# Introduction

Ce document a pour objectif de présenter la conception « technique » des modèles de pages pour le logiciel de publication collaborative en ligne SPIP et de servir de « mémo » pour son utilisation.
La rédaction avec SPIP et, au minimum, les bases de HTML sont ici considérées comme acquises.

# Principes de fonctionnement

Spip fonctionne avec des **« objets de contenus » : différents types de textes** associés éventuellement à d'autres média.
**Ces types correspondent à des modèles de page, les « SQUELETTES »**, qui permettent de concevoir la structure du site, son arborescence, sa navigation.
Les squelettes ne peuvent donc être créés qu'après la mise à plat des objectifs et besoins du site.
**L'objet « élémentaire » dans Spip est l'« ARTICLE ». Il est obligatoirement classé dans une « RUBRIQUE ».** On retrouve ainsi des squelettes de page « article », « rubrique », etc. autant que de type de textes. Il est aussi possible de créer des squelettes composites ou propres au site. La page d'accueil d'un site Spip est l'exemple d'un squelette composite : elle est un « puzzle » d'objets choisis selon une ligne éditoriale.

Pour « alléger le trafic » entre la base de données et le site public, Spip fonctionne avec un **système de cache** :
pour chaque page du site public, **Spip génère un fichier temporaire** qui est le **résultat final entre les contenus à afficher et la structure HTML de la page**. La structure est le squelette qui appèle en parallèle la mise en forme en CSS.
Ces trois éléments constituent une page Web complète : l'esprit (les contenus), le corps (le squelette) et l'habillage (la mise en forme).

# Le squelette a son propre langage

**Chaque « objet » de Spip est composé de plusieurs champs**. Dans un squelette, **ces champs sont appelés par des « BALISES »**. L'un de ces champs est l'identifiant de l'objet : tout nouvel élément créé est automatiquement numéroté et unique.

Pour appeler la plupart de ces balises, il est nécessaire d'utiliser un système de « BOUCLE ». **La boucle permet de définir le type d'objet recherché** et de répéter pour chaque élément trouvé une ou plusieurs actions.

Par exemple, pour un ou plusieurs articles, afficher le champ titre, le champ descriptif et un lien vers la page de l'article complet.

## Syntaxe

- ➔ Les **BALISES** sont des éléments « simples » et s'écrivent en majuscule, précédées d'un dièse. Par exemple : **#TITRE**
- ➔ Les **BOUCLES** sont des éléments « composés » puisqu'il faut d'abord cibler le type d'objet à afficher puis donner les instructions à réaliser.
  Le mot **BOUCLE**, en majuscule, est obligatoire, suivi d'un ***underscore*** puis d'un **nom unique** pour la boucle. On définit ensuite **entre parenthèses le type d'objet** appelé. **La boucle s'écrit comme une balise HTML fermante** : les instructions de la boucle se trouvent à l'intérieur et ne seront écrites dans le fichier cache que si la boucle trouve au moins un résultat.

```
<BOUCLE_nom_de_la_boucle(ARTICLES)>
     <structure_html> #TITRE </structure_html>
</BOUCLE_nom_de_la_boucle>
```

Cette boucle affichera tous les titres (**#TITRE**) des objets de type article (**ARTICLES**) du site.

- ➔ **Les CRITÈRES de sélection restreignent et/ou trient les résultats d'une boucle**. Pour chaque type d'objet, il existe une boucle et pour chacune, des critères. Certains d'entre eux sont communs à tous les types de boucle. Ces critères sont ajoutés à la boucle, encadrés d'accolades. Il peuvent être « simples » ou composés d'attributs ou de valeurs.

Par exemple, pour n'afficher que les articles d'une seule rubrique, le critère **id_rubrique** (identifiant de la rubrique) bloquera la boucle sur la seule rubrique définie :

```
<BOUCLE_nom_de_la_boucle(ARTICLES) {id_rubrique=2}>
     <structure_html> #TITRE </structure_html>
</BOUCLE_nom_de_la_boucle>
```

Ici, seront affichés tous les titres (**#TITRE**) des objets de type article (**ARTICLES**) enregistrés dans la rubrique numéro 2 (**{id_rubrique=2}**).

