Documentation SPIP

Quelques exemples de boucles

Quelques exemples de boucles

Quelques exemples de boucles, et autres petits morceaux de squelettes pour apprendre à utiliser le langage de SPIP en résolvant des cas particuliers.

Les jointures entre tables

juillet 2010 — maj : août 2010

exemples de boucles produisant une jointure multiple entre 3 tables

nous voulons, **en une boucle**, récupérer les informations des articles d'une (ou plusieures) rubrique à laquelle est affecté un mot clef (que nous connaissons).

```
<BOUCLE_a(ARTICLES spip_mots_rubriques spip_mots) {titre_mot=truc}>
```

produira la requête sql suivante :

```
 SELECT articles.id_rubrique,
        articles.id_article,
        ...
        articles.lang
   FROM spip_articles AS `articles`
        INNER JOIN spip_mots_rubriques AS L1
                ON L1.id_rubrique = articles.id_rubrique
        INNER JOIN spip_mots AS L2
                ON L2.id_mot = L1.id_mot
  WHERE articles.statut = 'publie'
    AND L2.titre = 'truc'
GROUP BY articles.id_article
```

Nous voulons *en une boucle* sélectionner aléatoirement un document dans un secteur. À savoir que ce secteur ne contient pas d'articles, seulement des rubriques avec des documents liés (photothèque).

```
<BOUCLE_d(spip_documents_liens rubriques)
         {objet = rubrique}
         {rubriques.id_secteur = 13}
         {par hasard}
         {0, 1}>
 #LOGO_DOCUMENT
</BOUCLE_d>
```

- la table spip_documents_liens est une table qui recense les liens (jointures) entre un document et un objet (article, rubrique, mot, site...).

des entrées type de cette table pourraient ressembler à :

id_document	id_objet	objet	vu
14	36	article	non
363	66	rubrique	non
...	...	...	...

nous allons donc chercher dans cette table les documents rattachés à une rubrique **`{objet = rubrique}`**

mais nous voulons aussi que cette rubrique soit descendante de la rubrique secteur d'id 13

nous devons donc établir dans notre requête une **jointure** entre la table spip_documents_liens et spip_rubriques

cette jointure sera faite sur d'un côté :
l'**id_objet** de spip_documents_liens, de l'autre
l'**id_rubrique** de spip_rubriques

pour cela nous indiquons à spip que nous voulons cette jointure spécifique en donnant les 2 tables à la boucle `<BOUCLE_d(spip_documents_liens rubriques)...`

enfin, pour spécifier notre restriction concernant le secteur 13, nous précisons le critère `{rubriques.id_secteur = 13}` en spécifiant explicitement le nom complet du champ (nom de la table inclu) pour que la requête n'aille pas chercher un champ *spip_documents_liens.id_secteur* qui n'existe pas.

- à partir de cette boucle, donc, nous avons alors accès à tous les champs de spip_documents_liens **et** tous ceux de spip_rubriques :

```
#ID_DOCUMENT
#ID_OBJET
#OBJET
#VU

#ID_RUBRIQUE
#ID_PARENT
#TITRE
#DESCRIPTIF
#TEXTE
#ID_SECTEUR
...
```

- la requête sql produite par notre boucle :

```
 SELECT rand() AS alea,
        spip_documents_liens.id_document
   FROM spip_documents_liens AS `spip_documents_liens`
        INNER JOIN spip_rubriques AS L1
                ON L1.id_rubrique = spip_documents_liens.id_objet
               AND spip_documents_liens.objet = 'rubrique'
  WHERE spip_documents_liens.objet = 'rubrique'
    AND L1.id_secteur = 13
GROUP BY spip_documents_liens.id_document,
        spip_documents_liens.id_objet,
        spip_documents_liens.objet
ORDER BY alea
  LIMIT 0,1
```

- enfin, la balise #LOGO_DOCUMENT nous retournera le source html :

```
<img src='local/cache-vignettes/L135xH150/Image_10-d84e2.png'
width='135' height='150' style='height:150px;width:135px;' alt=''
class='spip_logos' />
```

et si nous voulions **le nom du fichier** document, que nous ne pouvons avoir directement avec cette requête car le champ spip_documents.fichier n'est pas relevé (pas de jointure avec la table spip_documents), **alors il faut déclarer une jointure supplémentaire** sur la table spip_documents :

```
<BOUCLE_d(spip_documents_liens documents rubriques)
         {objet = rubrique}
         {rubriques.id_secteur = 13}
         {par hasard}
         {0, 1}>
#LOGO_DOCUMENT / #FICHIER
</BOUCLE_d>
```

requête produite :

```
 SELECT rand() AS alea,
        spip_documents_liens.id_document,
        L2.fichier
   FROM spip_documents_liens AS `spip_documents_liens`
        INNER JOIN spip_documents AS L2
                ON L2.id_document = spip_documents_liens.id_document
        INNER JOIN spip_rubriques AS L1
                ON L1.id_rubrique = spip_documents_liens.id_objet
               AND spip_documents_liens.objet='rubrique'
  WHERE spip_documents_liens.objet = 'rubrique'
    AND L1.id_secteur = 13
GROUP BY spip_documents_liens.id_document,
        spip_documents_liens.id_objet,
        spip_documents_liens.objet
ORDER BY alea
  LIMIT 0,1
```

attention n° 1 :
- ici, nous avons *presque* accès à tous les champs des 3 tables
« presque » car attention aux champs homonymes :
les #TITRE, #DESCRIPTIF, #MAJ, #STATUT et #DATE affichés, qui sont des champs homonymes dans spip_documents **et** spip_rubriques, seront ceux de spip_documents (*table première* de la requête) !

