

Licence Pro MPST 2008

Création de sites Web avec les outils SPIP, HTML, et CSS



Camille Diou, LICM
Juin 2008

Table des matières

Avertissement	5	7. Principe des boucles SPIP	35
1. Introduction.....	7	7.1. Création des fichiers squelettes	35
2. Environnement de conception.....	7	7.1.1. Mots-clefs SPIP	36
2.1. Utilisation d'un CMS	7	7.1.2. Les boucles SPIP – syntaxe	36
2.1.1. Programmation	7	7.1.3. Fonctionnement d'une boucle.....	37
2.1.2. Conception graphique	8	7.2. Exemple : fichier rubrique.html	38
2.1.3. Rédaction	8	7.2.1. La boucle RUBRIQUES principale	38
2.1.4. Et le <i>webmaster</i> dans tout ça ?	8	7.2.2. Les boucles imbriquées	40
2.2. Organigramme de fonctionnement d'un		7.2.3. La boucle HIERARCHIE.....	45
CMS	8	7.3. La gestion des logos	46
2.3. SPIP	9	7.4. Listing complet de rubrique.html	48
2.4. Environnement de travail WAMP	9	8. Questions.....	49
3. Installation de EasyPHP.....	10	Annexe A – Liste d'éléments XHTML	50
4. Installation de SPIP	13	Balises de structuration du texte	50
4.1. Mise en place de la structure du site	13	Balises de liste	51
4.2. Installation de l'arborescence du site.....	14	Balises de tableau	51
5. Utilisation de SPIP.....	15	Balises de formulaire.....	52
5.1. Philosophie d'utilisation des CMS.....	15	Balises génériques	53
5.1.1. Introduction.....	15	Annexe B – Liste des propriétés CSS	54
5.1.2. Frontend et backend.....	15	Police, taille et décorations.....	54
5.1.3. Thème et structure du site.....	16	Alignement.....	54
5.2. Création de rubriques.....	16	Propriétés de couleur et de fond	55
5.3. Création d'articles	17	Couleur.....	55
5.3.1. Ajout d'éléments à l'article (image		Image de fond	55
ou document)	17	Propriétés des boîtes	55
5.3.2. Ecriture du premier article	18	Dimensions	55
5.3.3. Edition du site et ajout d'articles et		Marges extérieures	56
de rubriques	19	Marges intérieures	56
5.4. Vérification du site en ligne	20	Bordures.....	56
5.4.1. Rubrique de premier niveau		Propriétés de positionnement et d'affichage	57
(secteur) : « Le permis moto »	20	Affichage.....	57
5.4.2. Sous-rubrique « Le plateau »	20	Positionnement	57
5.4.3. Sous-rubrique « La poussette ».....	21	Propriétés des listes.....	57
5.4.4. Article « Parcours 1 »	21	Propriétés des tableaux.....	58
6. Utilisation avancée de SPIP : squelettes,		Autres propriétés.....	58
HTML et CSS	21	Annexe C – Liste des critères communs aux	
6.1. Base commune à tous les squelettes	22	boucles SPIP	59
6.2. Placement des boîtes grâce à CSS.....	23	Classer les résultats	59
6.3. Présentation des boîtes grâce à CSS.....	24	Comparaisons, égalités.....	59
6.4. Décorations et gestion des images	29	Affichage en fonction de la date.....	60
6.4.1. Image du bandeau de titre	29	Affichage d'une partie des résultats	60
6.4.2. Logos des secteurs	30	Affichage entre les résultats.....	61
6.4.3. Les menus de navigation.....	32	Divers	61
6.4.4. Résumé.....	33	Notes	61
6.4.5. Et ensuite ?	35		

Avertissement

Ce document introductif à la conception de sites Web par CMS permet de s'initier à l'utilisation des outils PHP, MySQL, Apache, et SPIP. Il n'est en aucun cas exhaustif et ne prétend pas couvrir tous les aspects de SPIP, mais a pour but de vous guider, pas à pas, dans l'installation des outils et l'écriture de vos premiers squelettes. Nous vous invitons fortement à consulter aux documentations en ligne de SPIP qui couvrent tous les aspects de ce logiciel :

- L'utilisation de l'interface d'administration et de rédaction :

http://www.spip.net/fr_rubrique136.html

- La rédaction de squelettes :

http://www.spip.net/fr_rubrique135.html

Notes :

Les exemples de ce tutoriel ont été mis en œuvre à l'adresse <http://193.50.119.127/spip/spip.php?rubrique1>.

L'espace d'administration de ce site vous est ouvert à l'adresse <http://193.50.119.127/spip/crire> en utilisant le login `lpromptst`, le mot de passe étant `lpromptst2008`. Vous pouvez vous y connecter de façon à recopier le contenu des rubriques et articles sur votre propre site. Vous pouvez y ajouter tout ce que vous voulez pour faire des essais, en veillant à ne pas tout effacer. Merci.

Les différentes images ayant servi pour la conception de ce site sont disponibles ici : <http://193.50.119.127/spip/squelettes/IMAGES/>.

1.Introduction

Ce tutoriel se veut être un guide pas à pas d'initiation à la réalisation de sites Web avec les technologies actuelles de conception orientées Internet. Tous les outils nécessaires pour cette formation sont des logiciels libres disponibles en téléchargement sur Internet.

Au fur et à mesure de la progression de ce document, des pré-requis sont nécessaires pour appréhender de manière complète la formation, sans que ces pré-requis ne soient pour autant indispensables : ils doivent vous permettre d'assimiler plus rapidement les concepts sous-jacents à la conception Web : structuration d'une page Web (HTML et CSS), programmation (JavaScript, boucles SPIP), notions de bases de données (MySQL), et interaction entre les différents outils de conception.

2.Environnement de conception

Le site Web que nous allons créer repose sur différentes technologies liées à Internet :

- CMS (*Content Management System*) SPIP
 - o scripts en PHP
 - o base de données MySQL
 - o squelettes HTML
- Feuilles de style CSS
- Programmes JavaScript
- Serveur Web

Il est nécessaire de bien comprendre comment ces différents outils interagissent afin de savoir où intervenir lors de l'élaboration du cahier des charges de notre site.

2.1. Utilisation d'un CMS

Un *Content Management System*, ou « Outil de gestion de contenu », est un outil permettant de concevoir un site Web en détachant les trois aspects de la conception :

- Conception graphique (présentation, rôle du « designer » ou du « graphiste »)
- Ecriture des « pages Web » (contenu, rôle du « rédacteur »)
- Programmation (rôle de « l'informaticien »)

Typiquement, ce type d'outil permet d'offrir des possibilités évoluées de gestion d'un site internet sans nécessiter de la part du *webmaster* des compétences très poussées dans les différents aspects énoncés ci-dessus.

De nombreux CMS sont disponibles et utilisables librement selon différentes licences libres (GPL, CECIL, etc.). Outre SPIP (<http://www.spip.net/fr>) que nous verrons plus en détail dans ce tutoriel, nous pouvons entre autres citer Joomla (<http://www.joomla.fr>), PwsPHP (<http://pwsphp.free.fr>), ou KwsPHP (<http://kws.koogar.org>) ...

2.1.1. Programmation

L'aspect programmation est réduit à sa plus simple expression, dans la mesure où toutes les fonctionnalités nécessaires sont déjà incluses dans le CMS. Néanmoins, il peut être nécessaire d'ajouter certaines fonctionnalités absentes.

Si ces fonctionnalités concernent l'interface du CMS, il sera nécessaire de modifier les scripts propres au CMS. Mais si ces fonctionnalités concernent l'aspect Web, du code sera ajouté au sein des squelettes du site (par exemple en PHP pour les scripts exécutés côté serveur, ou en JavaScript pour ceux exécutés côté client) : cela implique que le concepteur graphique et le programmeur travaillent ensemble sur l'écriture des squelettes du site.

2.1.2. Conception graphique

L'utilisation de « squelettes » permet de définir un ou plusieurs aspects graphiques au site Web de manière totalement indépendante du contenu. Cela nécessite de bien définir les différents blocs qui seront affichés sur la page Web (titre, sous-titre, auteur, texte, etc.). La nature de ces blocs est intimement liée au CMS employé, et leur emploi nécessite de bien comprendre les mécanismes propres au CMS (boucles SPIP par exemple) et la présentation d'une page Web (HTML, utilisation des feuilles de style CSS).

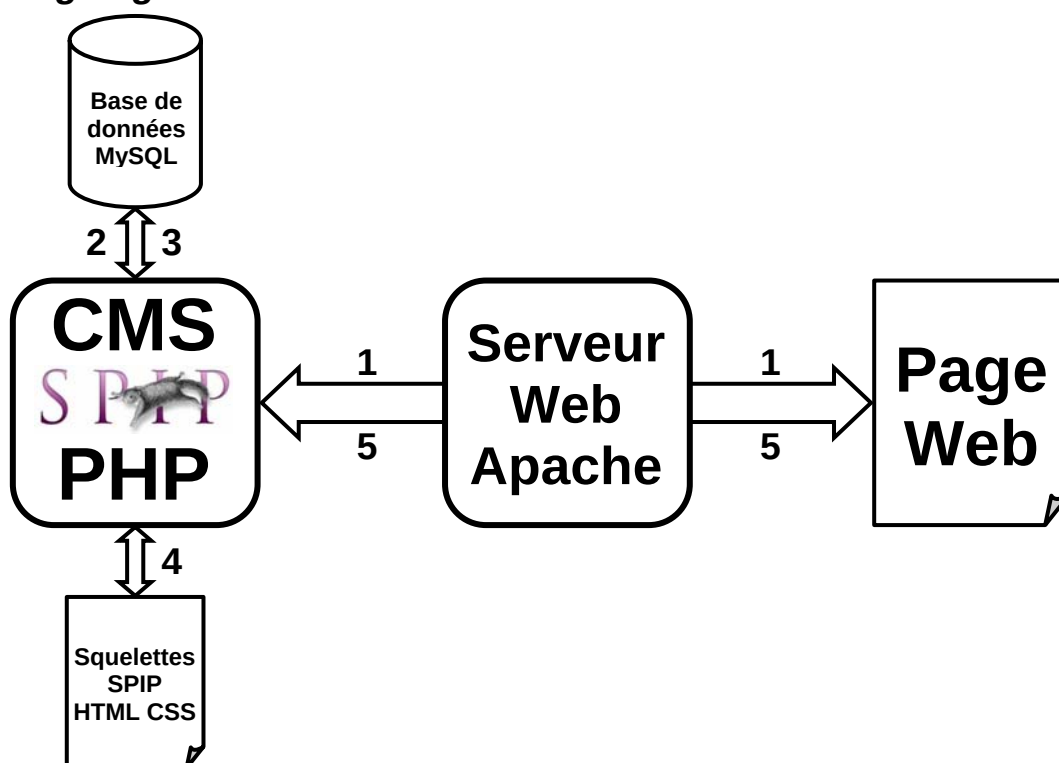
2.1.3. Rédaction

L'aspect rédaction se trouve grandement simplifié ; dès lors que le CMS est installé et fonctionnel, le rédacteur n'a besoin d'aucune compétence en programmation (HTML, JavaScript, PHP ou autre). De plus, les CMS permettent de définir un certain nombre de rédacteurs différents, chacun disposant de droits spécifiques ou communs dans l'utilisation de l'interface du CMS. Quiconque sait se servir d'un outil de rédaction de texte évolué (Microsoft Word ou OpenOffice) saura se servir de l'interface offerte par le CMS. La seule contrainte étant que la rédaction des pages Web se fait « en ligne » : il est nécessaire d'être connecté sur le site pour pouvoir ajouter du contenu.