Pour organiser plus finement l'affichage, il est possible d'associer plusieurs critères à une boucle. Pour trier par exemple par date ou ordre alphabétique, ou restreinte le nombre de résultats :

```
<BOUCLE_nom_de_la_boucle(ARTICLES) {id_rubrique=2} {par date} {0,5}>
     <structure_html> #TITRE </structure_html>
</BOUCLE_nom_de_la_boucle>
```

Ici, seront affichés les 5 premiers (`{0,5}`) titres d'article trouvés dans la rubrique n° 2, classés par date croissante (`{par date}`).
Le premier chiffre du critère `{0,5}` indique la position dans la liste du premier élément à afficher et le second chiffre, le nombre d'éléments à retourner.

## Boucles et structure HTML

On ne sait pas forcément si une boucle retournera des résultats mais pour réaliser des pages à la structure HTML correcte, il est souvent nécessaire de savoir s'il faut afficher ou non une liste, par exemple.
Si une liste d'articles est demandée, la structure HTML sera :

```
<ul>
     <li> Titre 1 </li>
     <li> Titre 2 </li> ...
<ul>
```

L'action à répéter pour la boucle est d'écrire les éléments `<li>`. On écrira donc :

```
<ul>
<BOUCLE_nom_de_la_boucle(ARTICLES) {id_rubrique=2} {par date} {0, 5}>
     <li> #TITRE </li>
</BOUCLE_nom_de_la_boucle>
</ul>
```

Si aucun résultat n'est trouvé, la page contiendra un couple `<ul></ul>` vide, ce qui posera problème pour la mise en forme.

Pour éviter cela, la syntaxe des boucles de Spip permet une écriture optionnelle. Il suffit d'ajouter en amont et en aval de la boucle une « sur-boucle » qui reprend le nom de la boucle. **Tout ce qui se trouve entre la « sur-boucle » et la boucle ne sera crée dans la page du cache que si la boucle retourne au moins un résultat.**

```
<B_nom_de_la_boucle>
<ul>
     <BOUCLE_nom_de_la_boucle(ARTICLES) {id_rubrique=2} {par date} {0, 5}>
     <li> #TITRE </li>
     </BOUCLE_nom_de_la_boucle>
</ul>
</B_nom_de_la_boucle>
```

Lorsqu'on souhaite **afficher une information si la boucle ne retourne rien** :

```
<B_nom_de_la_boucle>
<ul>
     <BOUCLE_nom_de_la_boucle(ARTICLES) {id_rubrique=2} {par date} {0, 5}>
     <li> #TITRE </li>
     </BOUCLE_nom_de_la_boucle>
</ul>
</B_nom_de_la_boucle>


Aucun résultat
<//B_nom_de_la_boucle>
```

Il est aussi possible d'**imbriquer les boucles** entre elles. Ainsi, pour une rubrique, on peut afficher ses propres champs et ceux des articles qu'elle contient :

```
<B_rubrique2>
<ul>
     <BOUCLE_rubrique2(RUBRIQUES) {id_rubrique=2}>
     <li> #TITRE <!-- (de la rubrique) -->

          <B_articles>
          <ul>
               <BOUCLE_articles(ARTICLES) {id_rubrique}
{par date} {inverse}>
               <li> #TITRE <!-- (de l'article) --> </li>
               </BOUCLE_articles>
          </ul>
          </B_articles>

     </li>
     </BOUCLE_rubrique2>
</ul>
</B_rubrique2>
```

Ici, le critère `id_rubrique` de la boucle « articles » est simple (sans valeur) car cette boucle est incluse dans une autre et il prendra automatiquement la valeur de la rubrique définie ou trouvée par la boucle supérieure.

Cet exemple affichera une rubrique et ses articles propres seulement. Si l'on souhaite afficher aussi ses sous-rubriques, il faudra ajouter à l'intérieur de la boucle principale, une autre boucle de type RUBRIQUES :

```
<B_rubrique2>
<ul>
     <BOUCLE_rubrique2(RUBRIQUES) {id_rubrique=2}>
     <li> #TITRE <!-- (de la rubrique) -->
          <B_articles> ... </B_articles>
     </li>

          <BOUCLE_sous_rubrique(RUBRIQUES){id_parent}>
          <li> ... </li>
          </BOUCLE_sous_rubrique>

     </BOUCLE_rubrique2>
</ul>
</B_rubrique2>
```

