHP 4.3.10* et *MySQL 4.1.9*. Cela fonctionne malgré ces versions préhistoriques et je pense que c'est une bonne indication de la portabilité de l'*URL Rewriting*.

III-1 - Première partie : traduire les requêtes du client

Nos *URLs* peuvent être réécrites principalement de deux manières : au moyen de noms de fichiers fictifs ou bien en simulant une arborescence.

Traduire une URL fictive en une URL réelle

Fichier : index.php

```
<a href="tous-les-articles.html">Tous les articles</a><br />
<a href="article-1-sur-les-vaches.html">Les vaches</a><br />
<a href="article-2-sur-le-php.html">Le PHP</a><br />
<a href="article-3-sur-le-papier-recycle.html">Le papier recyclé</a><br /><br />

<?php

if(!empty($_GET['a'])){
    switch($_GET['a']){
        case 1:
            echo '<u>Article "Les vaches"</u> :<br />Meuh';
            break;

        case 2:
            echo '<u>Article "Le PHP"</u> :<br />echo "Hello World!";';
            break;

        default:
            echo '<u>Article "Le papier recyclé"</u> :<br />Pensons-y';
            break;
    }
}

?>
```

Fichier : .htaccess

```
DirectoryIndex index.php
RewriteEngine on

RewriteRule article-([0-9]+).* /urlr/index.php?a=$1 [L]
RewriteRule tous-les-articles.* /urlr/index.php [L]
```

Des deux grandes solutions qui s'offrent à nous pour réécrire les liens d'une page Web, celle-ci est ma préférée.

Traduire une URL fictive en une URL réelle en simulant une arborescence

Fichier : index.php

```
<a href="http://localhost/urlr/tous-les-articles.html">Tous les articles</a><br />
<a href="http://localhost/urlr/articles/1-sur-les-vaches.html">Les vaches</a><br />
<a href="http://localhost/urlr/articles/2-sur-le-php.html">Le PHP</a><br />
<a href="http://localhost/urlr/articles/3-sur-le-papier-recycle.html">Le papier recyclé</a><br /><br />

<?php

if(!empty($_GET['a'])){
    switch($_GET['a']){
        case 1:
            echo '<u>Article "Les vaches"</u> :<br />Meuh';
            break;

        case 2:
            echo '<u>Article "Le PHP"</u> :<br />echo "Hello World!";';
            break;

        default:
            echo '<u>Article "Le papier recyclé"</u> :<br />Pensons-y';
            break;
    }
}

?>
```

Fichier : .htaccess
```
DirectoryIndex index.php
RewriteEngine on

RewriteRule articles/([0-9]+).* /urlr/index.php?a=$1 [L]
RewriteRule tous-les-articles.* /urlr/index.php [L]
```

Cette solution me plaît moins que la précédente mais, puisqu'elle est tellement fréquente, j'ai décidé de la traiter ici. À vous de choisir celle que vous préférez de celle-ci ou de la précédente. Pour ma part, je poursuivrai ma présentation d'exemples avec la solution sans arborescence.
Je dois avouer que c'est ma grande fainéantise qui oriente mon choix... J'imagine que simuler une arborescence a tout un tas d'intérêts d'optimisation ! Si vous en avez le temps, je vous recommande de vous pencher sérieusement sur la question.

III-2 - Seconde partie : modifier les pages envoyées au client

Utiliser le tampon pour modifier tous les liens plus facilement

Squelette du script :

1 Initialiser la mise en tampon de la sortie standard
2 Afficher les liens à réécrire
3 Afficher le contenu en fonction du paramètre appelé (facultatif)
4 Récupérer le tampon et arrêter la mise en cache
5 Réécrire les liens
6 Afficher la page