attention n° 2 :
- les écritures de nom de table

si :

```
 <BOUCLE_d(DOCUMENTS_LIENS
 <BOUCLE_d(documents_liens
  <BOUCLE_d(SPIP_DOCUMENTS_LIENS
  <BOUCLE_d(spip_documents_liens
```

sont équivalentes ;

tout comme :

```
 <BOUCLE_d(documents_liens documents rubriques
 <BOUCLE_d(documents_liens documents spip_rubriques
 <BOUCLE_d(documents_liens documents RUBRIQUES
```

sont équivalentes ;

et encore :

```
<BOUCLE_d(documents_liens documents
<BOUCLE_d(documents_liens spip_documents
<BOUCLE_d(spip_documents_liens spip_documents
<BOUCLE_d(spip_documents_liens documents
```

sont aussi équivalentes ;

il n’en va pas de même avec :

```
<BOUCLE_d(DOCUMENTS_LIENS DOCUMENTS
<BOUCLE_d(documents_liens DOCUMENTS
<BOUCLE_d(SPIP_DOCUMENTS_LIENS DOCUMENTS
<BOUCLE_d(spip_documents_liens DOCUMENTS
```

où la mise en capitales de *documents* occasionne une perte de jointure entre spip_documents_liens et spip_documents

Classer selon la date ou selon un ordre imposé

Septembre 2003 — maj : Novembre 2007

Nous souhaitons trier les articles de façon différenciée, de cette façon :

- dans certaines rubriques, les articles sont publiés les uns à la suite des autres ; on veut donc les présenter selon l'ordre chronologique : les plus récents en début de liste, les plus anciens en fin de liste ;
- dans d'autres rubriques, on souhaite afficher les articles dans un ordre précis, en les numérotant ; sur le site public, on veut donc les présenter selon cet ordre indiqué par la numérotation ;
- enfin, dans les autres rubriques, les articles doivent être classés dans l'ordre chronologique, *sauf certains* que l'on souhaite placer en tête de liste dans un ordre précis.

Il y a plusieurs méthodes pour réaliser ce classement. Notez bien que les deux méthodes présentées ci-après sont **valables tant pour ordonner les articles que les autres objets de SPIP** (rubriques, brèves, etc.).

Méthode simple

Rappel : Pour classer les articles selon un ordre imposé, on numérote leurs titres dans l'espace privé, avec un numéro suivi d'un point et d'un espace : « 1. Premier article », « 2. Deuxième article », etc.
Dans les squelettes, pour ne pas afficher ces numéros sur le site public, on passe le filtre **`|supprimer_numero`** sur la balise `#TITRE`, et on utilise le critère **`{par num titre}`** sur la boucle ARTICLES pour ordonner selon les numéros indiqués.

Pour classer selon un ordre imposé ET/OU selon la date, on écrira donc ceci (à l'intérieur d'une boucle RUBRIQUES) :

```
<BOUCLE_articles(ARTICLES) {id_rubrique} {par num titre}{!par date}>
      [(#TITRE|supprimer_numero)]
</BOUCLE_articles>
```

Ainsi les articles de la rubrique seront tout d'abord classés par numéros, puis ceux portant un numéro identique seront classés par date inverse.

Remarque importante : Cela est tout à fait satisfaisant tant que, dans une rubrique donnée, tous les articles sont numérotés ou qu'aucun ne l'est. Par contre, si une rubrique contient des articles non numérotés, sauf un par exemple, celui-ci s'affichera en dernier. En effet, **non numérotés, les articles sont considérés comme ayant zéro pour numéro**. Pour éviter ce classement, on emploie alors la méthode suivante.

Méthode fine

À l'intérieur d'une boucle RUBRIQUES, nous allons effectuer le test suivant : est-ce qu'il existe, dans cette rubrique, au moins un article dont le titre commence par un numéro suivi d'un point ?

```
<BOUCLE_test_numero(ARTICLES) {id_rubrique} {titre==^[0-9]+\.} {0,1}>
      Il existe un article numéroté dans cette rubrique
</BOUCLE_test_numero>
      Il n'y a pas d'article numéroté
<//B_test_numero>
```

Le critère intéressant ici est : **`{titre==^[0-9]+\.}`**

Il s'agit d'une sélection sur le `titre`, selon une *expression régulière* (« `==` » indique une sélection selon une expression régulière) dont la syntaxe est : au début du `titre` (« `^` » indique le début de la chaîne testée), il y a un ou plusieurs (« `+` » indique « au moins un des caractères précédents ») caractères compris entre 0 et 9 (« `[0-9]` » signifie « caractères compris entre 0 et 9 inclus »), suivis du caractère « point » (« `\.` »).

Notez enfin qu'on ne sélectionne qu'un seul article ainsi numéroté (`{0,1}`) ; de cette façon, l'intérieur de la boucle ne sera effectué qu'une seule fois. De plus, il suffit qu'il existe un seul article numéroté pour provoquer l'affichage d'une liste par ordre « numéroté ».

Cette boucle affiche ainsi « Il existe un article numéroté dans cette rubrique » s'il y a au moins un article dont le titre est précédé d'un numéro dans la rubrique, et « Il n'y a pas d'article numéroté » sinon.

Il suffit maintenant d'installer à la place de ces mentions des boucles d'affichage des articles selon l'ordre de présentation désiré :

```
<ul>
<BOUCLE_test_numero(ARTICLES) {id_rubrique} {titre==^[0-9]+\.} {0,1}>
      <BOUCLE_ordre_numeros(ARTICLES) {id_rubrique} {par num titre}>
            <li>[(#TITRE|supprimer_numero)]</li>
      </BOUCLE_ordre_numeros>
</BOUCLE_test_numero>
      <BOUCLE_ordre_date(ARTICLES) {id_rubrique} {par date}{inverse}>
            <li>#TITRE<li>
      </BOUCLE_ordre_date>
<//B_test_numero>
</ul>
```

Ainsi les articles numérotés de la rubrique sont affichés en premier, classés par numéros, suivis de ceux ne portant pas de numéro, classés du plus récent au plus ancien.

Il est possible de faire plus simple :

```
<BOUCLE_Classement(ARTICLES){par num titre, titre}>
<li>[(#TITRE|supprimer_numero)]</li>
</BOUCLE_Classement>
```

Classera les articles par ordre de titre puis par numéro.