2.1.4. Et le *webmaster* dans tout ça ?

Le rôle du webmaster est ici de faire le lien entre tous ces éléments de manière à proposer un site clair et bien structuré, en exploitant au maximum les possibilités offertes par le CMS, éventuellement en ajoutant les fonctionnalités absentes en les programmant (PHP ou JavaScript).

2.2. Organigramme de fonctionnement d'un CMS



L'organigramme ci-dessus montre bien le rôle central du CMS.

Outre le rôle d'interface (*frontend*) facilitant la rédaction des articles destinés à être publiés sur Internet, la véritable tâche du CMS est de préparer les pages Web (via le *backend*) afin de les afficher sur le navigateur, en suivant les étapes suivantes :

1. Réception de la requête via le serveur Web Apache
2. Envoi des informations à la base de données
3. Récupération des données dans la base
4. Formatage de la page Web en utilisant le squelette adapté à la requête
5. Envoi de la page Web au navigateur via le serveur Web Apache

2.3. SPIP

SPIP est un CMS très répandu qui utilise les technologies PHP et MySQL pour fonctionner.

PHP est un langage de script fonctionnant côté serveur (il est exécuté par le serveur, et non par le navigateur). C'est en PHP que sont codées toutes les fonctions du CMS : interface d'administration (*frontend*) et moteur de formatage des pages (*backend*).

MySQL est un système de gestion de bases de données libre reposant sur le langage SQL. La base de données contient toutes les informations nécessaires tant à l'interface de SPIP qu'au site Web en lui-même. Les requêtes SQL nécessaires pour gérer la base de données sont générées par le moteur de SPIP en fonction des requêtes HTTP (*HyperText Transfert Protocol*) : concrètement, à chaque fois qu'une page Web sera demandée, l'adresse de la page (URL) contiendra toutes les informations relatives à la nature de la page ainsi qu'à son contenu.

Un exemple de requête http envoyée à SPIP est le suivant :

<http://www.licm.fr/spip.php?rubrique46>

Cette adresse contient quatre informations indispensables à SPIP pour fonctionner :

1. `http://www.licm.fr/` est l'adresse de base du site
2. `spip.php` est le script PHP correspondant au moteur SPIP ; en pratique son rôle est de récupérer les informations suivantes et d'appeler les scripts PHP adéquats pour traiter l'information
3. `?rubrique46` est le paramètre envoyé au script `spip.php` ; il contient :
 - a. `rubrique` qui précise la nature du document (ici, une rubrique) ;
 - b. `46` qui identifie de manière unique le document (la 46ème rubrique du site).

Fort de toutes ces informations, le moteur de SPIP peut interroger la base de données afin de récupérer toutes les informations et données nécessaires au formatage de la page Web qui sera retournée au navigateur.

2.4. Environnement de travail WAMP

Nous savons maintenant quels sont les outils nécessaires à la réalisation d'un site Internet avec SPIP. L'utilisation intégrée de ces outils très largement répandus est référencée sous le terme de plateforme WAMP : **W**indows – **A**pache – **M**ySQL – **P**HP.

L'une des plateformes WAMP les plus répandues est EasyPHP. Cet outil rassemble bien entendu les 3 logiciels indispensables à notre projet (Apache, PHP, MySQL) mais également un outil d'administration de base de données (phpMyAdmin) lequel simplifie grandement notamment la création d'utilisateurs de la base de données, ainsi que la gestion de leurs droits. phpMyAdmin permet également de gérer et manipuler les bases de données ainsi que les tables dans ces bases de données, mais ceci sera effectué de manière totalement transparente lors de l'installation de SPIP.

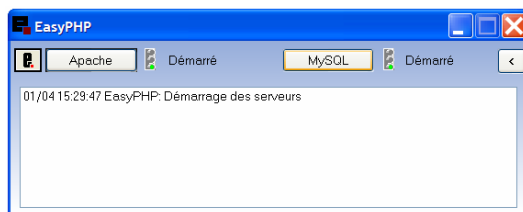
Nous allons maintenant procéder à l'installation d'EasyPHP.

3. Installation de EasyPHP

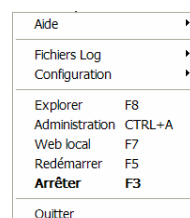
EasyPHP est disponible en libre téléchargement à l'adresse <http://www.easyphp.org/>.

Veiller à l'installer avec les droits administrateur sur votre machine, sinon il refusera de s'installer. Suivez les indications lors de l'installation, puis démarrez l'application. Une icône dans la barre des tâches vous indiquera si EasyPHP a bien été lancé (🖱).

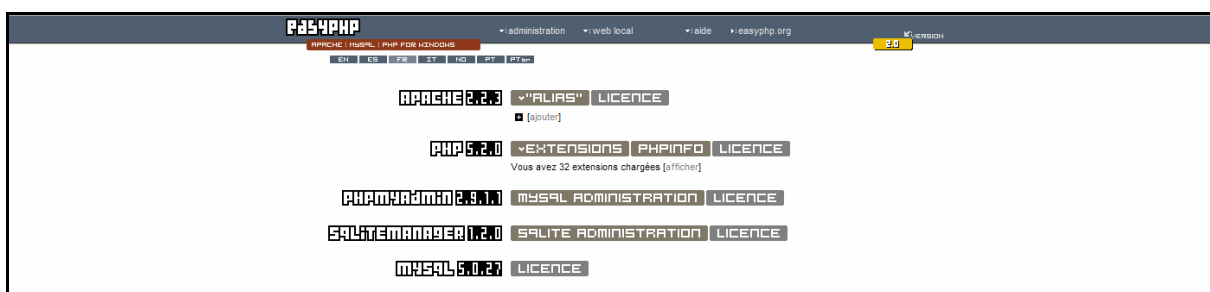
Un double-clic sur cette icône vous précisera l'état des serveurs Apache et MySQL :



Un clic droit sur cette icône laisse apparaître le menu déroulant ci-contre. Pour accéder à la page Web d'administration de vos serveurs (principalement MySQL via phpMyAdmin), cliquez sur Administration ou CTRL+A.



La page suivante devrait alors s'ouvrir dans votre navigateur Internet (nous vous conseillons d'utiliser Mozilla Firefox) :



Cliquez sur **MYSQL ADMINISTRATION** afin d'ouvrir la page d'administration des bases de données MySQL.



Pour fonctionner, SPIP a besoin d'accéder à la base de données afin d'y écrire et y lire les informations nécessaires à son fonctionnement. Pour ajouter un utilisateur, cliquez sur **Privilèges**.

La page qui s'ouvre alors recense tous les utilisateurs autorisés à se connecter à la base de données. Vous devriez avoir quelque chose ressemblant à la capture d'écran suivante :

phpMyAdmin

Seigneur: localhost

Bases de données SQL État Variables Jeux de caractères Moteurs Privilèges Log binaire Processus Exporter

Importer

Vue d'ensemble des utilisateurs

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [Tout afficher]

Utilisateur	Serveur	Mot de passe	Privilèges globaux	"Grant"
☐ N'importe quel	%	--	USAGE	Non
☐			SELECT, INSERT, UPDATE, CREATE	Non
☐			SELECT, INSERT, UPDATE, CREATE	Non
☐ root	localhost	Non	ALL PRIVILEGES	Oui

Tout cocher / Tout décocher

Ajouter un utilisateur

Effacer les utilisateurs sélectionnés.

(Effacer tous les privilèges de ces utilisateurs, puis les effacer.)

☐ Supprimer les bases de données portant le même nom que les utilisateurs

Exécuter

Note: phpMyAdmin obtient la liste des privilèges directement à partir des tables MySQL. Le contenu de ces tables peut être différent des privilèges effectifs, si des changements manuels ont été apportés. Dans ce cas, vous devriez recharger les privilèges avant de continuer.

Ouvrir une nouvelle fenêtre phpMyAdmin

NB : le cadre gris entouré de pointillés rouge ne sert qu'à masquer les utilisateurs déjà enregistrés sur le système ayant servi de plateforme de test pour ce tutoriel.

Cliquez sur **Ajouter un utilisateur** pour ajouter le compte spip. Ce compte doit au minimum avoir les privilèges globaux portant sur les données (Select, Insert, Update, Delete, File) pour la base nécessaire à notre site. SPIP pouvant lui-même créer une base de données, à condition d'en avoir les privilèges, nous allons également attribuer les privilèges portant sur les structures à l'utilisateur spip (Create, Alter, Drop).

Complétez les champs Nom d'utilisateur et Mot de passe, cochez les cases correspondant aux privilèges ci-dessus, puis validez en cliquant sur Exécuter.

phpMyAdmin

Seigneur: localhost

Bases de données SQL État Variables Jeux de caractères Moteurs Privilèges Log binaire Processus

Exporter Importer

Ajouter un utilisateur

Information pour la connexion

Nom d'utilisateur: Entrez une valeur: spip

Serveur: Tout serveur

Mot de passe: Entrez une valeur: *****

Entrer à nouveau: *****

Générer un mot de passe: Générer Copier

Base de données pour cet utilisateur

☒ Aucune

☐ Créer une base portant son nom et donner à cet utilisateur tous les privilèges sur cette base

☐ Donner les privilèges passepartout ("%")

Privilèges globaux (Tout cocher / Tout décocher)

Veuillez noter que les noms de privilèges sont exprimés en anglais

Données	Structure	Administration	Limites de ressources
☒ SELECT	☒ CREATE	☐ GRANT	Note: Une valeur de 0 (zéro) enlève la limite.
☒ INSERT	☒ ALTER	☐ SUPER	MAX QUERIES PER HOUR 0
☒ UPDATE	☐ INDEX	☐ PROCESS	MAX UPDATES PER HOUR 0
☒ DELETE	☒ DROP	☐ RELOAD	MAX CONNECTIONS PER HOUR 0
☒ FILE	☐ CREATE TEMPORARY TABLES	☐ SHUTDOWN	MAX USER CONNECTIONS 0
	☐ CREATE VIEW	☐ SHOW DATABASES	
	☐ SHOW VIEW	☐ LOCK TABLES	
	☐ CREATE ROUTINE	☐ REFERENCES	
	☐ ALTER ROUTINE	☐ REPLICATION CLIENT	
	☐ EXECUTE	☐ REPLICATION SLAVE	
		☐ CREATE USER	

Exécuter

Ouvrir une nouvelle fenêtre phpMyAdmin

En l'état actuel des choses, tout est prêt pour que SPIP puisse fonctionner. La page suivante s'affiche, récapitulant toutes les informations :

phpMyAdmin

Serveur: localhost

Bases de données SQL État Variables Jeux de caractères Moteurs Privilèges Log binaire Processus