Fichier : index.php
```php
<?php

// Initialiser la mise en tampon de la sortie standard
ob_start();

?>

<!-- Afficher les liens à réécrire -->
<a href="index.php">Tous les articles</a><br />
<a href="index.php?a=1">Les vaches</a><br />
<a href="index.php?a=2">Le PHP</a><br />
<a href="index.php?a=3">Le papier recyclé</a><br />

<?php

// Afficher le contenu en fonction du paramètre appelé (facultatif)
if(!empty($_GET['a'])){
    switch($_GET['a']){
        case 1:
            echo '<u>Article "Les vaches"</u> :<br />Meuh';
            break;

        case 2:
            echo '<u>Article "Le PHP"</u> :<br />echo "Hello World!";';
            break;

        default:
            echo '<u>Article "Le papier recyclé"</u> :<br />Pensons-y';
            break;
    }
}

// Récupérer le tampon et arrêter la mise en cache
$contents = ob_get_contents();
ob_end_clean();

$patterns[] = '<a href="index.php?a=1">Les vaches</a>';
```

Fichier : index.php
```php
$patterns[] = '<a href="index.php?a=2">Le PHP</a>';
$patterns[] = '<a href="index.php?a=3">Le papier recyclé</a>';

$new_urls[] = '<a href="article-1-les-vaches.html">Les vaches</a>';
$new_urls[] = '<a href="article-2-le-php.html">Le PHP</a>';
$new_urls[] = '<a href="article-3-le-papier-recycle">Le papier recyclé</a>';

// Réécrire les liens
$contents = str_replace($patterns, $new_urls, $contents);

// Afficher la page
echo $contents;

?>
```

Fichier : .htaccess
```
DirectoryIndex index.php
RewriteEngine on

RewriteRule article-([0-9]+).* /urlr/index.php?a=$1 [L]
RewriteRule tous-les-articles.* /urlr/index.php [L]
```

Observez ce qui a changé.
Non seulement les liens pointent vers des fichiers inexistants, mais nous avons également ajouté la propriété *title* à chacun d'eux. Notez également comme il est possible de mettre n'importe quoi à la suite du numéro d'article.
Et pourtant, cela fonctionne.

Il me paraît limpide que cet exemple peut être amélioré, au vu de la quantité d'information que nous dupliquons...
Étant donné que notre site piochera toutes ses données dans une base de données, autant s'y faire tout de suite !

Utiliser une base de données pour généraliser le script

Voici un exemple concret (indépendant de *phpBB*) utilisant le tampon et une base de données :

Squelette du script :

1. Initialiser la mise en tampon de la sortie standard
2. Se connecter à la base de données
3. Afficher les liens à réécrire
4. Afficher le contenu en fonction du paramètre appelé (facultatif)
5. Récupérer le tampon et arrêter la mise en cache
6. Récupérer les liens à l'aide d'une *expression régulière*
7. Parcourir les liens et les réécrire à l'aide de la base de données
8. Afficher la page

Code SQL pour créer la base de données
```sql
CREATE TABLE `article` (
  `id` INT(11) NOT NULL AUTO_INCREMENT,
  `author` VARCHAR(255) NOT NULL,
  `title` VARCHAR(255) NOT NULL,
  `text` TEXT NOT NULL,
  PRIMARY KEY (`id`)
);

INSERT INTO `article` (`author`, `title`, `text`)
VALUES
  ('Yogui', 'Les vaches', 'Meuh'),
  ('Yogui', 'Le PHP', 'echo "Hello world!";'),
  ('Yogui', 'Le papier recyclé', 'Pensons-y')
;
```

Fichier : index.php

```php
<?php

// Fonction pour nettoyer le titre afin de l'intégrer dans l'URL
function clean($string){
    return urlencode($string);
}


// Initialiser la mise en tampon de la sortie standard
ob_start();


// Se connecter à la base de données
mysql_connect('localhost', 'root', '')
    or die(__LINE__.' : '.mysql_error());

mysql_select_db('developpez')
    or die(__LINE__.' : '.mysql_error());


// Récupérer la liste des articles
$sql = 'SELECT `id`, `title`
        FROM `article`';

$result = mysql_query($sql)
    or die(__LINE__.' : '.mysql_error());

// Afficher les liens à réécrire
?>
<a href="index.php">Tous les articles</a><br />
<?php
while($article = mysql_fetch_assoc($result)){
    ?>
    <a href="index.php?a=<?php echo $article['id']; ?>"><?php echo $article['title']; ?></a>
    <br />
    <?php
}


// Afficher le contenu en fonction du paramètre appelé (facultatif)
if(!empty($_GET['a'])){
    $sql = 'SELECT `title`, `text`
            FROM `article`
            WHERE `id` = '.$_GET['a'];

    $result = mysql_query($sql)
        or die(__LINE__.' : '.mysql_error());

    if($article = mysql_fetch_assoc($result)){
        ?>
        <br /><br />
  <u>Article "<?php echo $article['title']; ?>"</u> :
  <br /><?php echo $article['text']; ?>
        <?php
    }
}


// Récupérer le tampon et arrêter la mise en cache
$contents = ob_get_contents();
ob_end_clean();


// Récupérer les liens à l'aide d'une expression régulière
if(preg_match_all(
 '#<a href="index.php\?a=([0-9]+)">(.+)</a>#Usi',
 $contents,
 $matches,
 PREG_SET_ORDER))
{
```