Trier par ordre alphabétique, sauf un article qu'il faut afficher en premier

Septembre 2003 — maj : Février 2007

Mettons que vous voulez présenter une série d'articles par ordre alphabétique, sauf le prologue que vous voulez légitimement afficher au début de la liste...

Il existe une astuce pour « duper » le tri effectué par SPIP : il suffit d'ajouter un espace au début du titre de l'article (par exemple, transformer « Prologue » en « Prologue »). Le tri alphabétique pensera alors que ce titre doit « passer avant les autres », et l'affichera en premier. Cependant l'espace supplémentaire du titre sera ignoré par le navigateur Web, et ne provoquera pas de décalage disgracieux dans la mise en page.

Notez bien : l'astuce ne fonctionnera que si vous utilisez un classement alphabétique sur le titre des articles (c'est-à-dire le critère `{par titre}`). Tout autre classement ne fonctionnera pas.

Remarque : cela s'applique également aux rubriques, brèves, sites référencés, etc.

Plusieurs logos pour un article

Août 2003 — maj : Janvier 2007

Il est fréquent, pour rythmer la navigation sur son site, de vouloir utiliser des logos différents (notamment de tailles différentes) pour un même article en fonction de l'endroit où il apparaît.

Par exemple, utiliser un « gros » logo sur la page d'accueil du site qui permette de bien mettre en valeur l'article principal du moment, et un « petit » logo pour la navigation générale du site.

Les webmestres disposent d'une fonction `image_reduire` qui permet de créer différentes versions (de différentes tailles) d'un *même* logo d'article. Mais il faut une autre méthode pour utiliser des logos *différents* pour un même article.

Les webmestres ont créé des méthodes personnelles basées sur l'utilisation différenciée du logo « normal » et du logo « pour survol ». Par exemple : le logo « normal » utilisé comme « petit logo » (appelé par la balise `#LOGO_ARTICLE_NORMAL`), et sur le sommaire, le logo « pour survol » (appelé par la balise `#LOGO_ARTICLE_SURVOL`) pour afficher la « grande » version du logo. Cette méthode complique souvent le code des squelettes, et interdit l'utilisation habituelle des logos « avec survol » que SPIP fournit automatiquement. Elle est de plus d'une souplesse très limitée.

Il est possible de joindre des *documents* aux articles (et, accessoirement, aux rubriques). Nous allons expliquer ci-dessous comment utiliser ces documents joints pour créer plusieurs logos pour un même article.

Principe général

- Nous continuerons à utiliser les deux logos de l'article pour afficher les logos « normaux » (ceux qui apparaissent dans les liens de navigation les plus fréquents, par exemple sur les pages des rubriques), ce qui permet de conserver la simplicité de gestion des logos avec SPIP et la gestion automatique du survol (on revient à l'utilisation évidente de la balise `#LOGO_ARTICLE`, ou de `#LOGO_ARTICLE_RUBRIQUE`).

- Nous déciderons de joindre aux articles un document (généralement une image aux formats GIF, JPEG ou PNG) auquel nous donnerons systématiquement le même nom. Il nous suffira d'afficher ce document (en l'appelant par son nom) à la place du logo « normal » lorsque nous le désirerons.

- Cette méthode permet ainsi de créer autant de logos différents que nécessaire pour un même article (pas seulement un grand logo et un petit logo, mais pourquoi pas une image pixelisée avec un travail typographique élaboré pour afficher le titre, etc.).

- Nous verrons de plus que, grâce aux boucles de SPIP, on pourra très facilement dans les squelettes déterminer si un tel « grand » logo (document portant le nom choisi par nous) est présent, et agir en conséquence (afficher à la place le logo « normal », du texte spécifique, ou carrément un autre élément graphique).

- Miracle de la technologie moderne, des formats propriétaires et de l'accès par haut-débit, de telles versions spécifiques des logos, étant des documents joints, pourront être d'un autre

format que des images. On pourra ainsi afficher, en tant que « grands » logos, des animations Flash ou Shockwave, des animations vidéo...

Mise en place des documents et choix des noms

- Nous décidons (arbitrairement, mais faites selon vos besoins) que les documents joints utilisés en tant que « gros » logo seront tous intitulés « spip_logo » ; ce document « spip_logo » sera affiché sur la page du sommaire de notre site à la place du logo normal.

Nous utiliserons d'autres noms dans la suite de cet exemple pour créer des effets plus fins : décidons immédiatement qu'ils auront tous des noms commençant par « spip_... ». (Cela nous permettra, dans l'affichage habituel des documents joints à un article d'exclure tous les documents dont le nom commence par « spip_... ». De cette façon, l'utilisation de documents en tant que logos alternatifs n'interfèrera pas avec l'affichage, par exemple, d'un portfolio.)

- Sur un article publié en ligne (de façon à pouvoir bidouiller nos squelettes en les testant), nous installons simplement :

- un logo normal ; nous pouvons, si nous le voulons, installer une version de l'image « pour survol » pour la gestion automatique du changement d'image lors du survol avec la souris ;

Le logo « normal »

Le logo est installé classiquement. A priori, il s'agit d'une image de taille modeste.

- un « document joint » (par le pavé « JOINDRE UN DOCUMENT ») ; pour faire simple, installons une image (JPEG, GIF, PNG) ; une fois ce document installé (« uploadé »), nous lui donnons pour titre « spip_logo ». Voilà, c'est la seule manip nécessaire... SPIP affiche ce document en bas de la page de l'article dans l'espace privé, en donnant son titre (« spip_logo ») et en indiquant ses dimensions en pixels.

Le document « spip_logo »

Le seul impératif est de donner à ce document le titre « spip_logo ». Il est inutile d'installer une vignette de prévisualisation.
Dans le cas d'un document multimédia (Flash, Shockwave...), il faut indiquer à la main ses dimensions en pixels.

- Nous décidons de l'usage de ce document intitulé « spip_logo » : il sera affiché sur la page d'accueil du site à la place du logo normal du dernier article publié. De cette façon, la page de Une du site peut afficher une « grande » image pour mettre en valeur l'article en vedette.