Exporter Importer

Vous avez ajouté un utilisateur

requête SQL:

```
CREATE USER 'spip'@'%' IDENTIFIED BY '*****';

GRANT SELECT ,
INSERT ,

UPDATE ,
DELETE ,
CREATE ,
DROP ,
FILE ,
ALTER ON * . * TO 'spip'@'%' IDENTIFIED BY '*****' WITH MAX_QUERIES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0 MAX_UPDATES_PER_HOUR 0
MAX_USER_CONNECTIONS 0 ;
```

[Modifier] [Créer source PHP]

Utilisateur 'spip'@'%': Changer les privilèges

Privilèges globaux (Tout cocher / Tout décocher)

Veillez noter que les noms de privilèges sont exprimés en anglais

Données	Structure	Administration	Limites de ressources
☒ SELECT	☒ CREATE	☐ GRANT	Note: Une valeur de 0 (zero) enlève la limite. MAX QUERIES PER HOUR 0 MAX UPDATES PER HOUR 0 MAX CONNECTIONS PER HOUR 0 MAX USER_CONNECTIONS 0
☒ INSERT	☒ ALTER	☐ SUPER	
☒ UPDATE	☒ INDEX	☐ PROCESS	
☒ DELETE	☒ DROP	☐ RELOAD	
☒ FILE	☐ CREATE TEMPORARY TABLES	☐ SHUTDOWN	
	☐ CREATE VIEW	☐ SHOW DATABASES	
	☐ SHOW VIEW	☐ LOCK TABLES	
	☐ CREATE ROUTINE	☐ REFERENCES	
	☐ ALTER ROUTINE	☐ REPLICATION CLIENT	
	☐ EXECUTE	☐ REPLICATION SLAVE	
		☐ CREATE USER	

Exécuter

Privilèges spécifiques à une base de données

Base de données Privilèges "Grant" Privilèges spécifiques à une table Action

aucune

Ajouter des privilèges sur cette base de données: Entrez une valeur: ?

Exécuter

Modifier le mot de passe

☐ aucun mot de passe

☐ Mot de passe:

Entrer à nouveau:

Exécuter

Changement des informations de connexion / Copie d'utilisateur

Information pour la connexion

Nom d'utilisateur: Entrez une valeur:

Serveur: Tout serveur

Mot de passe: Conserver le mot de pa

Entrer à nouveau:

Générer un mot de passe:

Créer un nouvel utilisateur avec les mêmes privilèges et ...

☒ ... conserver intact l'ancien utilisateur.

☐ ... supprimer l'ancien utilisateur.

☐ ... effacer tous les privilèges de l'ancien utilisateur, puis l'effacer.

☐ ... supprimer l'ancien utilisateur, puis recharger les privilèges au serveur.

Exécuter

Terminé

Pour compléter, il est indispensable de rappeler quelques notions élémentaires de sécurité.

L'utilisateur `spip` dispose de privilèges globaux lui donnant accès aux données et structures de toutes les bases de données du système. Il serait judicieux de n'attribuer ces droits que pour une base de données précise afin de limiter les risques.

Veillez à toujours attribuer un mot de passe à tous les comptes MySQL, y compris et surtout le compte `root` !

La gestion des bases de données sortant du cadre de ce tutoriel, nous vous invitons à consulter les documents relatifs à MySQL et phpMyAdmin pour de plus amples informations.

Notre serveur de base de données est maintenant prêt pour accueillir une base de données propre à notre serveur Web, bien que celle-ci ne soit pas encore créée : c'est SPIP qui s'en chargera, grâce à l'utilisateur `spip` que nous venons de créer.

4. Installation de SPIP

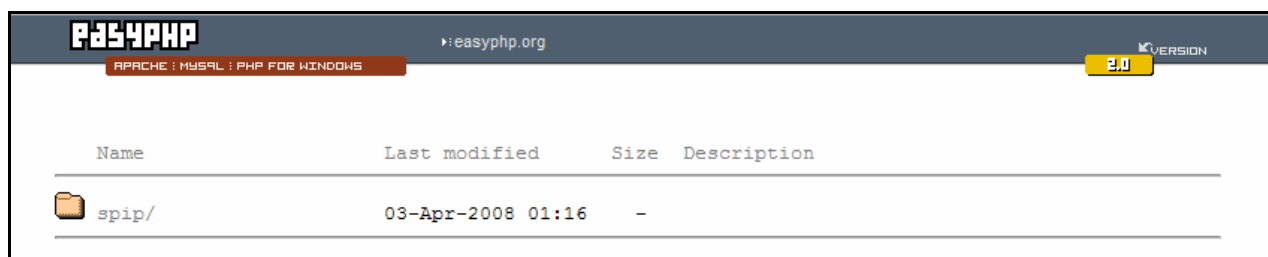
4.1. Mise en place de la structure du site

Téléchargez la dernière version de SPIP à l'adresse suivante : http://www.spip.net/fr_download.

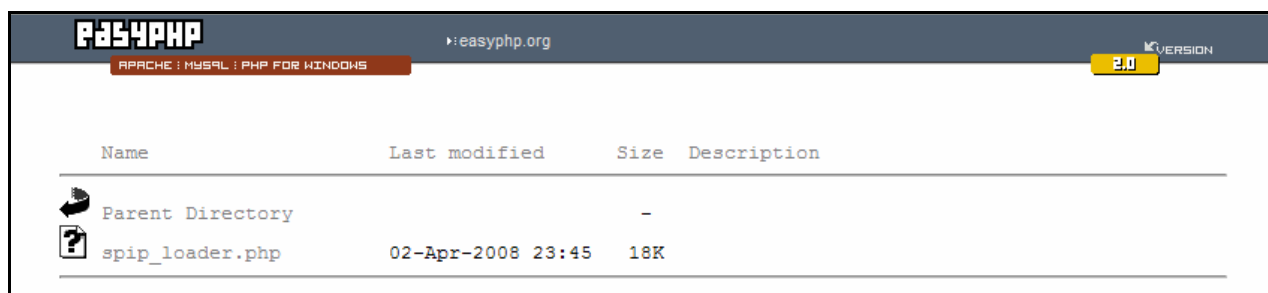
Prenez la version de SPIP marquée « Installation automatique », `spip_loader.php`, disponible à l'adresse suivante : http://www.spip.net/spip-dev/INSTALL/spip_loader.php, et copiez ce fichier dans le répertoire de base de votre site Web. Lors d'une installation par défaut de EasyPHP, le répertoire par défaut de tous les sites Web est `C:\Program Files\Internet\EasyPHP 2.0b1\www\`. Créez-y un répertoire `spip`, puis copiez-y le fichier `spip_loader.php`.

Ouvrez maintenant la page « Web local » de votre serveur Web en utilisant le choix correspondant dans le menu contextuel d'EasyPHP (clic droit sur ) ou en pressant la touche F7.

Vous obtenez la page suivante, où vous ne trouverez qu'un seul répertoire : le répertoire `spip` que vous avez créé précédemment :



Cliquez sur le répertoire `spip`, puis dans la page qui suivra, cliquez sur le fichier `spip_loader.php`. Ce script écrit en PHP sera exécuté par le serveur Apache via les extensions PHP afin de télécharger toute la structure d'un site SPIP :



Vous obtenez la page suivante, qui vous permettra de procéder pas à pas à l'installation de SPIP.

4.2. Installation de l'arborescence du site

Cliquez sur le bouton  afin de continuer. ⬆

Cliquez sur  afin de continuer, puis indiquez l'adresse (localhost), le login (spip) et le mot de passe de connexion à la base de données, puis cliquez sur le

Installation du système de publication...

Première étape : Votre connexion MySQL

Consultez les informations fournies par votre hébergeur : vous devez y trouver, si votre hébergeur supporte MySQL, les codes de connexion au serveur MySQL. (9)

Adresse de la base de données

(Souvent cette adresse correspond à celle de votre site, parfois elle correspond à la mention «localhost», parfois elle est laissée totalement vide.)

Le login de connexion

(Correspond parfois à votre login d'accès au FTP, parfois laissé vide)

Le mot de passe de connexion

(Correspond parfois à votre mot de passe pour le FTP, parfois laissé vide)

Suivant >>

Installation du système de publication...

Deuxième étape : Essai de connexion à la base

La connexion a réussi.

Vous pouvez passer à l'étape suivante.

Suivant >>

Cliquez sur le bouton  afin de continuer. ⬆

Dans la page suivante, vous pourrez sélectionner une base de données parmi celles disponibles, ou en créer une nouvelle.

Sélectionnez le dernier choix :
« Créer une nouvelle base
de données ». ➡

Installation du système de publication...

Troisième étape : Choix de votre base (8)

Choisissez votre base :

Le serveur MySQL contient plusieurs bases de données.
Sélectionnez ci-après celle qui vous a été attribuée par votre hébergeur :

☒ **informations_schema**

☐ test

ou... ☒ **Créer une nouvelle base de données :**

Suivant >>

Cliquez sur le bouton afin de continuer. ⬆

Installation du système de publication...

Quatrième étape : Création des tables de la base

La structure de votre base de données est installée.

Vous pouvez passer à l'étape suivante.

Suivant >>

Si la connexion à la base de données s'est correctement déroulée, la page ci-dessus apparaît. Cliquez sur le bouton  afin de continuer. ↑

Dans la page suivante,
complétez les informations
relatives à votre compte
administrateur SPIP, au
minimum vos signature

Installation du système de publication...

Cinquième étape : Informations personnelles

Le système va maintenant vous créer un accès personnalisé au site. 

(Note : s'il s'agit d'une réinstallation, et que votre accès marche toujours, vous pouvez laisser ces champs vides)

— Votre identité publique...

Signature
(Votre nom ou votre pseudo)

— Vos identifiants de connexion...

Votre login
(Plus de 3 caractères)

Votre mot de passe
(Plus de 5 caractères)

Confirmer ce nouveau mot de passe :

Cliquez sur le bouton Suivant >>. 

Installation du système de publication...

Dernière étape : C'est terminé !

N'oubliez pas vos propres codes d'accès !

Vous pouvez maintenant commencer à utiliser le système de publication assistée...

[Suivant >>](#)

Arrivé à cette étape, tout s'est bien passé et SPIP est prêt à être utilisé.

Cliquez sur le bouton  pour accéder à la fenêtre de login de SPIP. ⬆

Indiquez votre login avant de cliquer sur , puis indiquez votre mot de passe.

