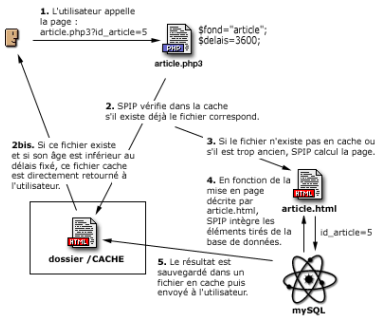
http://clx.asso.fr/spip/?SPIP-Principe-general

SPIP : Principe général

- L'Association - Libre et Libertés Numériques -



Date de mise en ligne : jeudi 20 mars 2003

Alors, un SPIP, comment ça marche ? Hein ?
L'article suivant va vous l'expliquer, en multiples blablablas, qui serviront - peut-être.

Tout le contenu d'un site géré sous SPIP est installé dans une base de données mySQL (dans la partie protégée par mot de passe /ecrire). Pour présenter ces informations aux visiteurs du site, il faut donc réaliser l'opération qui consiste à récupérer les informations, à les organiser et à les mettre en page, afin de délivrer une page HTML.

Cette opération est traditionnellement assez pénible :
[-] il faut connaître PHP et mySQL, et effectuer des routines relativement complexes ;
[-] l'intégration de telles routines dans une mise en page HTML élaborée est assez pénible ;
[-] le recours systématique à des requêtes mySQL à chaque affichage d'une page est gourmand en ressources, ralentit la visite et, dans des cas extrêmes, provoque des plantages du serveur.

SPIP propose ici une solution complète pour contourner ces difficultés :
[-] la mise en page du site est effectuée au moyen de pages HTML nommées squelettes, contenant des instructions simplifiées permettant d'indiquer où et comment se placent les informations tirées de la base de données dans la page ;
[-] un système de cache permet de stocker chaque page et ainsi d'éviter de provoquer des appels à la base de données à chaque visite. Non seulement la charge sur le serveur est réduite, la vitesse très largement accélérée, de plus un site sous SPIP reste consultable même lorsque la base mySQL est plantée.

Pour chaque type de document, un couple de fichiers

L'intérêt (et la limite) d'un système de publication automatisé, c'est que l'on ne va pas redéfinir une interface différente en HTML pour chaque page isolée. Par exemple, toutes les articles bénéficieront de la même interface, simplement le système placera des informations différentes dans ce graphisme (on verra plus loin que SPIP autorise cependant une certaine souplesse).

L'avantage de cette manière de procéder est évident : on définit un format-type (squelette) pour, par exemple, tous les articles, et le système fabriquera chaque page individuelle en plaçant automatiquement le titre, le texte, les liens de navigation... de chaque article.

Pour chaque type de document, SPIP vous demande deux fichiers : un fichier .php3 et un fichier .html . Lors de l'installation de SPIP, vous trouverez ainsi les couples : « article.php3 / article.html », « rubrique.php3 / rubrique.html », etc. Vous pouvez naturellement modifier ces couples, et en créer d'autres.

Le principe de fonctionnement du cache

L'appel d'une page spécifique se fait par l'intermédiaire du fichier .php3. Par exemple, pour appeler l'article 5, l'URL correspond est :

```
article.php3?id_article=5
```

[http://clx.asso.fr/spip/local/cache-vignettes/L400xH337/art877-1-25d2d.gif]

1. Le fichier appelé est donc article.php3, avec en variable id_article=5.

2. Le fichier article.php3 est un fichier PHP ; sa première tâche consiste à vérifier dans le dossier /CACHE sur votre serveur, s'il existe déjà un fichier correspondant à cet article.

2bis. Si un tel fichier existe dans /CACHE, article.php3 vérifie sa date de création. Si ce fichier est suffisamment récent, il le retourne directement à l'utilisateur. Le processus de consultation est alors terminé.

3. S'il n'existe pas un tel fichier dans /CACHE (première visite sur cet article, par exemple), ou si son âge est trop ancien, SPIP démarre le calcul de cette page.

4. C'est alors la page article.html qui est chargée et analysée. Cette page contient la mise en page correspondant à ce type de document. Il s'agit de HTML complété d'indications permettant de placer les éléments tirés de la base de données. En fonction des éléments requis par article.html, SPIP va chercher les informations nécessaires tirées de la base de données mySQL et les insérer aux endroits prévus.