Afficher « spip_logo » en Une du site

- Commençons par insérer une boucle toute simple pour afficher le dernier article publié sur le site et son logo « normal ». (Dans tous les exemples qui suivent, le code HTML est réduit à son strict minimum ; à vous d'enrober cela avec la mise-en-pages graphique qui vous convient.)

```
<BOUCLE_article_vedette(ARTICLES){doublons}{par date}{inverse}{0,1}>
      #LOGO_ARTICLE
      <h1>#TITRE</h1>
</BOUCLE_article_vedette>
```

Cette boucle très simple affiche le premier article (`{0,1}`) parmi tous les ARTICLES, sélectionnés par date de publication (`{par date}`) du plus récent au plus ancien (`{inverse}`). On affiche donc bien le dernier article publié sur le site. À l'intérieur de la boucle, on affiche le logo de l'article suivi du titre de l'article.

- Nous avons dit que nous voulions afficher, à la place du logo normal, le document joint à cet article dont le titre est « spip_logo ». Le code devient :

```
<BOUCLE_article_vedette(ARTICLES){doublons}{par date}{inverse}{0,1}>
      <BOUCLE_logo_article_vedette(DOCUMENTS){id_article}{titre=spip_logo}>
            [(#EMBED_DOCUMENT)]
      </BOUCLE_logo_article_vedette>
<h1>#TITRE</h1>
</BOUCLE_article_vedette>
```

La `BOUCLE_logo_article_vedette` sélectionne parmi les documents joints à cet article (`{id_article}`) celui dont le titre est « spip_logo » (`{titre=spip_logo}`). À l'intérieur de la boucle, on demande l'affichage de ce document joint (`#EMBED_DOCUMENT`).

L'usage de `#EMBED_DOCUMENT` dans les squelettes permet d'insérer, via le système de boucles, directement le document à l'intérieur de la page. SPIP se charge de créer le code correspondant à des images ou à des fichiers multimédia.

- Inconvénient : si l'article n'a pas de document joint intitulé « spip_logo », le code précédent n'affiche que le titre. On va donc effectuer une nouvelle modification, qui permet d'afficher le logo « normal » de l'article s'il n'existe pas de document joint pour cet usage. *Notez bien :* une fois cette méthode comprise, il n'y aura plus d'autres subtilités pour réaliser tous les effets suivants...

```
<BOUCLE_article_vedette(ARTICLES){doublons}{par date}{inverse}{0,1}>
      <BOUCLE_logo_article_vedette(DOCUMENTS){id_article}{titre=spip_logo}>
            [(#EMBED_DOCUMENT)]
      </BOUCLE_logo_article_vedette>
            #LOGO_ARTICLE
      <//B_logo_article_vedette>
      <h1>#TITRE</h1>
</BOUCLE_article_vedette>
```

Nous avons tout simplement ajouté l'appel au logo « normal » (`#LOGO_ARTICLE`) en texte alternatif (ce qui se trouve avant `<//B...>` d'une boucle s'affichant si la boucle ne fournit pas de résultat - ici, s'il n'y a pas de document joint à l'article portant le titre « spip_logo »).

Nous avons obtenu le résultat désiré :
- s'il existe un document joint associé à l'article auquel nous avons donné le titre « spip_logo », il est directement affiché ;
- *sinon*, c'est le logo « normal » qui est affiché.

Exclure ces documents spécifiques de l'affichage normal des documents joints

Dans le squelette des articles, on affiche les documents joints grâce à la `BOUCLE_documents_joints`, dont les critères essentiels sont :

```
<BOUCLE_documents_joints(DOCUMENTS){id_article}{mode=document}{doublons}>
```

On appelle les DOCUMENTS liés à cet article (`{id_article}`), qui sont bien des documents joints et non des images (`{mode=document}`) et qu'on n'a pas déjà affichés à l'intérieur du texte de l'article en utilisant le raccourci `<EMBxx>` (`{doublons}`).

Modifions ce code pour interdire l'affichage, dans cette boucle (qui est une sorte de « portfolio »), des documents dont le nom commence par « spip_... » (on ne veut pas afficher ici le « gros » logo utilisé en page de Une du site) :

```
<BOUCLE_documents_joints(DOCUMENTS){id_article}{titre!==^spip\_}
      {mode=document}{doublons}>
```

Le critère `{titre!==^spip\_}` est une expression régulière, dont la syntaxe est très codifiée. On sélectionne les documents dont le titre n'est pas formé ainsi (le `!==` signifie « qui ne correspond pas à l'expression régulière ») : les premiers caractères (le symbole `^` indique le début de la chaîne de caractères) sont « spip » suivi de « _ » (dans la syntaxe des expressions régulières, « _ » indique le caractère « _ », de la même façon que « \. » indique le caractère « . »).

Ce critère sélectionne donc les documents joints dont le titre *ne commence pas* par « spip_ ».

L'exposé du principe général est terminé, vous avez largement de quoi vous amuser avec ça sur votre propre site. Les exemples suivants n'en sont que des variations.

Afficher toujours un gros logo en haut de page

Je décide, toujours de manière arbitraire, qu'il doit toujours y avoir une grosse image en haut de page de mes articles. Il s'agit d'un choix graphique de ma part : pour assurer l'unité graphique de mon site, j'affiche en haut de page une version de grand format liée à l'article (une variation du principe du « grand » logo) et, à défaut, une image stockée ailleurs sur mon site.

- Toujours le même principe : je joins à mon article un document dont je fixe le titre à « spip_haut ». (Pour éviter que ce document ne s'affiche dans le « portfolio » de la `BOUCLE_documents_joints` précédente, je fais commencer son titre par « spip_... ».)

Dans mon squelette des articles, j'affiche simplement en haut de page ce document :

```
<BOUCLE_doc_haut(DOCUMENTS){id_article}{titre=spip_haut}>
      #EMBED_DOCUMENT
</BOUCLE_doc_haut>
```

Comme dans l'exemple précédent, j'affiche le document, lié à l'article de cette page, et dont le titre est « spip_haut ». Fastoche.