Mon site SPIP

accès à l'espace privé

français

Identifiants personnels

Login : **camille**
[se connecter sous un autre identifiant]

Mot de passe :
* * * * *

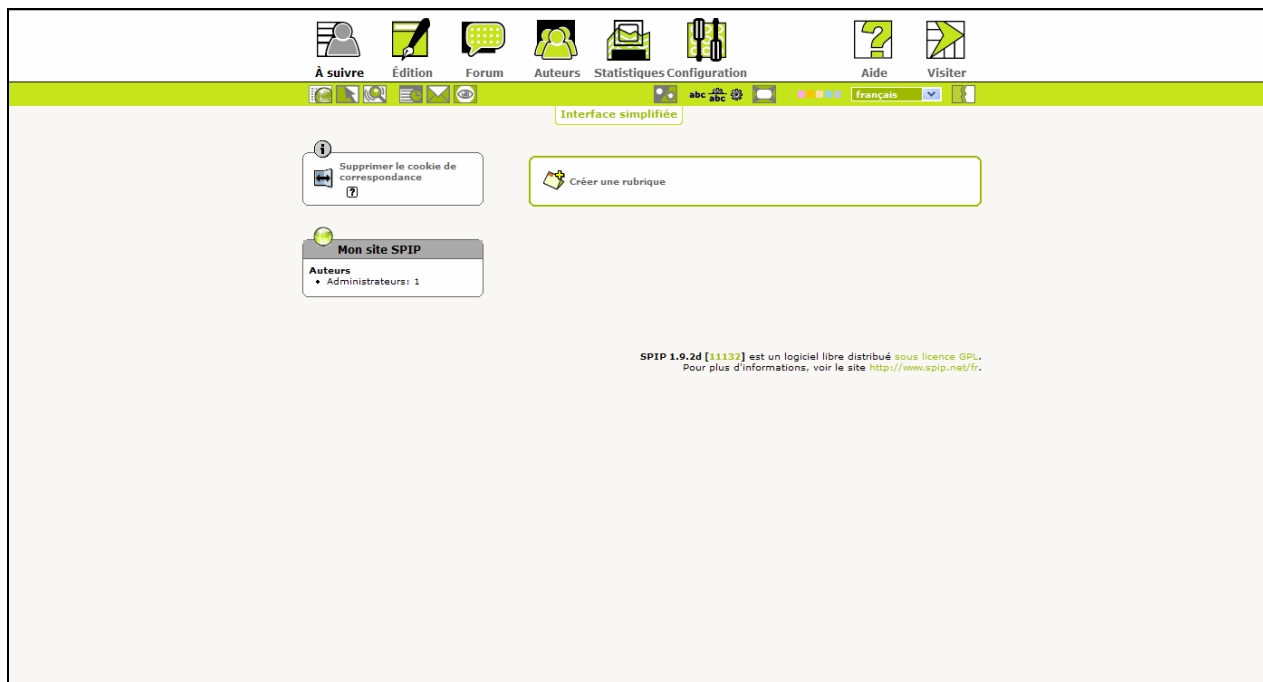
☒ Rester identifié quelques jours

Valider

Cliquez sur  pour accéder à l'interface de SPIP :



5. Utilisation de SPIP



5.1. Philosophie d'utilisation des CMS

5.1.1. Introduction

Les CMS tels que SPIP sont conçus de manière à faciliter la réalisation et la gestion de sites internet de nature rédactionnelle (« journaux » en lignes) destinés à évoluer régulièrement. Il est donc nécessaire de bien définir l'architecture globale du site en « rubriques », lesquelles vont accueillir les « articles » proposés par les rédacteurs (ou « auteurs »). Les « rubriques » correspondent donc aux différentes thématiques traitées par le site. Le rôle des « articles » semble évident.

Le but de cette section n'est pas de vous donner un aperçu exhaustif des fonctionnalités de SPIP, mais plutôt de vous guider dans son utilisation pour permettre de lui donner suffisamment de contenu afin de développer ensuite un squelette pour le site. Nous vous renvoyons donc à la documentation de SPIP pour de plus amples informations, mais ce tutoriel contiendra toute la matière nécessaire à la réalisation du site.

5.1.2. Frontend et backend

Comme tout CMS, SPIP est constitué de deux principales parties :

- Le *frontend*, ou interface, est la partie d'administration du site qui vous permet d'ajouter du contenu. Il permet entre autres de créer des rubriques, des articles, des brèves, d'ajouter des auteurs, des mots-clefs associés aux rubriques ou aux articles, et de configurer les nombreuses fonctionnalités de SPIP ;
- Le *backend* est le moteur de SPIP. C'est lui qui va analyser les squelettes et interroger la base de données pour formater les pages HTML destinées à être affichées sur le navigateur Web. La connaissance du fonctionnement du *backend* n'est pas indispensable à la réalisation d'un site, mais la connaissance du langage propre à SPIP est fortement recommandée pour développer un site personnalisé et exploitant au maximum les fonctionnalités du CMS.

Sachez cependant que de nombreux squelettes sont disponibles sur internet : ils vous permettront de proposer un site très riche en termes de présentation et de fonctionnalités

sans avoir à « mettre les mains dans le cambouis », mais pas forcément très adapté à vos besoins.

Nous aborderons ce second point dans la partie consacrée à l'écriture de squelettes.

5.1.3. Thème et structure du site

Le site que nous allons développer a pour seul objectif de vous permettre d'appréhender simplement les notions élémentaires de « boucles SPIP » et l'utilisation des mots-clefs. Il sera donc construit au fur et à mesure de ce tutoriel, simple et peu fourni, mais vous permettra ensuite d'envisager le développement d'un site beaucoup plus riche tant en présentation qu'en fonctionnalités : les notions abordées lors de ce tutoriel seront directement transposables aux autres fonctionnalités offertes par SPIP.

À titre d'exemple, nous allons construire un site sur le thème de la moto architecturé comme suit (les rubriques apparaissent en gras) :

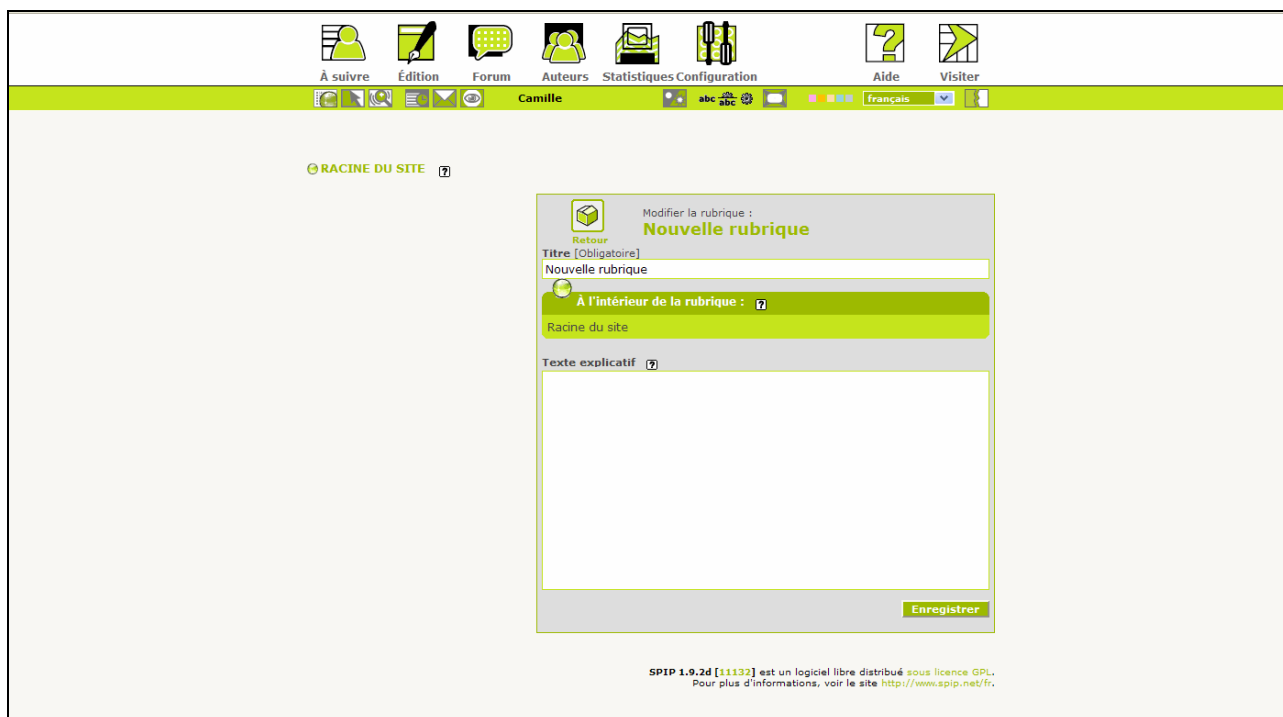
- Sommaire
- **Le permis moto : les différentes étapes de la formation**
 - o **Le plateau**
 - **La poussette**
 - Parcours 1
 - Parcours 2
 - Parcours 3
 - **Les vérifications techniques**
 - Chaine cinématique et livret
 - Partie cycle et gants
 - Eclairage et accessoires de sécurité moto/pilote
 - o **La conduite**
- **Après le permis**
- **Les différentes catégories de moto**
 - o Roadsters
 - o Routières
 - o Sportives
 - o Trails
 - o Customs
 - o Supermotards

5.2. Création de rubriques

En l'état actuel des choses, votre site ne contient naturellement aucune rubrique ni aucun article. Avant d'ajouter des articles, il est nécessaire de créer les rubriques devant les recevoir. Le « sommaire » étant constitué d'une page particulière, il n'est pas nécessaire de créer une rubrique dédiée. Nous aborderons ce point particulier plus tard.

Pour créer une rubrique, cliquez sur  [Créer une rubrique](#). Vous obtenez la page présentée page suivante. Donnez un titre à la rubrique puis éventuellement, une courte description de son contenu avant de cliquer sur [Enregistrer](#).

Note : une rubrique qui ne contient aucun article n'apparaît pas en ligne



Répétez cette opération pour créer toutes les rubriques du site.

5.3. Création d'articles

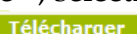
La création d'articles est aussi simple que la création de rubriques à ceci près que beaucoup plus de possibilités sont offertes.

Un article comporte un titre (obligatoire), un sous-titre, un chapeau (texte introductif à l'article), ainsi que le texte de l'article. Il peut également comporter un post-scriptum, à condition d'activer l'interface complète via le bouton  de la barre de menu.

Les différentes options offertes par l'interface complète de SPIP sortant du cadre de ce tutoriel, nous n'en ferons plus mention, sauf exceptions clairement indiquées (certaines fonctionnalités nécessitent d'activer l'interface complète).

5.3.1. Ajout d'éléments à l'article (image ou document)

Vous pouvez illustrer votre article à l'aide d'images ou de documents divers. Ces images et ces documents pourront être gérés directement depuis l'interface de SPIP et insérés dans l'article, ou bien à l'aide des boucles SPIP au sein du squelette de l'article.

Pour insérer une image, cliquez sur le bouton  dans la zone « ajouter une image », sélectionnez l'image sur votre disque dur, validez, puis cliquez sur . La fenêtre ci-contre apparaît alors, dans laquelle vous pourrez renseigner les champs « titre » et « description ». Vous pourrez également modifier la date de mise en ligne si nécessaire.