Fichier : index.php

```php
    // Parcourir les liens et les réécrire à l'aide de la base de données
    foreach($matches as $match){
        $pattern = $match[0];
        $article_id = $match[1];
        $anchor = $match[2];

        $sql = 'SELECT `title`, `text`
                FROM `article`
                WHERE `id` = '.$article_id;

        $result = mysql_query($sql)
            or die(__LINE__.' : '.mysql_error());

        if($article = mysql_fetch_assoc($result)){
            $new_url =
                '<a href="article-'.$article_id.'-sur-'.clean($article['title']).'.html" '
                    .'title="'.$article['title'].'">'
                    .$article['title']
                    .'</a>';
            $contents = str_replace($pattern, $new_url, $contents);
        }
    }
}


// Afficher la page
echo $contents;

?>
```

Fichier : .htaccess

```
DirectoryIndex index.php
RewriteEngine on

RewriteRule article-([0-9]+).* /urlr/index.php?a=$1 [L]
RewriteRule tous-les-articles.* /urlr/index.php [L]
```

Utiliser un fichier de langue pour traduire les URLs du site

Voici un exemple concret (indépendant de *phpBB*) utilisant le tampon, une base de données (multilingue) et un fichier de langue :

Squelette du script :

1 Initialiser la mise en tampon de la sortie standard
2 Déterminer dans quelle langue les URLs seront traduites
3 Se connecter à la base de données
4 Afficher les langues disponibles
5 Afficher les liens à réécrire
6 Afficher le contenu en fonction du paramètre appelé (facultatif)
7 Récupérer le tampon et arrêter la mise en cache
8 Récupérer les liens à l'aide d'une *expression régulière*
9 Parcourir les liens et les réécrire à l'aide de la base de données et du fichier de langue
10 Afficher la page

Code SQL pour créer la base de données

```sql
CREATE TABLE `language` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NOT NULL,
  PRIMARY KEY (`id`)
);

INSERT INTO `language` (`name`)
VALUES
```

Code SQL pour créer la base de données

```
  ('fr'),
  ('en'),
  ('es')
;

CREATE TABLE `article` (
  `id` int(11) NOT NULL auto_increment,
  `author` VARCHAR(255) NOT NULL,
  PRIMARY KEY (id)
);

INSERT INTO `article` (`author`)
VALUES
  ('Yogui'),
  ('Yogui'),
  ('Yogui')
;

CREATE TABLE `article_lang` (
  `article_id` int(11) NOT NULL default '0',
  `lang_id` int(11) NOT NULL default '0',
  `title` varchar(255) NOT NULL default '',
  `text` text NOT NULL,
  PRIMARY KEY (`article_id`, `lang_id`)
);

INSERT INTO `article_lang` (`article_id`, `lang_id`, `title`, `text`)
VALUES
  (1, 1, 'Les vaches', 'Meuuh'),
  (1, 2, 'Cows', 'Muuh'),
  (1, 3, 'Las vacas', 'Muu'),
  (2, 1, 'Le PHP', 'echo "Bonjour le monde !";'),
  (2, 2, 'PHP', 'echo "Hello world!";'),
  (2, 3, 'El PHP', 'echo "¡Hola mundo!";'),
  (3, 1, 'Le papier recyclé', 'Pensons-y'),
  (3, 2, 'Recycled paper', 'Think about it'),
  (3, 3, 'Papel reciclado', 'Piensa en ello')
;
```