Pour pouvoir afficher toutes les rubriques et sous-rubriques, quelque soit la profondeur de hiérarchisation, il est possible d'utiliser la **récursivité** :

```
<B_rubrique>
<ul>
     <BOUCLE_rubrique(RUBRIQUES) {racine}>
          <li> ...
          <B_sous_rubrique>
          <ul>
               <BOUCLE_sous_rubrique(RUBRIQUES)
{id_parent}>
               <li> ...


               <BOUCLE_recursive(BOUCLE_sous_rubrique)>
               </BOUCLE_recursive>

               </li>
               </BOUCLE_sous_rubrique>
       </ul>
          </B_sous_rubrique>
          </li>
     </BOUCLE_rubrique2>
</ul>
</B_rubrique2>
```

## Les filtres

Les contenus récupérés de la base et compilés par Spip peuvent être traités dans les squelettes.
Ce traitement est géré par des `FILTRES`. Il sont **associés à la balise**, séparés par un « pipe » et encadrés de crochets puis de parenthèses :

```
[(#TITRE|supprimer_tags)]
```

Ce qui se trouve entre les crochets et les parenthèses ne sera affiché que si le filtre retourne quelque chose ou si la balise n'est pas « vide ».

Ainsi, si le champ optionnel d'un article « Sous-titre » (appelé par la balise **#SOUSTITRE**) est vide, il est possible de gérer la structure HTML de la page :

```
[<p>(#SOUSTITRE)</p>]
```

ou

```
[<p>(#SOUSTITRE|supprimer_tags)</p>]
```

Ces deux exemples précédents n'écriront le paragraphe que si le sous-titre de l'article est renseigné.

Il est possible de combiner les filtres :

```
[<p>(#SOUSTITRE|supprimer_tags|couper{40})</p>]
```

Certains filtres nécessitent ou acceptent des **arguments**. Ceux-ci sont **définis entre accolades** tout de suite après le nom du filtre, sans espace. Si plusieurs arguments sont autorisés, ils sont **séparés par des virgules**.
Dans l'exemple ci-dessus, le filtre **couper** retourne le texte de la balise réduit au nombre de caractères défini par son argument (**{40}**).

## *Affichage conditionnel*

À l'intérieur ou hors d'une boucle, quelques moyens d'afficher des informations de manière conditionnelle existent, grâce aux **filtres de comparaison et de test**.
Par contre, **une boucle ne peut pas être incluse dans un filtre conditionnel**.
La syntaxe est toujours la même que pour appliquer un filtre puisqu'il s'agit de tester une balise :

```
[(#ID_SECTEUR|=={4}|?{oui})]
```

ou

```
[(#ID_SECTEUR|=={4}|?{oui, non})]
```

Ici, l'**id_secteur** (identifiant de la rubrique principale, à la « racine » du site) est testé. S'il vaut la valeur indiquée dans le filtre de comparaison (**=={n}**), le filtre de test (**?{oui, non}**) retourne les données indiquées par **oui**, sinon retourne **non**.

# INCLURE des squelettes

Il est parfois utile de créer des « bout de page » que l'on peut réutiliser dans plusieurs squelettes. Un en-tête ou un pied de page par exemple.
La balise (à la syntaxe particulière) **<INCLURE>** permet... d'inclure des éléments réutilisables.
Le fichier à inclure est appelé par le paramètre entre accolades « **fond** » et **nommé sans l'extension : <INCLURE{fond=fichier}>**.

Pour prendre en compte le contexte du squelette contenant, il est possible d'ajouter des paramètres à cette balise.

Dans une boucle RUBRIQUES du squelette contenant par exemple, il suffit d'ajouter le paramètre **id_rubrique** :

```
<BOUCLE_rubrique(RUBRIQUES)>
     <INCLURE{fond=mots_cles} {id_rubrique}>
</BOUCLE_rubrique>
```

Ce paramètre est récupéré et/ou traité dans le fichier appelé par **INCLURE**, **mots_cles.html** :

```
<BOUCLE_mots_rubrique(MOTS){id_rubrique} {" - "}>
     #TITRE
</BOUCLE_mots_rubrique>
```

## Variables personnalisées

Il est possible grâce aux variables d'utiliser plusieurs fois dans un squelette des éléments ou de les traiter différemment.

- ➔ On récupère une variable depuis l'URL ou depuis un **INCLURE** avec la balise **#ENV{nom_var[, valeur par défaut]}**
- ➔ On crée une « variable Spip » avec la balise : **#SET{nom_var, valeur}**.
- ➔ On récupère la valeur d'une « variable Spip » avec la balise **#GET{nom_variable}**.