- Comme dans le premier exemple, je pourrais décider d'afficher le logo de l'article si ce document n'existe pas :

```
<BOUCLE_doc_haut(DOCUMENTS){id_article}{titre=spip_haut}>

      #EMBED_DOCUMENT
</BOUCLE_doc_haut>
      #LOGO_ARTICLE
<//B_doc_haut>
```

- Mais ça n'est pas le résultat désiré. Je veux, pour des impératifs graphiques, toujours afficher une grande image aux dimensions prédéterminées.

Je vais donc (toujours un choix arbitraire de ma part) créer des images de substitution, utilisées « par défaut », au cas où un article n'aurait pas d'image en propre. Ces images répondent à mes impératifs graphiques (par exemple, elles ont toutes les mêmes dimensions que les documents que j'utilise d'habitude en « spip_haut »).

Sur mon site, je créée une rubrique pour accueillir « en vrac » ces documents de substitution. J'active les documents associés aux rubriques. (Je peux aussi créer un article qui accueillerait tous ces documents, et ainsi ne pas activer les documents joints aux rubriques. Le code serait à peine différent.) Admettons que cette rubrique porte le numéro 18 (c'est SPIP qui va fixer automatiquement le numéro de la rubrique lors de sa création). J'installe tous mes documents de substitution à l'intérieur de cette rubrique numéro 18. (Il est inutile de leur donner un titre.)

Pour appeler, au hasard, un document installé dans cette rubrique, il me suffit d'invoquer les critères suivants :

```
<BOUCLE_doc_substitution(DOCUMENTS){id_rubrique=18}{0,1}{par hasard}>
      #EMBED_DOCUMENT
</BOUCLE_doc_substitution>
```

(Notez bien : le critère `{par hasard}` ne signifie pas que l'image sera différente à chaque visite de la page, mais qu'elle sera sélectionnée au hasard à chaque recalcul de la page.)

On prendra soin, dans la navigation du site, d'interdire l'affichage de la rubrique 18, qui n'a pas besoin d'être présentée aux visiteurs. Le critère `{id_rubrique!=18}` *fera l'affaire.*

- Pour terminer la mise en place du dispositif, il nous suffit d'insérer cette boucle affichant un document de substitution pris au hasard dans la rubrique 18 en tant que texte alternatif à notre `BOUCLE_doc_haut` (à la place du `#LOGO_ARTICLE`) :

```
<BOUCLE_doc_haut(DOCUMENTS){id_article}{titre=spip_haut}>
      #EMBED_DOCUMENT
</BOUCLE_doc_haut>
      <BOUCLE_doc_substitution(DOCUMENTS){id_rubrique=18}{0,1}{par hasard}>
            #EMBED_DOCUMENT
      </BOUCLE_doc_substitution>
<//B_doc_haut>
```

Afficher un titre graphique

Toujours sur le même principe, nous allons afficher une version graphique du titre de l'article. Il s'agit d'une image réalisée avec un logiciel de dessin, où apparaît , avec une typographie particulièrement soignée (effets de relief, de dégradés de couleurs, avec des polices de caractère exotiques...) le titre de l'article.

- Décrétons qu'il s'agira d'un document joint associé à l'article, que nous titrerons « spip_titre ».

- Pour appeler ce document, à l'endroit où doit être affiché le `#TITRE` de l'article, installons la boucle désormais connue :

```
<BOUCLE_doc_titre(DOCUMENTS){id_article}{titre=spip_titre}>
      #EMBED_DOCUMENT
</BOUCLE_doc_titre>
```

Notez à nouveau que cette méthode permet non seulement d'utiliser une image pour afficher le titre, mais aussi une animation Flash, un film... Dans ces cas, il vous faudra indiquer à la main pour votre document joint quelles sont ses dimensions en pixels.

- Complétons le dispositif : s'il n'existe pas de document joint portant le titre « spip_titre », il faut afficher en tant que texte HTML classique les informations nécessaires :

```
<BOUCLE_doc_titre(DOCUMENTS){id_article}{titre=spip_titre}>
      #EMBED_DOCUMENT
</BOUCLE_doc_titre>
      [<div>(#SURTITRE|majuscules)</div>]
      <div><font size=6>#TITRE</font></div>
      [<div>(#SOUSTITRE|majuscules)</div>]
      <br>[(#DATE|nom_jour)] [(#DATE|affdate)]</p>
<//B_doc_titre>
```

*** * ***

Signalons un dernier avantage à cette méthode...

Elle permet de faire par la suite encore évoluer radicalement l'interface graphique de votre site. Sans avoir à supprimer un par un tous les documents intitulés « spip_haut », « spip_titre »..., il vous suffit de créer de nouveaux squelettes qui ne les appellent tout simplement pas.

Par exemple, si les documents « spip_haut » étaient précédemment tous conçus pour une largeur de 450 pixels, et que la nouvelle interface graphique requiert des images d'une largeur de 600 pixels, vous n'aurez pas besoin de modifier un par un tous vos fichiers « spip_haut ». Il vous suffit, dans les squelettes, de ne plus faire appel aux documents intitulés « spip_haut », mais d'utiliser un nouveau nom (par exemple « spip_large ») et d'installer au fur et à mesure vos nouvelles versions de documents en les titrant « spip_large ». Pendant la transition, il n'y aura pas d'incohérences graphiques.

Les plus jetés d'entre vous peuvent même imaginer toutes sortes de tests sur le type de document (`{extension=...}`) ou sur leur taille (`{largeur=...}`, `{hauteur=...}`) (aucun PHP nécessaire) pour réaliser des interfaces selon ces tests (par exemple, une certaine interface graphique si « spip_haut » fait 450 pixels de largeur, et une autre s'il fait 600 pixels de largeur...).