Fichier : index.php

```php
<?php

// Fonction pour nettoyer le titre afin de l'intégrer dans l'URL
function clean($string){
    return urlencode($string);
}


// Initialiser la mise en tampon de la sortie standard
ob_start();


// Déterminer dans quelle langue les URLs seront traduites
if(!empty($_GET['lang']) and is_file('lang_'.$_GET['lang'].'.php')){
    define('LANG_ID', $_GET['lang']);
}
else{
    define('LANG_ID', 'fr');
}
require('lang_'.LANG_ID.'.php');


// Se connecter à la base de données
mysql_connect('localhost', 'root', '')
    or die(__LINE__.' : '.mysql_error());

mysql_select_db('developpez')
    or die(__LINE__.' : '.mysql_error());
```

Fichier : index.php

```php
// Afficher les langues disponibles
$sql = 'SELECT `name`
        FROM `language`';

$result = mysql_query($sql)
    or die(__LINE__.' : '.mysql_error());

while($language = mysql_fetch_assoc($result)){
    ?>
    [ <a href="index.php?lang=<?php echo $language['name']; ?>">
    <?php echo $language['name']; ?>
    </a> ]
    <?php
}
echo '<br /><br />';


// Récupérer la liste des articles
$sql = 'SELECT a.`id`, a.`author`, al.`title`, al.`text`
        FROM `article_lang` AS al
        INNER JOIN `article` AS a ON al.`article_id` = a.`id`
        INNER JOIN `language` AS l ON al.`lang_id` = l.`id`
        WHERE l.`name` = "'.LANG_ID.'"';

$result = mysql_query($sql)
    or die(__LINE__.' : '.mysql_error());

// Afficher quelques liens à partir de la base de données
while($article = mysql_fetch_assoc($result)){
    ?>
    <a href="index.php?a=<?php echo $article['id']; ?>"><?php echo $article['title']; ?></a>
    <br />
    <?php
}


// Afficher le contenu en fonction du paramètre appelé (facultatif)
if(!empty($_GET['a'])){
 $sql = 'SELECT a.`id`, a.`author`, al.`title`, al.`text`
         FROM `article_lang` AS al
            INNER JOIN `article` AS a ON al.`article_id` = a.`id`
            INNER JOIN `language` AS l ON al.`lang_id` = l.`id`
            WHERE a.`id` = '.$_GET['a'].'
                AND l.`name` = "'.LANG_ID.'"';

    $result = mysql_query($sql)
        or die(__LINE__.' : '.mysql_error());

    if($article = mysql_fetch_assoc($result)){
        ?>
        <br /><br />
  <u>Article "<?php echo $article['title']; ?>"</u> :
  <br /><?php echo $article['text']; ?>
        <?php
    }
}


// Récupérer le tampon et arrêter la mise en cache
$contents = ob_get_contents();
ob_end_clean();


// Récupérer les liens à l'aide d'une expression régulière
if(preg_match_all(
 '#<a href="index.php\?a=([0-9]+)">(.+)</a>#Usi',
 $contents,
 $matches,
 PREG_SET_ORDER))
{
```

Fichier : index.php

```php
    // Parcourir les liens et les réécrire à l'aide de la base de données
    foreach($matches as $match){
        $pattern = $match[0];
        $article_id = $match[1];
        $anchor = $match[2];

  $sql = 'SELECT a.`id`, a.`author`, al.`title`, al.`text`
          FROM `article_lang` AS al
                INNER JOIN `article` AS a ON al.`article_id` = a.`id`
                INNER JOIN `language` AS l ON al.`lang_id` = l.`id`
                WHERE a.`id` = '.$article_id.'
                    AND l.`name` = "'.LANG_ID.'"';

        $result = mysql_query($sql)
            or die(__LINE__.' : '.mysql_error());

        if($article = mysql_fetch_assoc($result)){
            $new_url    = sprintf($lang['url'], $article_id, clean($article['title'])).'.html';
            $new_title  = sprintf($lang['title'], $article['title']);
        }
        else{
            $new_url    = $lang['index'].'.html';
            $new_title  = '';
  }

        $new_link   = '<a href="'.$new_url.'" title="'.$new_title.'">'.$article['title'].'</a>';
        $contents   = str_replace($pattern,
                                  $new_link,
                                  $contents);
    }
}


// Afficher la page
echo $contents;

?>
```