Pour utiliser l'image dans l'article, les raccourcis typographiques spécifiques à SPIP sont précisés dans le cadre de l'image. Pour insérer l'image, indiquez simplement `<img1>`. Pour l'insérer sous forme de flottant aligné à gauche, indiquez `<img1|left>`, pour la centrer indiquez `<img1|center>`, ou `<img1|right>` pour l'aligner à droite. L'image sera alors parfaitement intégrée au texte, le moteur d'affichage de SPIP se chargeant de l'insertion du texte autour de l'image en fonction du paramètre d'alignement précisé (à titre d'exemple, le placement de l'image ci-dessus à droite serait obtenu par `<img1|right>`).



L'insertion de documents suit la même logique, à ceci près que le document est inséré dans le texte à l'aide du raccourci <docXY> où XY représente le numéro du document. Dans le cas d'images insérées sous forme de documents, c'est alors une miniature de l'image qui serait insérée : un clic sur cette miniature affichera alors l'image dans sa taille d'origine.

5.3.2. Ecriture du premier article

Complétez le premier article (« Parcours 1 ») avec les informations suivantes :

Titre : Parcours 1

Sous-titre : Premier parcours sans l'aide du moteur

Chapeau : Schéma du premier des trois parcours à connaître pour l'épreuve du plateau sans moteur

Texte de l'article :

```
<img1|right>
{{Position de la moto au départ :}}

L'arrière est au niveau de la ligne
C3. La moto est sur la béquille
centrale ou latérale.

-# débéquiller
-# pousser la moto dans le sens
{{a-b}}
-# rebéquiller (arrière au niveau
C3)

|{{Évaluation}}|{{A}}|{{B}}|
|Exercice réalisé||X|
|Exercice non réalisé||X|
|Chute de la machine||X|
|Cône déplacé par la moto||X|
|Erreur de parcours||X|
|5 vérifications exactes||X|
|Moins de 5 vérifications
exactes||X|
```

Vous pouvez reconnaître ici 4 types de raccourcis typographiques :

1. insertion d'image : `<img...>`
2. texte gras : `{{ ... }}`
3. liste numérotée : `-#`
4. tableau : `|...|...|...|`

Cliquez sur **Enregistrer** pour enregistrer l'article.

À ce stade, l'article est présent en base de données, mais n'est toujours pas présent sur le Web. Vous êtes par défaut l'administrateur de ce site, vous allez donc pouvoir immédiatement valider cet article. Cliquez sur **Cet article est :** `en cours de rédaction`, sélectionnez `publié en ligne` puis cliquez sur **Valider**. Votre article est dorénavant présent en ligne. Cliquez sur  **Voir en ligne** juste sous le numéro de l'article pour le voir tel qu'il apparaît en ligne.

5.3.3. Edition du site et ajout d'articles et de rubriques

Avant d'aller plus loin, et de définir des squelettes sur mesure pour le site, complétez les différentes rubriques et les différents articles du site, afin de constater l'évolution du site en ligne en fonction de l'ajout des nouveaux articles. Il peut être nécessaire de cliquer sur **Recalculer cette page** en haut à droite de votre écran pour forcer le recalcul des pages en fonction du nouveau contenu de la base de données.

C'est au fur et à mesure de l'ajout de nouveaux articles et de nouvelles rubriques que l'on constate tout l'intérêt d'un outil comme SPIP : l'entretien du site se limite à du rédactionnel, toutes les fonctionnalités nécessaires à sa mise en œuvre étant offertes par l'interface ou par les squelettes que nous découvrirons dans la section 6.

N'hésitez pas à consulter l'aide en ligne de SPIP pour découvrir ou approfondir toutes les fonctions offertes aux rédacteurs ainsi qu'aux administrateurs du site. Cette aide en ligne est accessible en cliquant sur l'icône



ARTICLE NUMÉRO :
1

LOGO DE L'ARTICLE ?

FORUM & PÉTITION ?

Parcours 1

DATE DE CRÉATION DE L'ARTICLE : 16 AVRIL 2008 ?

LES AUTEURS

Camille 1 article

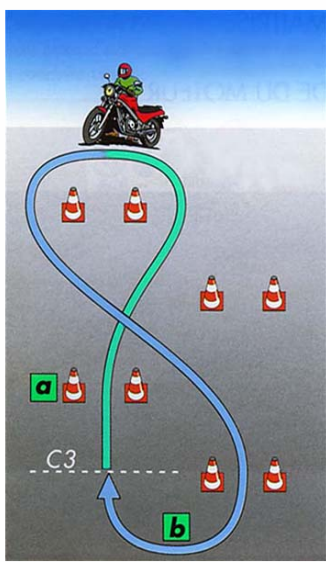
Cet article est : `en cours de rédaction`

Position de la moto au départ :

L'arrière est au niveau de la ligne C3. La moto est sur la béquille centrale ou latérale.

1. débéquiller
2. pousser la moto dans le sens a-b
3. rebéquiller (arrière au niveau C3)

Évaluation A B	
Exercice réalisé	X
Exercice non réalisé	X
Chute de la machine	X
Cône déplacé par la moto	X
Erreur de parcours	X
5 vérifications exactes	X
Moins de 5 vérifications exactes	X



Modifier cet article

JOINDRE UN DOCUMENT

Télécharger depuis votre ordinateur :

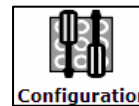
Parcourir...

Télécharger

5.4. Vérification du site en ligne

Les captures d'écran pages suivantes montrent l'aspect du site SPIP par défaut. On peut constater que si la mise en place d'un site sous SPIP est très simple, la charte graphique par défaut est très pauvre. C'est ici qu'interviennent les compétences du concepteur de site pour enrichir les pages et leur donner l'allure qu'il souhaite.

À ce stade, nous pouvons constater quelques « oublis » dans la configuration du site. En premier lieu, le titre du site est « Mon site SPIP ». Pour le changer, il suffit de cliquer sur « configuration » dans la barre de menu de l'interface SPIP, puis d'indiquer le nom que vous voulez donner à votre site dans la zone de texte adéquate. Remplacez simplement « Mon site SPIP » par le nom que vous souhaitez. « Tout sur la moto » serait un nom adapté, bien que trompeur.



Configuration



D'autres « défauts » apparaissent, tels que les numéros devant les titres de rubriques. Nous verrons, lors de l'élaboration des squelettes, leur intérêt ainsi que le moyen de ne pas les faire apparaître sur le site en ligne.

Profitez également de votre visite du site pour vérifier la présence des différents éléments relatifs aux rubriques, aux articles, ainsi qu'à l'auteur : titres, sous-titres, chapeau, descriptions, dates, nom de l'auteur, etc.

5.4.1. Rubrique de premier niveau (secteur) : « Le permis moto »

Mon site SPIP Rechercher Modifier cette rubrique (1) Recalculer cette page

Accueil du site > 1. Le permis moto : les différentes étapes de la formation

1. Le permis moto : les différentes étapes de la formation

Dernier ajout : 17 avril.

Cette rubrique est consacrée au permis moto. Vous y trouverez les différentes étapes de la formation ainsi que toutes les informations nécessaires et indispensables pour le mener à bien.

Vous y trouverez les différents parcours à connaître et à maîtriser pour passer l'épreuve plateau.

Toutes les connaissances théoriques (vérifications technique, fiches moto, et panneaux) que vous devrez apprendre vous seront présentées sous forme de fiches.

Sous-rubriques

- 1. Le plateau
 - 1. La poussette (parcours sans l'aide du moteur)

Rubriques

- 1. Le permis moto : les différentes étapes de la formation
 - 1. Le plateau

RSS 2.0 | Plan du site | Espace privé | spip | squelette

5.4.2. Sous-rubrique « Le plateau »

Mon site SPIP Rechercher Modifier cette rubrique (4) Recalculer cette page

Accueil du site > 1. Le permis moto : les différentes étapes de la formation > 1. Le plateau

1. Le plateau

Dernier ajout : 17 avril.

Sous-rubriques

- 1. La poussette (parcours sans l'aide du moteur)

Rubriques

- 1. Le permis moto : les différentes étapes de la formation
 - 1. Le plateau
 - 1. La poussette (parcours sans l'aide du moteur)

RSS 2.0 | Plan du site | Espace privé | spip | squelette

5.4.3. Sous-rubrique « La poussette »

Mon site SPIP

Accueil du site > 1. Le permis moto : les différentes étapes de la formation > 1. Le plateau > 1. La poussette (parcours sans l'aide du moteur)

1. La poussette (parcours sans l'aide du moteur)

Dernier ajout : 17 avril.

Parcours 1

17 avril, par Camille

Rubriques

1. Le permis moto : les différentes étapes de la formation

1. Le plateau

1. La poussette (parcours sans l'aide du moteur)

RSS 2.0 | Plan du site | Espace privé | spip | squelette

5.4.4. Article « Parcours 1 »

Mon site SPIP

Accueil du site > 1. Le permis moto : les différentes étapes de la formation > 1. Le plateau > 1. La poussette (parcours sans l'aide du moteur) > Parcours 1

Parcours 1

Premier parcours sans l'aide du moteur

Jeu 17 avril 2008, par Camille

Schéma du premier des trois parcours à connaître pour l'épreuve du plateau sans moteur.

Position de la moto au départ :

L'arrière est au niveau de la ligne C3. La moto est sur la béquille centrale ou latérale.

1. débéquiller

2. pousser la moto dans le sens a-b

3. rebéquiller (arrière au niveau C3)

Évaluation	A	B
Exercice réalisé	X	
Exercice non réalisé		X
Chute de la machine	X	
Cône déplacé par la moto	X	
Erreur de parcours	X	
5 vérifications exactes	X	
Moins de 5 vérifications exactes		X

Dans la même rubrique

Parcours 1

Répondre à cet article

RSS 2.0 | Plan du site | Espace privé | spip | squelette

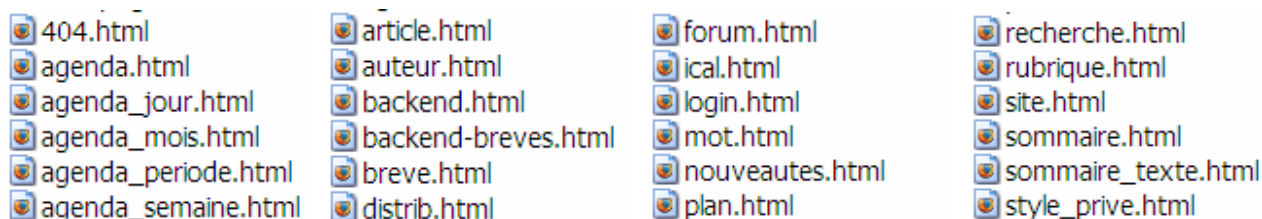
6.Utilisation avancée de SPIP : squelettes, HTML et CSS

Le véritable travail de conception d'un site sous SPIP est l'écriture des squelettes. Ces squelettes sont des fichiers HTML qui décrivent la présentation graphiques des différents éléments constituant le site (rubriques, articles, brèves, calendrier, etc.). Chaque type de page qu'est capable de générer SPIP dispose donc d'un squelette adapté ; en voici une liste exhaustive ¹ à ce jour :

¹ « exhaustive » n'est pas tout à fait le terme adapté, puisque vous pouvez définir autant de fichiers squelettes que vous le désirez : SPIP est un outil très souple qui permet de nombreuses adaptations personnelles. Ce terme fait en fait référence aux squelettes « par défaut » fournis avec la distribution SPIP.