Par exemple, dans un squelette rubrique.html, pour afficher le titre de la rubrique dans une boucle ARTICLES.
1. dans la boucle RUBRIQUES :

```
#SET{titre_rub, #TITRE}
```

2. et dans la boucle ARTICLES :

```
#GET{titre_rub}
```

Cette variable **#GET** peut être récupérée ailleurs dans le même squelette (hors ou dans une boucle) mais pas directement dans les squelettes inclus.

Pour récupérer cette variable dans un squelette inclus :

```
<INCLURE{fond=mots_cles}{id_rubrique}
{titre_rub=#GET{titre_rub}}>
```

Et dans mots_cles.html :

```
<B_mots_rubrique>
<p>#ENV{titre_rub} : </p>
<p>
     <BOUCLE_mots_rubrique(MOTS){id_rubrique} {" - "}>
          <a href="#URL_MOT">#TITRE</a>
     </BOUCLE_mots_rubrique>
</p>
</B_mots_rubrique>
```

# Mémo

Liste des boucles, de balises, critères et filtres principaux. Liste non exhaustive. Le glossaire de Spip reprend tous les éléments de syntaxe : http://www.spip.net/@?lang=fr

## Les types de boucles

- ➔ **BOUCLE(ARTICLES)** → Retourne un / des articles.
- ➔ **BOUCLE(AUTEURS)** → Retourne un / des auteurs.
- ➔ **BOUCLE(BREVES)** → Retourne une / des brèves. Les brèves ne peuvent être créées que dans une rubrique « racine ».
- ➔ **BOUCLE(DOCUMENTS)** → Retourne un / des documents
- ➔ **BOUCLE(FORUMS)** → Retourne les messages d'un forum.
- ➔ **BOUCLE(MOTS)** → Retourne un / des mots-clés.
- ➔ **BOUCLE(GROUPES_MOTS)** → Retourne un / des groupes de mot-clés.
- ➔ **BOUCLE(HIERARCHIE)** → Retourne la liste des rubriques depuis la racine jusqu'à la page en cours.
- ➔ **BOUCLE(RUBRIQUES)** → Retourne une / des rubriques
- ➔ **BOUCLE(SIGNATURES)** → Retourne une / des signatures de pétition.
- ➔ **BOUCLE(SITES)** → Remplace l'ancienne boucle SYNDICATION. Retourne un / des sites référencés ou syndiqués dans une / des rubriques.
- ➔ **BOUCLE(SYNDIC_ARTICLES)** → Retourne une liste d'articles d'un / des sites syndiqués.

## Des balises...

### *...propres aux articles*

- Champs de saisie (modifiables par les auteurs) :

➔ **#TITRE**

➔ **#SOUSTITRE**

➔ **#SURTITRE**

➔ **#DESCRIPTIF**

➔ **#CHAPO** → Chapeau de l'article

➔ **#TEXTE**

➔ **#DATE** → Date de publication de l'article. Éditable après la publication de l'article.

➔ **#LOGO_ARTICLE**

➔ **#PS** → Post-scriptum.

- Automatiques :

➔ **#ID_ARTICLE** → Identifiant unique de l'article

➔ **#ID_RUBRIQUE** → Identifiant unique de la rubrique contenant l'article

➔ **#ID_SECTEUR** → Identifiant unique de la rubrique « racine » contenante

- Calculées par Spip :

➔ **#LESAUTEURS** → Tous les auteurs associés à un article

➔ **#INTRODUCTION** → Retourne le descriptif s'il existe, sinon les 600 premiers caractères du chapeau et du texte de l'article

➔ **#NOTES** → Notes générées depuis les raccourcis typographiques dans le texte de l'article.

➔ **#URL_ARTICLE** → « href » qui pointe vers l'article

### *...propres aux rubriques*

- Champs de saisie :

➔ **#TITRE**

➔ **#DESCRIPTIF**

➔ **#TEXTE**

➔ **#LOGO_RUBRIQUE**

- Automatiques :

➔ **#ID_RUBRIQUE**

➔ **#ID_PARENT** → Identifiant de la rubrique contenante. S'il n'y en a pas, vaut zéro. La rubrique est alors considérée comme rubrique « secteur » (à la racine).

➔ **#ID_SECTEUR**

➔ **#URL_RUBRIQUE**

➔ **#DATE** → Date de la dernière modification sur un des éléments contenus dans la rubrique.