Fichier : lang_fr.php

```php
<?php

$lang = array();
$lang['index']  = 'tous-les-articles';
$lang['url']    = 'article-%d-sur-%s';
$lang['title']  = 'Article : %s';

?>
```

Fichier : lang_en.php

```php
<?php

$lang = array();
$lang['index']  = 'all-the-articles';
$lang['url']    = 'article-%d-about-%s';
$lang['title']  = 'Article: %s';

?>
```

Fichier : lang_es.php

```php
<?php

$lang = array();
$lang['index']  = 'todos-los-articulos';
$lang['url']    = 'articulo-%d-sobre-%s';
$lang['title']  = 'Artículo: %s';

?>
```

Fichier : .htaccess
```
DirectoryIndex index.php
RewriteEngine on

RewriteRule article-([0-9]+)-sur.* /urlr/index.php?a=$1 [L]
RewriteRule tous-les-articles.* /urlr/index.php [L]

RewriteRule article-([0-9]+)-about.* /urlr/index.php?a=$1&lang=en [L]
RewriteRule all-the-articles.* /urlr/index.php&lang=en [L]

RewriteRule articulo-([0-9]+)-sobre.* /urlr/index.php?a=$1&lang=es [L]
RewriteRule todos-los-articulos.* /urlr/index.php&lang=es [L]
```

IV - Conclusion

IV-1 - Pour aller plus loin

Le présent article, bien que complet, laisse de la place à davantage de recherche. J'ai couvert tout ce qui a directement trait aux liens d'une page Web, comment les réécrire et comment traiter les demandes en provenance du client mais je n'ai pas évoqué les redirections...
Ce sera l'objet de futurs tutoriels ^^

Des améliorations : les prochaines mises à jour du tutoriel

- Pour les sites qui sont déjà référencés par les moteurs de recherche, il faudra agrémenter la *réécriture de liens* par un script qui s'occupera de gérer le *duplicate content* en envoyant un "*header 301 : Moved permanently*". Cela évitera des résultats catastrophiques pour le référencement du site...
- Comment appliquer cette réécriture à tous les liens du site ? Mon script permet de réécrire toutes les *URLs* d'un forum *phpBB* mais j'ai donné peu d'informations sur la manière d'y parvenir.
- *phpBB* comporte quelquefois l'attribut *title* dans ses liens. J'ai préféré peu modifier le code de *phpBB* afin de rendre l'installation de mon Mod la plus simple possible. Cependant, il ne faut pas que cela nous empêche de remplacer ces textes peu optimisés : ne pas y toucher serait même un manque à gagner !

D'autres cours

SEO :

- **Introduction à la SEO**, par Guillaume Rossolini ;
- **Techniques de SEO dites de "white hat"**, par Guillaume Rossolini ;
- **Réécriture de liens (URL Rewriting)**, par Guillaume Rossolini ;
- **Faire évoluer sa réécriture de liens**, par Guillaume Rossolini.

Thèmes divers :

- **Les principales utilisations du htaccess avec Apache**, par Cédric Chatelain ;
- **La réécriture d'URL avec Apache**, par f-demu01 ;
- **Les expressions régulières en PHP**, par Guillaume Rossolini.

IV-2 - Le Mod phpBB

J'ai commencé à écrire ce tutoriel lorsque j'étais en train de coder le Mod *URL Rewriting* pour le *Forum Cinéma*. Il serait très égoïste de ma part de ne pas le mettre à disposition...

Attention : je ne garantis absolument aucun résultat de SEO. Je ne garantis pas même que votre forum fonctionne aussi bien qu'avant, une fois ce Mod installé, surtout s'il était déjà moddé. Il faudra y mettre du vôtre à partir de maintenant.