C. Diou – LICM

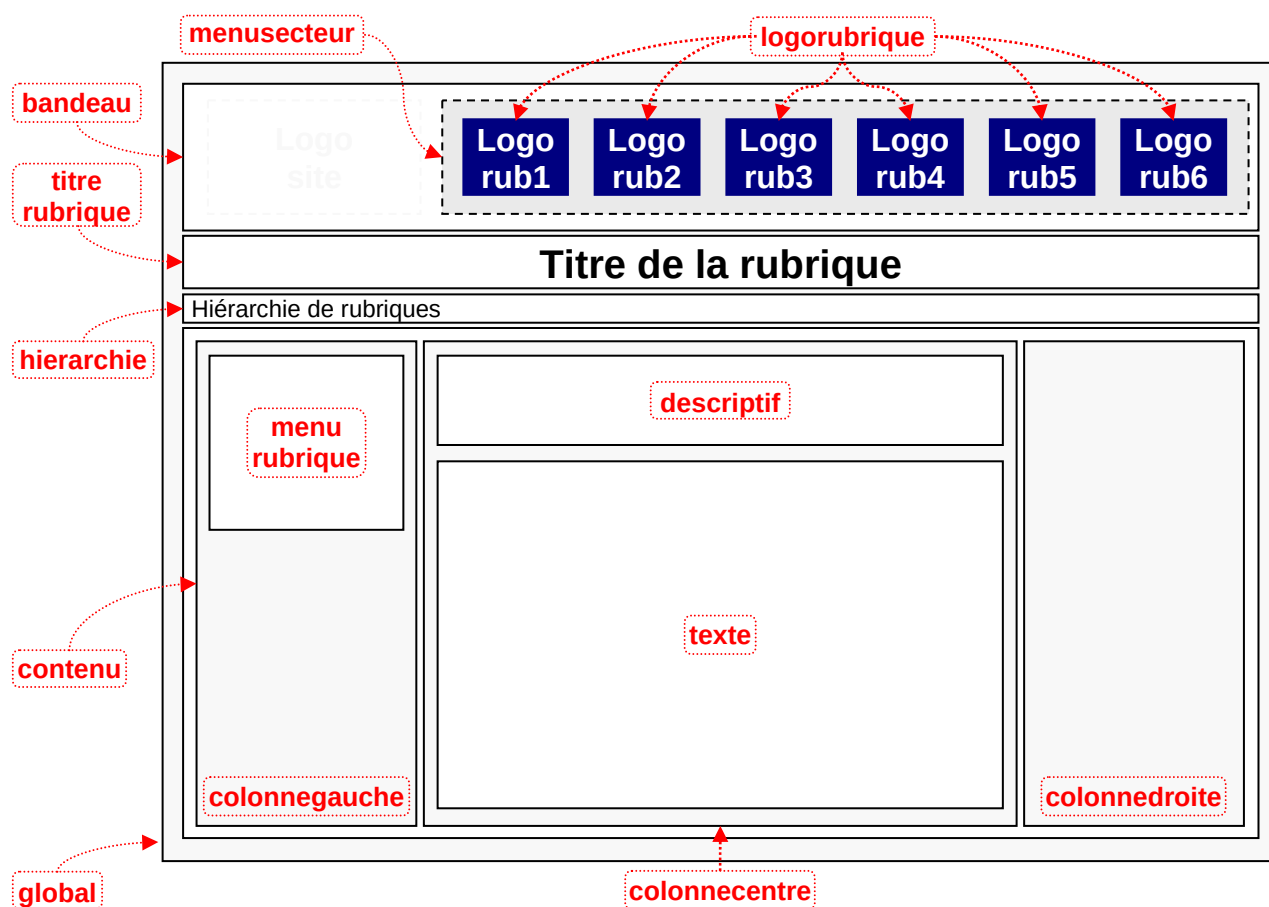
- page 21/62 -



Ainsi, ces fichiers contiennent le code HTML qui permettra de donner au site SPIP la présentation que l'on souhaite. La différence entre un fichier squelette et un fichier HTML classique se situe dans la présence au sein des fichiers squelettes des « boucles » SPIP et de leurs mots-clés associés qui permettent d'insérer au sein du fichier HTML qui sera généré par le moteur de SPIP les données enregistrées dans la base de données (contenu des articles, logos, nom des auteurs, etc.). Il est donc nécessaire, lors de l'écriture de ces fichiers, de bien prendre conscience du fait qu'une boucle peut renvoyer plusieurs éléments (ceci sera détaillé juste après) et donc que le code HTML doit être prévu en conséquence.

Par exemple, l'insertion du titre d'un article, effectuée grâce au mot-clé #TITRE, ne posera aucun problème puisque ce titre sera unique ; par contre, l'affichage de la liste de tous les articles d'un auteur pourra retourner zéro, un ou plusieurs résultats : l'affichage doit donc pouvoir s'adapter à ces différents cas de figure.

6.1. Base commune à tous les squelettes



Avant de rentrer dans les détails d'utilisation de ce qui fait de SPIP un outil puissant (la programmation de boucles), nous devons écrire la base HTML sur laquelle reposeront les squelettes. Il s'agit donc ici de structurer notre page Web en blocs devant chacun accueillir un élément particulier : titre, logo, menu, contenu, etc.

Nous allons placer ces différents blocs selon la disposition présentée page précédente.

Créez un répertoire **squelettes** à la racine du site, et copiez-y le fichier **base.html** dont le contenu est décrit ci-dessous. On constate que ce code ne contient que très peu de choses relatives au contenu ni à la mise en forme, mais pratiquement uniquement une structuration et une hiérarchisation des différentes boîtes (<div>) de la page.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
  <title>
    [NOM DU SITE]
  </title>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body>
<center>
  <div id="GLOBAL">
    <div id="BANDEAU">
      <div id="MENUSECTEUR">
        [ (#LOGO_RUBRIQUE) ]
      </div>
    </div>
    <div id="TITRERUBRIQUE">
      [ (#TITRE) ]
    </div>
    <div id="HIERARCHIE">
      [ (#URL_RUBRIQUE) ]
    </div>
    <div id="CONTENU">
      <div id="COLONNEGAUCHE">
        <div id="MENURUBRIQUES" class="ELEMENTMENU">
          <div class="TITREMENU">Articles de cette rubriques :</div>
          <ul><li>[ (#TITRE) ]</li></ul>
          <div class="TITREMENU">Sous-rubriques :</div>
          <ul><li>[ (#TITRE) ]</li></ul>
          <div class="TITREMENU">Rubriques connexes :</div>
          <ul><li>[ (#TITRE) ]</li></ul>
        </div>
      </div>
      <div id="COLONNECENTRE">
        <div id="DESCRIPTIF">[ (#DESCRIPTIF) ]</div>
        <div id="TEXTE">[ (#TEXTE) ]</div>
      </div>
      <div id="COLONNEDROITE">
        Colonne de droite
      </div>
    </div>
    <div id="FIN">Texte de fin de page</div>
  </div>
</center>
</body>
```

Cependant, même si l'imbrication des boîtes les unes dans les autres est importante, et conditionne le placement final de ces boîtes sur la page, on n'a pour le moment aucune indication du placement réel des boîtes. Nous allons donc maintenant définir dans un premier temps le placement de ces boîtes, ainsi que leur présentation dans un deuxième temps.

6.2. Placement des boîtes grâce à CSS

Avant de modifier quoi que ce soit, voyons quelle serait l'apparence de la page telle qu'elle est définie dans le code précédent. Les éléments de texte en lettres capitales grasses et entre crochets seront remplacés par le véritable contenu dans la version finale : ils ne

servent pour l'instant qu'à nous donner une idée de l'apparence et du placement des différents blocs.



On le voit dans la capture d'écran précédente, aucune structuration n'apparaît : les différentes boîtes sont placées les unes sous les autres dans l'ordre où elles apparaissent dans le code source.

6.3. Présentation des boîtes grâce à CSS

Faisons dans un premier temps apparaître les boîtes. Ajoutons pour cela la ligne suivante dans le champ d'en-tête du fichier HTML :

```
<link href="moto.css" rel="stylesheet" type="text/css">
```

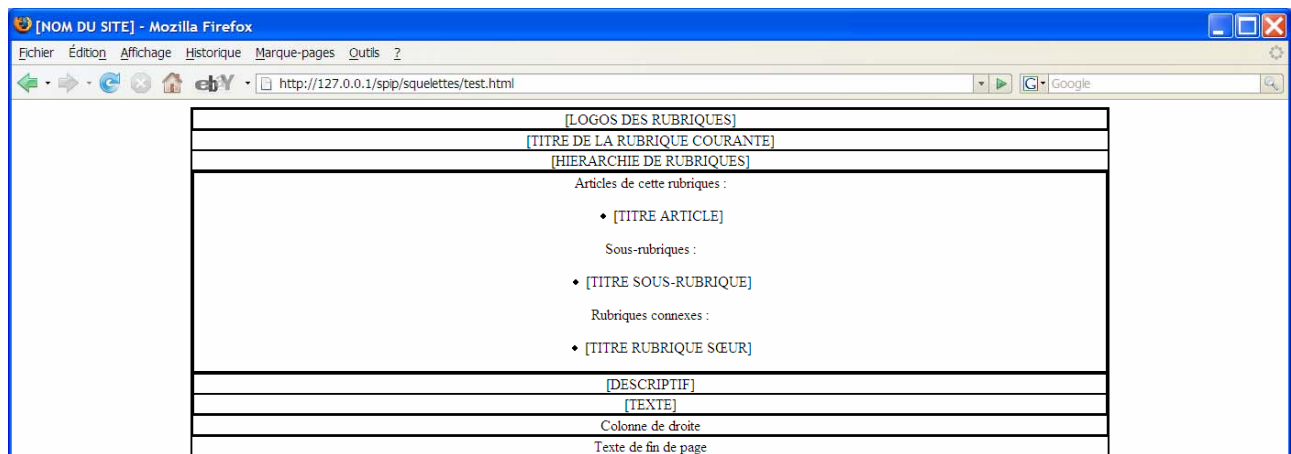
Puis créons le fichier `moto.css` et plaçons-le dans le même répertoire que le fichier `base.html`. Ce fichier contiendra les lignes suivantes :

```
#GLOBAL          { width:1000px;border-style:solid;border-width:1px;}
#BANDEAU,
#MENSECTEUR,
#HIERARCHIE,
#TITRERUBRIQUE,
#CONTENU,
#COLONNEGAUCHE,
#MENURUBRIQUES,
#MENUARTICLES,
#COLONNECENTRE,
#DESCRIPTIF,
#TEXTE,
#COLONNEDROITE,
#FIN             { border-style:solid;border-width:1px;}
```

Nous avons pour l'instant défini la largeur (`width :1000px ;`) de la boîte globale à 1000 pixels, ainsi que le style des bordures de toutes les boîtes en trait plein de largeur 1 pixel (`border-style :solid ;border-width :1px;`). On peut voir ce que cela donne sur la capture d'écran page suivante.

On constate :

- que les boîtes sont bien positionnées les unes sous les autres par défaut, et non les unes à côté des autres ;
- que toutes les boîtes à l'intérieur de la boîte globale héritent de son attribut de largeur : elles occupent donc toute la largeur disponible, soit 1000 pixels ;
- que l'épaisseur des bordures des boîtes imbriquées se cumulent.



Il est temps maintenant de positionner ces boîtes de manière convenable, de façon à obtenir à peu près le schéma de la section 6.1. Modifions la feuille de style CSS pour y introduire les notions de placement :