- Calculées par Spip :

➔ **#INTRODUCTION**

➔ **#NOTES**

## *...propres au site SPIP*

- Champs de configuration du site :

➔ **#DESCRIPTIF_SITE_SPIP**

➔ **#EMAIL_WEBMASTER**

➔ **#LOGO_SITE_SPIP**

➔ **#NOM_SITE_SPIP**

➔ **#URL_SITE_SPIP**

- Calculées par Spip :

➔ **#CACHE{durée}** → Durée de vie en secondes du fichier temporaire généré par Spip.

➔ **#CHEMIN{fichier}** → Retourne l'adresse du fichier, qu'il soit à la racine ou dans le dossier « squelettes ». Si le fichier est dans un dossier ou sous-dossier non connu de Spip, ajouter le nom de ce dossier, en encadrant la balise des caractères de filtrage :
`[(#CHEMIN{images/toto.png})]` pour le dossier « images » à la racine ou dans « squelettes ».

➔ **#INSERT_HEAD** → Ajoute des éléments d'en-tête dans la page, utiles à Spip et certains plugins. À placer entre `<head>` et `</head>`

➔ **#LANG** → Langue principale de site (peut être utilisée dans les boucles ARTICLES, RUBRIQUES, BREVES et AUTEURS).

➔ **#URL_PAGE** → Dans le cas d'un squelette personnel, permet de générer son URL : `#URL_PAGE{nom_squelette}`. Pour passer des variables à cette URL les ajouter après le nom du squelette, séparé par une virgule :
`#URL_PAGE{nom_squelette, id_secteur=#ID_SECTEUR}`

➔ **#REM** → Permet d'insérer un commentaire dans le fichier squelette :
`[(#REM) commentaire non généré dans le fichier du cache ]`

➔ **#MODELE{modele}** → modele.html est un fichier placé dans le sous-dossier « modeles » dans « squelettes ». Peut être appelé comme raccourcis typographique sous la forme :
`<modeleXX>` (XX est l'identifiant de l'objet à traiter)
ou `<modeleXX|parametre1|parametre2=YY>`
ou `<modele|parametre1=ZZ>` (dans les cas où il n'y a pas « d'objet Spip » à traiter)

➔ **#ENV{variable, défaut}** → Retourne une variable passée dans l'URL ou dans les paramètres d'une balise `#MODELE` ou d'un `INCLURE`.
`défaut` est optionnel. Il est appliqué à `variable` si aucune valeur n'est trouvée.

➔ **#SET{nom_var, valeur}** → Crée une variable.

➔ **#GET{nom_var}** → Récupère la variable crée par `#SET`. Ne fonctionne que dans le contexte d'un squelette et n'est pas répercutée dans un `#MODELE` ou un `INCLURE`

➔ **#FORMULAIRE_RECHERCHE**

➔ **#RECHERCHE**

## *...propres aux auteurs*

- Champs de saisie

➔ **#EMAIL** → Email de l'auteur. Est aussi utilisée pour les SIGNATURES et FORUMS.

- Automatiques :

➔ **#ID_AUTEUR**

- Calculées par Spip :

➔ **#FORMULAIRE_ECRIRE_AUTEUR** → Peut être aussi utilisée dans la boucle ARTICLES. Auquel cas, le message est envoyé à tous les auteurs de l'article.

➔ **#URL_AUTEUR**

### *...propres aux documents*

- **#FICHIER** → Affiche l'URL relative du fichier
- **#ID_DOCUMENT**
- **#LOGO_DOCUMENT** → Vignette selon le type de fichier par défaut, sinon la vignette personnalisée par l'auteur
- **#TYPE_DOCUMENT**
- **#URL_DOCUMENT**

### *...d'autres boucles*

- **#ID_BREVE** → Boucle BREVES
- **#URL_BREVE** → Boucle BREVES
- **#LOGO_BREVE** → Boucle BREVES
- **#ID_MOT** → Boucle MOTS
- **#TYPE**→ Boucle MOTS
- **#URL_MOT** → Boucle MOTS
- **#LOGO_MOT** → Boucle MOTS
- **#ID_GROUPE** → Boucles MOTS et GROUPES_MOTS
- **#ID_SYNDIC** → Boucles SITES, FORUMS et SYNDIC_ARTICLES
- **#LOGO_SITE** → Boucle SITES
- **#NOM_SITE et #URL_SITE** → Boucles ARTICES, BREVES, AUTEURS, SITES, FORUMS, SYNDIC_ARTICLES et SIGNATURES
- **#ID_SYNDIC_ARTICLE** → Boucle SYNDIC_ARTICLES
- **#URL_SYNDIC** → Boucle SYNDIC_ARTICLES
- **#NOM** → Boucles AUTEURS. FORUMS et SIGNATURES
- **#ID_FORUM** → Boucle FORUMS
- **#URL_FORUM** → Boucle FORUMS