La procédure d'installation est la suivante :

1 Télécharger l'archive :
Mod URL Rewriting pour phpBB 2 (version française) : [**FTP**] ou [**HTTP**]
Mod URL Rewriting pour phpBB 2 (version anglaise) : [**FTP**] ou [**HTTP**]

2 Ouvrir : */.htaccess*

3 Trouver :

```
phpbb_test/
```

4 Remplacer toutes les occurences par le nom du répertoire de votre forum (à partir de la racine du site)

5 Ouvrir : */includes/page_header.php*

6 Trouver :

```
//
// Parse and show the overall header.
```

7 Ajouter, avant :

```
ob_start();
```

8 Ouvrir : */includes/page_tail.php*

9 Trouver :

```
//
// Close our DB connection.
//
$db->sql_close();
```

10 Ajouter, avant :

```
//
// Retrieve output buffer contents, then clear the buffer without displaying anything
//
$contents = ob_get_contents();
ob_end_clean();


require_once('functions_urlrewrite.'.$phpEx);
//
// Find all the links in the page
// There are 4 parenthesized parts in this regular expression (the last can be overlooked):
//   #1: there might be an option like class="..."
//   #2: full path to the target Web page
//   #3: #1 bis
//   #4: text of the link
//
if(preg_match_all('#<a(.+)href="([^"]+)"([^>]*)>(.+)</a>#Usi', $contents, $matches,
 PREG_SET_ORDER)){
    $contents = rewrite_urls($contents, $matches, 'a');
}

if(preg_match_all('#<form(.*)action="([^"]+)"([^>]*)>#Usi', $contents, $matches,
 PREG_SET_ORDER)){
    $contents = rewrite_urls($contents, $matches, 'form');
}

if(preg_match_all('#<link rel(.*)href="([^"]+)"([^>]*)>#Usi', $contents, $matches,
 PREG_SET_ORDER)){
    $contents = rewrite_urls($contents, $matches, 'link rel');
}
```

```
    echo $contents;
```

11 Sauvegarder tous les fichiers, uploader sur le serveur et tester

IV-3 - Épilogue

Dans les grandes lignes, je pense avoir tout dit de la méthode que j'ai adoptée pour réécrire les adresses sur le *Forum Cinéma*.
Avant de vous lancer à installer ce Mod sur votre site, je ne peux que vous inciter à lire tout ce que vous pouvez sur le sujet (notamment sur **Développez.com** et **Abondance.fr**) et, surtout, à bien réfléchir à la forme que vous souhaitez donner à vos *URLs*. Gardez bien à l'esprit qu'il s'agit principalement d'améliorer le référencement d'un site auprès des moteurs de recherche en leur facilitant la navigation, mais que ces mêmes moteurs de recherche ne pourront pas savoir que voir-le-sujet-15.php et voir-le-sujet-15-qui-parle-des-vaches-et-qui-a-ete-lance-le-23-septembre-2005-par-so.php sont en fait la même page et que vous avez changé votre gestion de l'*URL Rewriting* entre deux de leurs passages chez vous (euh, désolé So, il est 6h30 du mat' et j'ai toujours pas dormi... bref). Rapport aux *backlinks* et au *duplicate content*, le pire qu'il puisse arriver est de changer l'*URL Rewriting* de son site (et je ne parle pas de la compatibilité descendante du fichier *.htaccess*, ça devient rapidement la foire : sur le *Forum Cinéma*, nous en sommes les premières victimes !).

Une dernière chose : le fichier *.htaccess*, s'il est correctement écrit, n'empêchera pas les liens classiques de fonctionner. Ainsi, viewtopic.php?t=3678 et sujet-3678,tuto-l-url-rewriting.htm sont tout autant valides l'un que l'autre. Cela peut s'avérer bien pratique mais cela veut aussi dire qu'il faut faire attention notamment au fichier *robots.txt* et y inclure toutes les formes valides de liens. Si vous avez réécrit *search.php* en *chercher.html* ou *chercher-les-messages-de-x.html* suivant le cas, votre fichier *robots.txt* doit prévoir d'interdire l'accès à ces trois formes de liens.

Souvenez-vous qu'une mauvaise réécriture des adresses aura pour conséquence la mise hors service partielle, voire complète, de votre site !

Je vous laisse vous documenter pleinement. N'installez ce Mod sur votre forum *en production* qu'une fois que vous l'aurez longuement testé sur un forum de test.

Démonstration sur le **Forum Cinéma**.