Afficher les derniers articles de vos rédacteurs par rubrique

Mai 2002 — maj : Novembre 2007

Par défaut SPIP vous propose une page auteur qui vous permet de montrer la liste des auteurs/rédacteurs participant à votre site, ainsi que leurs dernières contributions.

Mais un problème vient à se poser quand vous avez plusieurs rédacteurs et que ceux-ci participent activement à votre site. Cela finit par être une page à rallonge.

Cependant il existe un moyen de montrer les dernières contributions de vos auteurs/redacteurs et ce pour chacun d'eux.

Comment procéder ?

Tout d'abord, on va créer un fichier :

myauteur.html

Création du fichier myauteur.html

Dans le fichier mettre le code suivant :

- Juste après la balise `<body>`, mettre

```
<BOUCLE_principale(AUTEURS){id_auteur}{unique}>
```

- Juste avant la balise `</body>`, mettre

```
</BOUCLE_principale>
```

- Dans le corps de la page HTML, voici le code à installer (on ne peut déterminer une rubrique car par défaut l'auteur n'est pas associé à une rubrique mais à un article, le code peut parait biscornu mais on va donc retrouver la rubrique par rapport à l'article) :

Code pour le dernier article

```
<B_appel_article>
      Dernier article écrit par
<BOUCLE_nom_auteur(AUTEURS){id_auteur}>[(#NOM)]</BOUCLE_nom_auteur><br>
<BOUCLE_appel_article(ARTICLES){id_auteur}>
      <BOUCLE_appel_rubrique_article(RUBRIQUES){id_rubrique}{par
titre}{doublons}>
            [(#TITRE|majuscules)]
            <ul>
            <BOUCLE_rappel_article(ARTICLES){id_rubrique}{par
date}{inverse}{doublons}{0,15}>
                <li><a href="#URL_ARTICLE">[(#TITRE)<br></a>]
            </BOUCLE_rappel_article>
            </ul>
      </BOUCLE_appel_rubrique_article>
</BOUCLE_appel_article>
</B_appel_article>
Cette auteur n'a pour l'instant écrit aucun article
<//B_appel_article>
```

Code pour article choisi au hasard

```
<B_appel_article>
Dernier article écrit par
<BOUCLE_nom_auteur(AUTEURS){id_auteur}>[(#NOM)]</BOUCLE_nom_auteur><br>
<BOUCLE_appel_article(ARTICLES){id_auteur}>
      <BOUCLE_appel_rubrique_article(RUBRIQUES){id_rubrique}{par
titre}{doublons}>
            [(#TITRE|majuscules)]
            <ul>
            <BOUCLE_rappel_article(ARTICLES){id_rubrique}{par
hasard}{doublons}{0,15}>
                <li><a
href="#URL_ARTICLE">[(#TITRE)<br></a>]
            </BOUCLE_rappel_article>
            </ul>
      </BOUCLE_appel_rubrique_article>
</BOUCLE_appel_article>
</B_appel_article>
Cette auteur n'a pour l'instant écrit aucun article
<//B_appel_article>
```

Et enfin

Maintenant, il faut configurer votre page auteur (page où vous énumérez vos différents auteurs) pour que, en cliquant sur le lien auteur, celui-ci, dirigera vers la page *myauteur* où sera inscrit les derniers articles écrits par l'auteur.

Le lien devra être écrit de la manière suivante :

```
<a href="#URL_PAGE{myauteur, id_auteur=#ID_AUTEUR}">nom du lien</a>
```

Afficher des éléments par lignes dans un tableau

Avril 2002 — maj : Juillet 2007

Soit à créer un tableau de trois colonnes contenant les titres des articles d'une rubrique, le nombre de lignes dépendant du nombre total d'articles ; sur le principe :

article 1 article 2 article 3
article 4 article 5 article 6
article 7 article 8 article 9

Il faut évidemment utiliser une boucle Articles, mais la difficulté réside dans le placement de balises `tr` ouvrante et fermante tous les trois passages dans la boucle. L'astuce consiste à utiliser la balise `#COMPTEUR_BOUCLE` et le filtre `alterner`. Cette balise indique le nombre de passages déjà effectués dans la boucle : si le reste de sa division par 3 vaut 0 il faut produire une balise `tr` ouvrante au début du code, s'il vaut 2, il faut produire une balise `tr` fermante à la fin.

Pour cela, point n'est besoin d'opérations arithmétiques dans le texte du squelette. Il suffit d'utiliser le filtre `alterner`, qui accepte un nombre quelconque d'arguments, disons *n*, le premier devant être un entier, appelons-le `i`. Ce filtre calcule le reste *k* de la division de *n-1* par *i* et retourne alors son *k*+unième argument. Il suffit donc de lui donner 3 arguments vides sauf un pour ne produire les balises `tr` qu'aux passages appropriés.

Si l'on veut un code HTML irréprochable, il faut produire une séquence de `<td></td>` après la boucle pour compléter la dernière ligne du tableau lorsque le nombre total d'articles n'est pas divisible par 3. Il suffit à nouveau d'utiliser le filtre `alterner`, mais appliqué cette fois à la balise `#TOTAL_BOUCLE`.

Ce qui donne :

```
<B_ligne>
<table>
<BOUCLE_ligne (ARTICLES) {id_rubrique} {par titre}>
[(#COMPTEUR_BOUCLE|alterner{'<tr>','',''})]
      <td width="33%">
      <a href="#URL_ARTICLE">#TITRE</a>
      </td>
[(#COMPTEUR_BOUCLE|alterner{'','','</tr>'})]
</BOUCLE_ligne>
[(#TOTAL_BOUCLE|alterner{'<td></td><td></td></tr>','<td></td></tr>',
 ''})]
</table>
</B_ligne>
```