5. Un fichier est ainsi fabriqué, à partir de la description contenue dans article.html, avec les éléments tirés de la base de données. Ce fichier est alors sauvegardé dans le dossier /CACHE et renvoyé au visiteur.

En cas de plantage de la base de données, c'est forcément le fichier en cache qui est retourné, même s'il est « trop âgé ».

Lors d'une visite suivante, si le délais entre les deux visites est suffisamment court, c'est donc ce nouveau fichier stocké dans /CACHE qui est retourné, sans avoir à faire un nouveau calcul à partir de la base de données. En cas de plantage de la base de données, c'est forcément le fichier en cache qui est retourné, même s'il est « trop âgé ».

Remarque. On voit ici que chaque page du site est mise en cache individuellement, et chaque recalcul est provoqué par les visites du site. Il n'y a pas, en particulier, un recalcul de toutes les pages du site d'un seul coup à échéance régulière (ce genre de « grosse manoeuvre » ayant le bon goût de surcharger le serveur et de le faire parfois planter).

Le fichier .PHP3

Le fichier .php3 est très simple. Par exemple, `article.php3` contient uniquement :

```
<?
 $fond = "article";
 $delais = 24 * 3600;

 include ("inc-public.php3");
 ?>
```

Son seul but est donc de fixer deux variables ($fond et $delais) et d'appeler le fichier qui déclenche le fonctionnement de SPIP (inc-public.php3).

La variable $fond est le nom du fichier qui contient la description de la mise en page (le squelette). Ici, puisque $fond="article", le fichier de description sera contenu dans article.html. (Notez bien que, dans la variable $fond, on

n'indique pas la terminaison .html.)

Remarque. L'intérêt de choisir soi-même le nom du fichier de squelette (que l'on aurait pu déduire automatiquement du nom du fichier .php3) est, si nécessaire, d'utiliser un autre nom. Cela pour ne pas écraser, éventuellement, des fichiers HTML qui subsisteraient d'une ancienne version du site que l'on ne souhaite pas supprimer. S'il existe, d'une ancienne version du site, un fichier article.html que l'on ne souhaite pas effacer, on utilisera par exemple un fichier squelette pour SPIP intitulé spip-article.html, et on fixera dans article.php3 : $fond="spip-article".

La variable $delais est l'âge maximum pour l'utilisation du fichier stocké en /CACHE. Ce délai est fixé en secondes. Un délai de 3600 correspond donc à une heure ; un délai de 24*3600 est donc de 24 heures.

On jouera sur cette valeur en fonction de la fréquence des ajouts de contenu du site (nouveaux articles, nouvelles brèves...). Un site actualisé plusieurs fois par jour pourra adopter un délai d'une heure ; un site publiant quelques articles par semaine pourra adopter un délai nettement plus long. De même, le contenu des pages est important : si vous insérez la syndication du contenu de sites fréquemment mis à jour, vous souhaiterez sans doute adapter votre propre délais à celui des sites référencés.

Remarque. Certains webmestre succombent à la tentation de fixer des délais dérisoires (quelques secondes), pour que le site corresponde très exactement, à chaque instant, aux dernières modifications de la base de données. Dans ce cas, vous perdez tous les avantages du système de cache : les visites sont nettement ralenties et, à l'extrême, sur des sites très fréquentés, vous pouvez provoquer des plantages de la base de données (ou vous faire virer par votre hébergeur parce que vous monopolisez la puissance de sa machine...).

Remarque. Une autre raison qui semble pousser les webmestres à fixer des délais très courts est la présence des forums sur leur site. En effet, pour que les contributions s'affichent dès qu'elles sont postées, ils pensent nécessaire de réduire le délais de recalcul. C'est inutile : SPIP gère cet aspect automatiquement ; lorsqu'une contribution à un forum est postée, la page correspondante est effacée du cache, et recalculée immédiatement, quel que soit le délai fixé pour cette page.

Le fichier .HTML

Dans SPIP, nous appelons les fichiers .html les squelettes. Ce sont eux qui décrivent l'interface graphique de vos pages.

Ces fichiers sont rédigés directement en HTML, auquel on ajoute des instructions permettant d'indiquer à SPIP où il devra placer les éléments tirés de la base de données (du genre : « placer le titre ici », « indiquer à cet endroit la liste des articles portant sur le même thème »...).