### *...communes à toutes les boucles*

- **#PAGINATION** → Pagine une liste d'éléments d'une boucle. S'utilise avec le critère {pagination} dans la boucle et se place dans les parties optionnelles de la boucle (entre `<B_boucle>` et `<BOUCLE_boucle(...) ...>` et/ou `</BOUCLE_boucle>` et `</B_boucle>`).
- **#ANCRE_PAGINATION** → Crée une ancre pour la pagination
- **#COMPTEUR_BOUCLE** → Affiche le numéro de passage de la boucle. Commence par 1.
- **#TOTAL_BOUCLE** → Total affiché par la boucle
- **#GRAND_TOTAL** → Utilisé avec une boucle paginée : retourne le nombre total de résultats.
- **#EXPOSE** → Vérifie si l'objet est celui en cours. Si oui, affiche par défaut « on » (pour `[ class="(#EXPOSE)"]`). Cette balise peut aussi être utilisée comme test : `[(#EXPOSE{oui, non})]`

## Des critères

- **{age < = >. ..}** → Toutes les boucles.
  Age en jours, de l'objet depuis sa date de publication.
- **{date}** → Boucle ARTICLES.
- **{doublons}** ou **{unique}** → Toutes les boucles.
  Évite de ré-afficher un « objet » déjà retourné par une autre boucle.
- **{exclus}** → Toutes les boucles.
  Ne retourne pas l'objet en cours dans une boucle.
- **{id_article}** → Boucles ARTICLES, DOCUMENTS, MOTS, AUTEURS, FORUMS, HIERARCHIE et SIGNATURES.
  Sélectionne le/s objets rattachés à l'article dont l'identifiant est `id_article`
- **{id_auteur}** → Boucles ARTICLES et AUTEURS.
- **{id_breve}** → Boucles BREVES, DOCUMENTS, MOTS et FORUMS
- **{id_document}** → Boucle DOCUMENTS
- **{id_enfant}** → Boucles RUBRIQUES et FORUMS.
  Sélectionne la rubrique contenante ou le message auquel se rattache le message en cours.
- **{id_forum}** → Boucles MOTS et FORUMS
- **{id_groupe}** → Boucles ARTICLES, RUBRIQUES, MOTS, SITES, FORUMS et BREVES.
  Sélectionne le/s objets associés à au moins un mot-clé du groupe `id_groupe`

➔ **{id_mot}** → Boucles ARTICLES, RUBRIQUES, MOTS, SITES, FORUMS et BREVES.
Sélectionne le/s objets associés au mot-clé dont l'identifiant est `id_mot`

➔ **{id_parent}** → Boucles RUBRIQUES et FORUMS.
Sélectionne les rubriques contenues dans la rubrique ou les messages dépendants d'un autre message.

➔ **{id_rubrique}** → Boucles ARTICLES, RUBRIQUES, DOCUMENTS, MOTS, SITES, FORUMS, HIERARCHIE, BREVES et SYNDIC_ARTICLES.
Sélectionne les objets contenus directement dans la rubrique.

➔ **{id_secteur}** → Boucles ARTICLES, RUBRIQUES, SITES, FORUMS et SYNDIC_ARTICLES.
Sélectionne les objets contenus dans la rubrique racine.

➔ **{inverse}** → Toutes les boucles. S'utilise avec un critère de tri : change le sens du tri.

➔ **{lang}** → Boucles ARTICLES, RUBRIQUES, AUTEURS et BREVES.
Sélectionne les objets dont la langue est renseigné dans l'URL de la page ou celle spécifiée si le critère est écrit : `{lang=XX}`

➔ **{meme_parent}** → Boucles RUBRIQUES et FORUMS.
Sélectionne les rubriques « sœurs » ou les messages « frères ».