```
#GLOBAL { width:1000px;border-style:solid;border-width:1px;}
#BANDEAU { width:1000px;height:150px;
            border-style:solid;border-width:1px;}
#FIN { border-style:solid;border-width:1px;}
#COLONNEGAUCHE {width:190px;float:left;padding:5px;
                 border-style:solid;border-width:1px;}
#COLONNECENTRE {width:590px; float:left;padding:5px;
                 border-style:solid;border-width:1px;}
#COLONNEDROITE {width:190px; float:left;padding:5px;
                 border-style:solid;border-width:1px;}
#MENUSECTEUR { position:relative;top:45px;left:300px;width:700px;height:100px;
                border-style:solid;border-width:1px;}
#TITRERUBRIQUE {border-top-width:3px;border-top-style:solid;border-bottom-width:3px;
                 border-bottom-style:solid;}
#HIERARCHIE { border-top-width:1px;border-top-style:solid;
               border-bottom-width:1px;border-bottom-style:solid;
               margin-top:5px;margin-bottom:5px;padding:5px;}
#CONTENU { width:1000px;margin-top:5px;
            border-style:solid;border-width:1px;}
#MENURUBRIQUES {border-style:solid;border-width:1px;}
#DESCRIPTIF { margin:5px;margin-top:0px; padding-left:5px;padding-right:5px;
               border-color:#005500;border-width:3px;border-style:inset;
               background:white;color:#005500;}
#TEXTE { margin:5px;margin-top:0px;z-index:1;padding:5px;height:auto;
          border-color:#005500;border-width:2px;border-style:solid;
          background:white;color:#005500;}
```

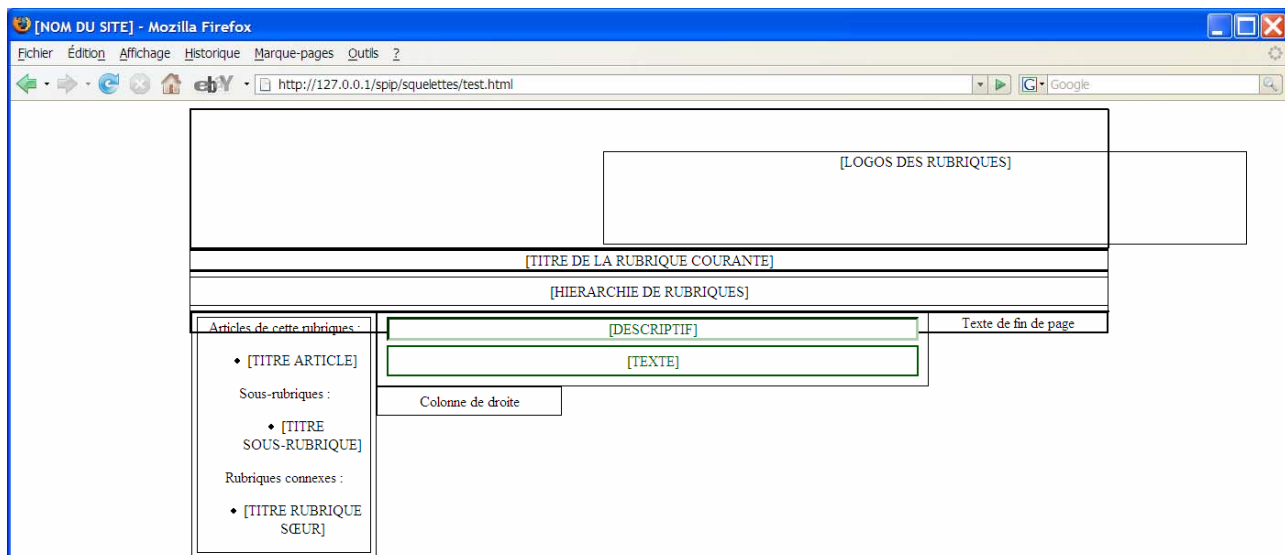
La nouvelle allure de la page est présentée page suivante.

Bien que l'aspect est très proche de ce que l'on souhaite obtenir, quatre problèmes majeurs sont immédiatement visibles :

- la colonne de droite est mal placée ;
- le texte de fin de page n'apparaît pas au bon endroit ;
- la boîte des logos des rubriques est trop décalée vers la droite ;
- la bordure de la boîte globale apparaît bien trop haut (juste après « Texte de fin de page »).

Ces trois problèmes apparemment similaires sont en fait dus à deux causes différentes.

La colonne de droite est trop large et ne rentre pas dans l'espace disponible restant de la boîte globale : la somme des largeurs des boîtes, de l'épaisseur de leurs bordures, des marges extérieures et intérieures aux boîtes doit être inférieure à 1000 pixels, qui est la taille de la boîte englobante.



a. Notions de placement de boîtes : positions relative, absolue, fixe, et flottante

Par défaut, les navigateurs affichent les boîtes issues du document html dans l'ordre du flux normal, le navigateur parcourant la page de code HTML du début à la fin et retranscrivant son contenu au fur et à mesure des balises rencontrées.

En position relative (`position:relative;`), un élément peut être décalé verticalement et/ou horizontalement par rapport à sa place « normale ».

La position float retire une boîte du flux normal pour la placer le plus à droite (`float:right;`) ou le plus gauche (`float:left;`) possible dans son conteneur. Le contenu suivant cette boîte flottante s'écoule le long de celle-ci, dans l'espace laissé libre. Le paramètre `float:left;` indique donc que la boîte est une boîte flottante qui sera placée dans le flux de texte en partant du plus à gauche possible : ainsi, lorsque plusieurs boîtes correspondant aux différentes rubriques seront créées, elles seront toutes flottantes à gauche et se positionneront de manière naturelle de gauche à droite. Ainsi, le positionnement des trois colonnes (qui n'est pas parfait pour l'instant, nous allons y revenir), est effectué dans l'ordre de leur apparition en fonction de la place disponible.

Les positions absolue (`position:absolute;`) et fixe (`position:fixed;`) permettent de placer une boîte par rapport aux limites de la zone d'affichage ou du conteneur. Le positionnement absolu « retire » totalement du flux le contenu concerné : sa position est déterminée par référence aux limites du conteneur. Celui-ci peut-être une boîte elle-même positionnée (position relative ou absolue) ou le bloc conteneur initial, à défaut de boîte positionnée, c'est à dire en pratique le plus souvent la fenêtre du navigateur.

Comme dans un positionnement absolu, le contenu concerné est retiré totalement du flux. Mais il est cette fois positionné uniquement par rapport aux limites de la zone de visualisation, autrement dit la fenêtre du navigateur. Le défilement de la page n'a aucun effet sur un contenu en position fixe.

b. Correction de tailles de boîtes

La largeur cumulée des boîtes COLONNEGAUCHE, COLONNECENTRE, COLONNEDROITE, additionné de leurs marges et bordures, ainsi que des bordures de la boîte CONTENU qui contient les trois précédentes doit faire 1000 pixels.

Or, nous trouvons respectivement :

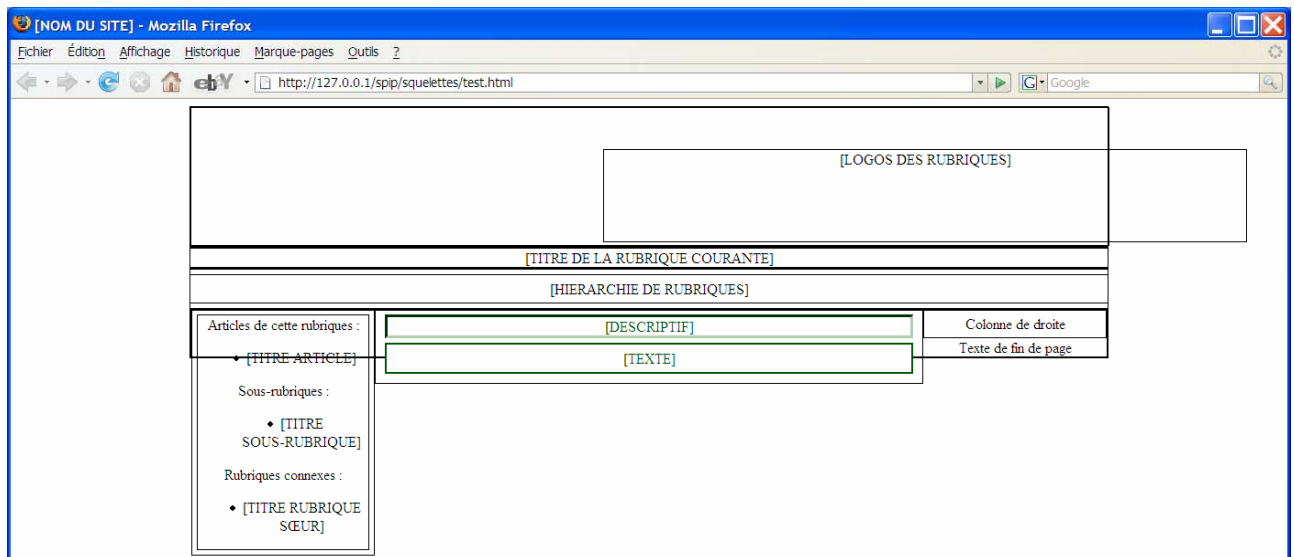
- COLONNEGAUCHE : $190+2 \times 5+2 \times 1 = 202$ pixels
- COLONNEDROITE : $190+2 \times 5+2 \times 1 = 202$ pixels

– COLONNECENTRE : $590+2\times 5+2\times 1 = 602$ pixels

Soit un total de 1006 pixels, donc 6 de trop.

De plus, la boîte CONTENU étant elle-même incluse dans la boîte globale, du fait de sa bordure elle est trop large de 2 pixels également. Donc, en réduisant la largeur de la boîte CONTENU de 2 pixels, et celles des trois colonnes de 8 pixels, on doit pouvoir corriger ce problème. En modifiant les dimensions en conséquence :