Pour améliorer la lisibilité, on peut colorer différemment les lignes paires et impaires en mettant un attribut `style` dans la balise `tr` ouvrante. Il faut alors remplacer `'<tr>'` ci-dessus par une nouvelle utilisation de `#COMPTEUR_BOUCLE|alterner`, l'argument d'un filtre pouvant lui-même utiliser des balises et des filtres. Le squelette suivant donnera ainsi le tableau visualisé au début de cet article :

```
<B_ligne>
<table>
<BOUCLE_ligne (ARTICLES) {id_rubrique} {par titre}{0,7}>
[(#COMPTEUR_BOUCLE|
alterner{[(#COMPTEUR_BOUCLE|alterner{'<tr style="background: #eee;">',
                    '<tr style="background: #ddd;">'})],    '','']}]
      <td width="33%">
      <a href="#URL_ARTICLE">#TITRE</a>
      </td>[
(#COMPTEUR_BOUCLE|alterner{'','','</tr>'})
]
</BOUCLE_ligne>
[(#TOTAL_BOUCLE|alterner{'<td></td><td></td></tr>','<td></td></tr>', ''})]
</table>
</B_ligne>
```

Le même type de boucle, en remplaçant l'appel du titre par le logo (avec la balise #LOGO_ARTICLE), permet d'afficher une galerie où chaque logo d'article donne un aperçu (dont la taille sera de préférence fixée afin d'avoir une belle mise en page), et le texte de l'article contient la ou les œuvres exposées.

P.-S.

Pour obtenir ce résultat, la version initiale de cet article utilisait les boucles récursives de manière détournée. Ce détournement ne sera plus possible à terme dans SPIP, et est en outre moins efficace que la méthode ci-dessus qui n'effectue qu'une seule requête au serveur SQL.

Ne pas afficher les articles publiés depuis plus d’un an

Avril 2002 — maj : Janvier 2007

Cela s’effectue avec le critère « *age* », qui est l’âge de l’article (calculé depuis sa date de mise en ligne dans l’espace public) en nombre de jours.

Ainsi pour conserver tous les articles de moins d’un an dans la rubrique courante, nous utiliserons le critère `{age < 365}` :

```
<B_articles_recents>
 <ul>
  <BOUCLE_articles_recents(ARTICLES) {id_rubrique} {age < 365} {par titre}>
      <li>#TITRE</li>
  </BOUCLE_articles_recents>
 </ul>
</B_articles_recents>
```

Pour prendre en compte l’âge vis-à-vis de la date de première publication au lieu de la date de mise en ligne, il faudra utiliser le critère « *age_redac* » au lieu de « *age* ». L’âge est indiqué en nombre de jours. Voir : « La gestion des dates ».

Notons que cette manipulation est possible avec tous les types de données auxquels est associée une date (brèves, forums...).

Présenter les résultats d’une recherche par secteurs

Avril 2002 — maj : Septembre 2003

Il suffit d’inclure la boucle de recherche dans une boucle de type rubriques sélectionnant les rubriques de premier niveau ; dans la boucle de recherche, on ajoute alors le critère « *id_secteur* » pour se limiter au secteur courant.

```
<BOUCLE_secteurs(RUBRIQUES){racine}>

<B_recherche>
      <b>#TITRE</b>
      <ul>
<BOUCLE_recherche(ARTICLES){recherche}{id_secteur}
            {par points}{inverse}{0,5}>
      <li><a href="#URL_ARTICLE">#TITRE </a>
</BOUCLE_recherche>
      </ul><hr>
</B_recherche>

</BOUCLE_secteurs>
```

On remarquera que le titre du secteur n’est affiché que si la recherche a donné des résultats pour ce secteur. D’autre part, pour chaque secteur on n’affiche que les cinq articles les mieux classés, par ordre décroissant de pertinence.

Attention cependant, comme la recherche est effectuée autant de fois qu’il y a de secteurs, le calcul risque d’être ralenti.

Afficher le nombre de messages répondant à un article

Avril 2002 — maj : Février 2007

Pour afficher le nombre de messages du forum lié à un article, nous devons créer une boucle de type FORUMS, liée à un article, de façon à compter son nombre de résultats.

À première vue, il est très simple de connaître le nombre d'éléments d'une boucle : il suffit d'utiliser la balise : **`#TOTAL_BOUCLE`**. Mais son usage est un poil acrobatique.

Bien sélectionner *tous* les messages

Première subtilité : nous voulons *tous* les messages des forums liés à l'article, en comptant les réponses aux messages, les réponses aux réponses...

Une simple boucle de type : `<BOUCLE_forum(FORUMS) {id_article}>` sélectionne uniquement les messages qui répondent directement à l'article. Pour accéder aux réponses à ces messages, on inclut habituellement une seconde boucle à l'intérieur de celle-ci, mais ce n'est pas ce que nous souhaitons faire ici.
Pour pouvoir tous les compter, nous voulons que la boucle sélectionne absolument tous les messages du forum lié à l'article, sans tenir compte de leur hiérarchie. Pour cela, il faut spécifier le critère **`{plat}`**, qui comme son nom l'indique sert à afficher un forum à plat. Ce qui donne : `<BOUCLE_forum(FORUMS) {id_article} {plat}>`

N'afficher *que* le total

Voyons maintenant comment compter les éléments qu'elle contient. La difficulté, ici, c'est que justement cette boucle ne doit rien afficher ! Elle n'affiche ni les messages ni leur titre, et on évitera même de lui faire afficher des espaces ou des retours à la ligne (sinon votre code HTML final contiendra des dizaines de lignes vides, inélégantes), en ne laissant pas même un espace entre les deux tags ouvrants et fermants de la boucle : `<BOUCLE_forum(FORUMS) {id_article} {plat}></BOUCLE_forum>`.

L'intérieur de la boucle n'affiche donc rigoureusement rien, mais on doit afficher, après la boucle, le nombre de résultats. La balise `#TOTAL_BOUCLE` peut s'utiliser non seulement à l'intérieur de la boucle, mais aussi (c'est la seule dans ce cas) dans le *texte conditionnel après* (le texte qui s'affiche après la boucle si elle contient des éléments) et le *texte conditionnel alternatif* (le texte qui s'affiche si la boucle est vide).

Une subtilité à bien comprendre : le texte conditionnel alternatif s'affiche *si la boucle ne trouve aucune réponse* ; il est donc affiché même si la boucle sélectionne des éléments (ici des messages de forum) mais qu'elle ne contient aucun affichage.