➔ **{mode=...}** → Boucle DOCUMENTS.
`{mode=document}` ou `{mode=image}`

➔ **{pagination}** → Toutes les boucles.
Pagine les résultats d'une boucle par 10. Pour modifier ce nombre, utiliser `{pagination=N}`

➔ **{par ... }**→ Toutes les boucles.
Critère de tri des résultats par un critère. Par date, par titre...

➔ **{par hasard}** → Toutes les boucles.
Retourne les résultats dans un ordre aléatoire.

➔ **{par num ...}** → Toutes les boucles.
Critère de tri des résultats par un critère numéroté. `{par num titre}` permettra de trier les résultats dont les titres ont été préalablement numérotés pour ne par prendre en compte l'ordre alphabétique : « 1. Mon premier article », « 2. Le deuxième article »...

➔ **{racine}** → Boucle RUBRIQUES.
Retourne les rubriques « secteur ».

➔ **{recherche}** → Boucles ARTICLES, RUBRIQUES, BREVES et FORUMS.
Affichage des résultats depuis une requête envoyée par `#FORMULAIRE_RECHERCHE`

➔ **{statut=...}** → Boucle ARTICLES. Sélectionne les articles selon leur état (`prop`, `prepa`, `publie`, `refuse` ou `poubelle`).

➔ **{tout}** → Boucles ARTICLES, RUBRIQUES, MOTS, FORUMS, AUTEURS et BREVES.
Retourne tous les objets même s'ils ne sont pas publiés.

➔ **{titre_mot=...}** et **{type_mot=...}** → Boucles ARTICLES, RUBRIQUES, SITES, FORUMS et BREVES.
`{titre_mot}` sélectionne les objets selon le titre d'un mot-clé appliqué.
`{type_mot}` sélectionne les objets selon le titre d'un groupe de mots-clés appliqué.

➔ **{titre=...}** → Boucle MOTS.
Sélectionne le mot-clé dont le titre est renseigné.

➔ **{type=...}** → Boucle MOTS.
Sélectionne les mots-clés appartenant au groupe de mots-clés indiqué.

➔ **{"inter"}** → Toutes les boucles.
Insère « inter » entre chaque résultats retournés. Peut être un caractère ou du code HTML.

➔ **{a, b}** → Toutes les boucles.
Limite les résultats retournés. `a` est le premier résultat à afficher, `b` le nombre de résultats.

➔ **{a/b}** → Toutes les boucles.
Limite les résultats de manière proportionnelle: ½ par exemple retournera la moitié des résultats.

➔ **{critère IN a,b,c...}** → Toutes les boucles.
Limite les résultats aux critères égaux à `a`, `b`, `c`, etc. Peut être un nombre pour les identifiants ou une chaîne de caractères.

## Des filtres

- **!={a}** ; **<={a}** ; **<{a}** ; **=={a}** ; **>={a}** ; **>{a}** → Compare le résultat de la balise avec la valeur `a`. Peut être un nombre ou une chaîne de caractères. s'utilise avec le filtre `?{...}`.
- **?{sioui, sinon}** → Teste une balise ou une comparaison. Retourne `sioui` si le retour du test est non vide. `sinon` est optionnel.
- **sinon{"..."}** → Équivalent en plus simple du filtre précédent.
- **affdate** → Pour afficher la date d'une autre manière que par défaut (`AAAA-MM-JJ hh:mm:ss`). Retourne la date au format texte « 12 mars 2009 »
- **couper** → Retourne les 50 premiers caractères du texte filtré, sans mise en forme. Le nombre de caractères peut être modifié : `couper{75}`
- **attribut_html** → Permet d'insérer l'élément filtré comme attribut dans une balise HTML.
- **inserer_attribut{attr, val}** → Insère des attributs dans les balises HTML générées par les balises Spip.
- **supprimer_numero** → Retire le numéro ajouté au titre d'un article, par exemple, pour le tri `{par num titre}`
- **supprimer_tags** → Retire les balises HTML de l'élément filtré.
- **textebrut** → Idem que `supprimer_tags` mais gère les caractères « invisibles » (sauts de lignes, espaces insécables...)
- **taille_en_octets** → Ajuste la valeur de la taille du fichier en octets, Ko, Mo...
- **unique** → N'affiche qu'une seule fois l'élément filtré.
- **couleur_xxx** → Filtres de couleurs. Extraire, foncer, éclaircir...
- **image_xxx** → Filtres d'images. Recadrer, réduire, flou, rotation, typo...