Les instructions de placement des éléments sont rédigées dans un langage spécifique, qui fait l'objet du présent manuel d'utilisation. Ce langage constitue par ailleurs la seule difficulté de SPIP.

« Encore un langage ? » Hé oui, il va vous falloir apprendre un nouveau langage. Il n'est cependant pas très compliqué, et il permet de créer des interfaces complexes très rapidement. Par rapport au couple PHP/mySQL, vous verrez qu'il vous fait gagner un temps fou (surtout : il est beaucoup plus simple). C'est un markup language, c'est-à-dire un langage utilisant des balises similaires à celles du HTML.

Remarque. De la même façon que l'on apprend le HTML en s'inspirant du code source des sites que l'on visite, vous

pouvez vous inspirer des squelettes utilisés sur d'autres sites fonctionnant sous SPIP. Il suffit d'aller chercher le fichier « .html » correspondant. Par exemple, vous pouvez voir le squelette des articles d'uZine (visualisez le code source pour obtenir le texte du squelette).

Une interface différente dans le même site

Tout d'abord, notez qu'il est possible de créer des couples de fichiers pour le même élément logique (articles, rubriques, ...). Par exemple, vous pouvez créer des fichiers pour visiter un même article avec des interfaces différentes : article.php3/html pour le format normal, imprimer.php3/html pour le même article dans un format adapté à l'impression, article-texte.php3/html pour l'article dans un format texte (adapté aux mal-voyants par exemple), article-heavy.php/html avec une interface lourdingue adaptée au haut-débit, etc.

Vous pouvez, pour un même type de document, créer des squelettes différents selon les rubriques du site.

Une interface différente selon les rubriques. Vous pouvez, pour un même type de document, créer des squelettes différents selon les rubriques du site. Il s'agit de créer simplement de nouveaux fichiers .html en fonction des rubriques (inutile, ici, de modifier le fichier .php3, on se contente de jouer sur les noms des fichiers squelettes.

Il suffit de compléter le nom du fichier squelette de « -numéro » (tiret suivi d'un numéro de rubrique). Par exemple, si vous créez un fichier : article-60.html, tous articles contenus dans la rubrique 60 utiliseront ce squelette (et non plus le squelette par défaut article.html). Notez bien : le numéro indiqué est celui d'une rubrique. Si cette rubrique 60 contient des sous-rubriques, les articles contenus dans ces sous-rubriques utiliseront également ce nouveau squelette.

Remarque. Dans notre exemple, on aura certainement également intérêt à créer un rubrique-60.html, un breve-60.html, etc. pour accompagner le changement de mise en page de cette rubrique.

Une interface pour une seule rubrique. (SPIP 1.3) On peut créer une interface qui s'applique à une rubrique, mais pas à ses sous-rubriques. Pour cela, il faut créer un fichier : article=60.html, qui s'appliquera uniquement aux articles de la rubrique 60, mais pas à ses sous-rubriques.

Que peut-on mettre dans un fichier .HTML

Les fichiers .html sont essentiellement des fichiers « texte », complétés d'instructions de placement des éléments de la base de données.

SPIP analyse uniquement les instructions de placement des éléments de la base de données (codées selon le langage spécifique de SPIP) ; il se contrefiche de ce qui est placé dans ce fichier et qui ne correspond pas à ces instructions.

Leur contenu essentiel est donc du HTML. Vous déterminez la mise en page, la version du HTML désiré, etc. Vous pouvez évidemment y inclure des feuilles de style (CSS), mais également du JavaScript, du Flash... en gros : tout ce qu'on place habituellement dans une page Web.

Mais vous pouvez également (tout cela n'est jamais que du texte) créer du XML (par exemple, « backend.php3/html » génère du XML).

Plus original : toutes les pages retournées au visiteur sont tirées du /CACHE par une page écrite en PHP. Vous pouvez donc inclure dans vos squelette des instructions en PHP, elles seront exécutées lors de la visite. Utilisée de manière assez fine, cette possibilité permet une grande souplesse à SPIP, que vous pouvez ainsi compléter (par exemple ajouter un compteur, etc.), ou même faire évoluer certains éléments de mise en page en fonction des informations tirées de la base de données.

Post-scriptum :

L'article original a été écrit par l'équipe de SPIP. Il est bien sûr consultable directement sur le site principal.