Nous devons donc placer `#TOTAL_BOUCLE` dans le texte conditionnel alternatif. S'il n'y aucun message de forum attaché à l'article, `#TOTAL_BOUCLE` sera vide, il ne faut donc pas afficher le texte englobant (« il y a N message(s) de forum ») dans ce cas.

```
<BOUCLE_combien(FORUMS) {id_article} {plat}>
</BOUCLE_combien>
[Il y a (#TOTAL_BOUCLE) message(s) de forum.]
<//B_combien>
```

Un message, des messages...

Nous affichons désormais le nombre de messages, s'il en existe. Mais nous voulons faire mieux : nous souhaitons que le terme « message » soit automatiquement accordé au singulier lorsque la boucle ne trouve qu'un seul résultat, et accordé au pluriel lorsqu'elle en trouve plusieurs.

Pour cela, nous allons utiliser les filtres de test `|=={...}` et `|?{...,...}` pour accorder « message » selon la valeur de `#TOTAL_BOUCLE` :
`[(#TOTAL_BOUCLE|=={1}|?{message,messages})]`

Ce qui nous donne donc :

```
<BOUCLE_combien(FORUMS) {id_article} {plat}></BOUCLE_combien>
[Il y a (#TOTAL_BOUCLE) [(#TOTAL_BOUCLE|=={1}|?{message,messages})] de
forum.]
<//B_combien>
```

Un menu déroulant pour présenter une liste d’articles

Avril 2002 — maj : Novembre 2007

On souhaite réaliser un menu déroulant en utilisant les commandes HTML adaptées à la création de formulaire ; de plus on veut que ce menu serve à aller à l’URL de l’article sélectionné. Si l’URL des articles est du type `#URL_PAGE{article, id_article=123}`, le bout de code suivant conviendra :

```
<FORM ACTION="#URL_PAGE{article}" METHOD="get">

      <SELECT NAME="id_article">
            <BOUCLE_menu_articles(ARTICLES) {id_rubrique} {par titre}>
            <OPTION VALUE="#ID_ARTICLE">#TITRE</OPTION>
            </BOUCLE_menu_articles>
      </SELECT>

      <INPUT TYPE="submit" NAME="Valider" VALUE="Afficher l'article">
</FORM>
```

Les critères de la boucle articles (ici : les articles de la rubrique courante, triés par titre) seront modifiés selon vos besoins. Ce type de construction marche bien sûr aussi pour les brèves, rubriques...

Selon le même principe, il est tout aussi facile de présenter une liste de rubriques, de brèves... ou même l’intégralité de la structure du site.

Remplir les meta-tags HTML des pages d'article

Avril 2002 — maj : Mai 2009

Le but de cet exemple est d'installer dans les méta-tags de notre page, la liste des mots-clés associés à l'article ainsi que le nom des auteurs.

Si l'on veut optimiser le référencement du site par les moteurs de recherche, on peut utiliser des balises HTML spéciales, appelées « Métas NAME », situées dans l'en-tête du document HTML. Avec SPIP, vous pouvez remplir ces balises de façon automatique, en utilisant par exemple le descriptif de l'article, les mots-clés associés, ainsi que le nom du ou des auteurs. Voici comment faire.

Exemples de Métas NAME remplis par SPIP

Rôle	Syntaxe HTML/SPIP
Titre de la page	`<title>[(#NOM_SITE_SPIP|textebrut)]</title>`
Description	`[<meta name="description" content="(#INTRODUCTION|couper{200}|attribut_html)" />]`

Mots-clés	`<B_keywords>`
	`<meta name="keywords" content="<BOUCLE_keywords(MOTS) {id_article} {","}>[(#TITRE|attribut_html)]</BOUCLE_keywords>" />`
	`</B_keywords>`

Auteurs	`[<meta name="author" content="(#LESAUTEURS|attribut_html)" />]`
Nom du logiciel	`<meta name="generator" content="SPIP[ (#SPIP_VERSION)]" />`
Courriel du webmestre	`[<meta name="reply-to" content="(#EMAIL_WEBMASTER|attribut_html)" />]`

N'oubliez pas de passer les filtres **`textebrut`** et **`attribut_html`** sur les balises SPIP pour supprimer les tags, paragraphes et espaces insécables qui n'ont rien à faire ici. Limitez également le nombre de caractères, conformément aux limitations propres à chaque meta-tags, avec le filtre **`couper`**.

On remarquera que pour les mots-clés on utilise une boucle imbriquée pour aller chercher ces informations à partir de l'*id_article* courant. De plus, on spécifie une virgule comme séparateur afin que le contenu du meta-tag soit compréhensible (y compris par un moteur de recherche).

Ces meta-tags ne sont pas indispensables pour assurer le bon référencement d'un site. Par contre, n'oubliez pas le titre de la page, qui reste important, notamment parce qu'il permet d'identifier clairement chaque page de votre site dans les résultats de recherche.

Voici donc l'exemple complet pour le squelette `article.html` (à placer dans une boucle ARTICLES, entre les balises `head` de la page) :

```
<head>
<title>[(#TITRE|textebrut) - ][(#NOM_SITE_SPIP|textebrut)]</title>
[<meta name="description" content="(#INTRODUCTION|
      sinon{#DESCRIPTIF_SITE_SPIP}|couper{200}|attribut_html)" />]
<B_keywords><meta name="keywords"
      content="<BOUCLE_keywords(MOTS) {id_article}
      {","}>[(#TITRE|attribut_html)]</BOUCLE_keywords>" />
</B_keywords>
[<meta name="author" content="(#LESAUTEURS|attribut_html)" />]
[<meta name="generator" content="SPIP[ (#SPIP_VERSION)]" />]
[<meta name="reply-to" content="(#EMAIL_WEBMASTER|attribut_html)" />]
</head>
```

Vous pouvez adapter cet exemple à chaque type d'élément : rubriques, brèves, etc.