```
#CONTENU {      width:998px;margin-top:5px;
                  border-style:solid;border-width:1px;}
#COLONNEGAUCHE {width:188px;float:left;padding:5px;
                  border-style:solid;border-width:1px;}
#COLONNECENTRE {width:586px;float:left;padding:5px;
                  border-style:solid;border-width:1px;}
#COLONNEDROITE {width:188px;float:left;padding:5px;
                  border-style:solid;border-width:1px;}
```



Nous obtenons le résultat escompté (cf capture d'écran page précédente).

Pour la même raison que précédemment, vous pourrez ramener la taille de la boîte BANDEAU à 998 pixels seulement.

Il nous reste à régler le problème de la boîte contenant les logos, et celle du texte de fin de page. Le premier cas est délicat à détecter, et le second est lié à la gestion du placement des boîtes.

c. Correction des problèmes d'alignement

Pourquoi la boîte est-elle MENUSECTEUR décalée ? En fait, le bord de cette boîte est centré par rapport à la boîte BANDEAU, bien qu'aucun attribut des boîtes ne précise le mode d'alignement. En fait, la boîte MENUSECTEUR hérite des propriétés de la boîte BANDEAU, qui hérite elle-même des propriétés de la boîte GLOBAL ... qui hérite des propriétés de l'élément HTML dans lequel elle est déclarée.

Vérifions le code source réduit aux seuls éléments qui nous intéressent :

```
<center>
  <div id="GLOBAL">
    <div id="BANDEAU">
      <div id="MENUSECTEUR">[ (#LOGO_RUBRIQUE)]</div>
    </div>
    ...
  </div>
</center>
```

Toutes ces boites héritent en fait de la propriété de <center> qui consiste à ... centrer le texte et tous les éléments qui dépendent de l'alignement. Nous allons donc ajouter une propriété d'alignement à la boite GLOBAL dans notre feuille de style : toutes les boites héritant celle-ci, nous allons pouvoir de ce fait corriger ce défaut. Nous en profiterons pour réduire un peu la taille de la boite MENUSECTEUR à 694 pixels :

```
#GLOBAL {      width:1000px;border-style:solid;border-width:1px;
                text-align :justify;}
#MENUSECTEUR { position:relative;top:45px;left:300px;width:694px;height:100px;
                border-style:solid;border-width:1px;}
```

Le texte sera donc par défaut justifié, donc aligné à gauche, puis à droite. La boite MENUSECTEUR sera donc alignée à gauche également, puis décalée de 300 pixels vers la droite comme le précise l'attribut `left : 300px`;

d. Réinitialisation du positionnement des boites

Il ne nous reste plus qu'à placer la boite FIN qui contient le texte « Texte de fin de page ». Cette boite FIN est placée, dans notre fichier source HTML, après la boite CONTENU. Un court exemple valant mieux qu'un long discours, faites l'essai en forçant la largeur de la boite FIN à 1000 pixels :

```
#FIN {          border-style:solid;border-width:1px; width:1000px;
                border-color:red;background-color:yellow;}
```

Puis 100 pixels :

```
#FIN {          border-style:solid;border-width:1px;width:100px;
                border-color:red;background-color:yellow;}
```

La boite est jaune avec des bordures rouges afin de bien voir l'influence de la largeur. On constate que la boite s'étend bien sur la largeur désirée, mais que le **texte se positionne dès qu'il le peut dans un espace non couvert par une autre boite**. Refaites l'essai avec une largeur de 400 pixels, mais essayez de deviner où sera placé le texte.

Un autre effet secondaire de ce comportement est que comme la boite FIN est la dernière boite incluse dans la boite GLOBAL, la bordure inférieure de la boite GLOBAL est dessinée juste après celle de la boite fin, même si elle doit pour cela recouvrir d'autres boites.

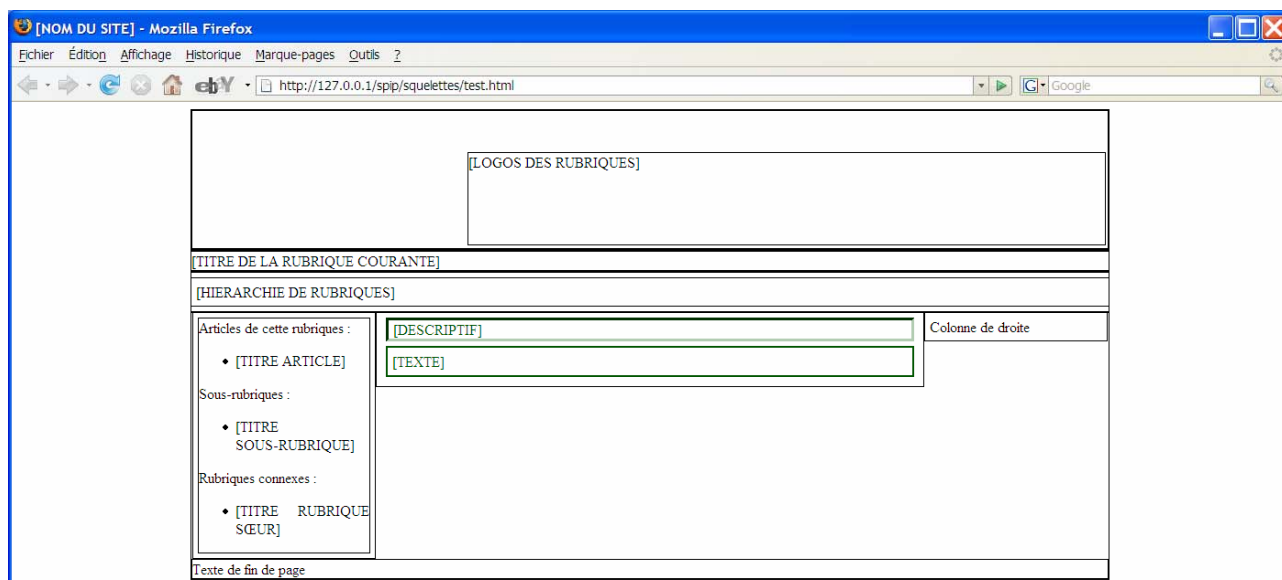
La solution pour résoudre ce problème est d'utiliser la propriété `clear` qui consiste justement à préciser que l'on souhaite réinitialiser le placement des boites. Consultez les documentations sur CSS pour plus de précision, mais sachez que la propriété `clear : both`; force la boite à se placer sous la boite qui la précède :

```
#FIN {          border-style:solid;border-width:1px;width:400px;clear:both;
                border-color:red;background-color:yellow; }
```

Pour la suite, vous pouvez donner à la boite FIN une largeur de 998 pixels, puis supprimer les attributs de couleur :

```
#FIN {          border-style:solid;border-width:1px;width:998px;clear:both;}
```

La capture d'écran page suivante présente l'allure de la page en l'état actuel des choses.



6.4. Décorations et gestion des images

Afin de donner au site une apparence un peu plus attrayante, nous pouvons agrémenter le squelette d'images, statiques ou dynamiques.

Les images statiques seront définies dans le fichier source HTML, ou dans la feuille de style associée. Les images dynamiques seront ajoutées via l'interface d'administration de SPIP, et intégrées à la page via les boucles SPIP, ce dont nous parlerons plus tard. Nous ne nous intéressons ici qu'aux images statiques.

6.4.1. Image du bandeau de titre

Pour ajouter une image à ce bandeau, nous avons le choix entre deux solutions :

- Ajouter une boîte contenant le logo du site ;
- Définir cette image comme paramètre de la boîte BANDEAU.

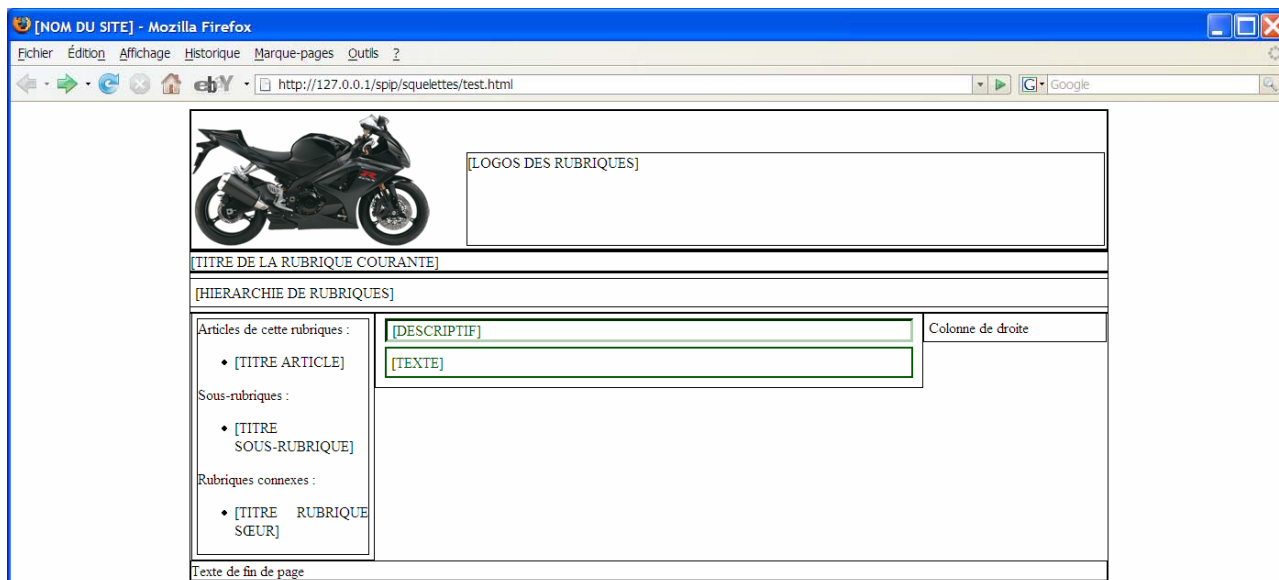
a. Utilisation du paramètre **background-image**

Nous allons dans un premier temps définir une image pour le bandeau-titre.

```
#BANDEAU { width:998px;height:150px;
            border-style:solid;border-width:1px;
            background-image:url(IMGES/bandeau moto.png);}
```

Le paramètre **background-image** permet, comme son nom l'indique, d'appliquer une image de fond à une boîte, selon le principe du « fond d'écran ».

Dans cet exemple, l'image `bandeau-moto.png` fait exactement la taille de la boîte, ce qui permet de s'affranchir de toutes les options de positionnement. Vous pouvez consulter les documentations sur le CSS pour en apprendre plus sur les paramètres **background-attachment**, **background-repeat**, et **background-position**. Nous ne les aborderons pas dans ce tutoriel.



b. Insertion d'une image à l'aide de

Exercice : Il est possible d'insérer une image dans la boîte BANDEAU à l'aide de la commande vue en cours, éventuellement en créant une nouvelle boîte LOGOSITE qui sera utilisée comme container à cette image. Proposez une solution permettant d'y arriver.

6.4.2. Logos des secteurs

Nous souhaitons faire apparaître les logos des secteurs (rubriques de premier niveau) dans la boîte MENUSECTEUR. Ces logos seront cliquables, et conduiront aux rubriques concernées. Pour corser un peu la chose, nous souhaitons que ces logos apparaissent dans un cadre graphique représentant un roue de moto.

Exemple :



L'image « a » est l'image de la rubrique, enregistrée via l'interface d'administration, est référencée dans la base de données. L'image « b » est une image utilisée via le squelette pour encadrer la précédente : son centre est donc transparent. Pour ce faire, nous allons exploiter la gestion de la transparence (couche alpha) des images GIF, ainsi que la notion de « profondeur » des boîtes CSS via le paramètre z-index.

Modifions le code source HTML de la partie BANDEAU en conséquence :

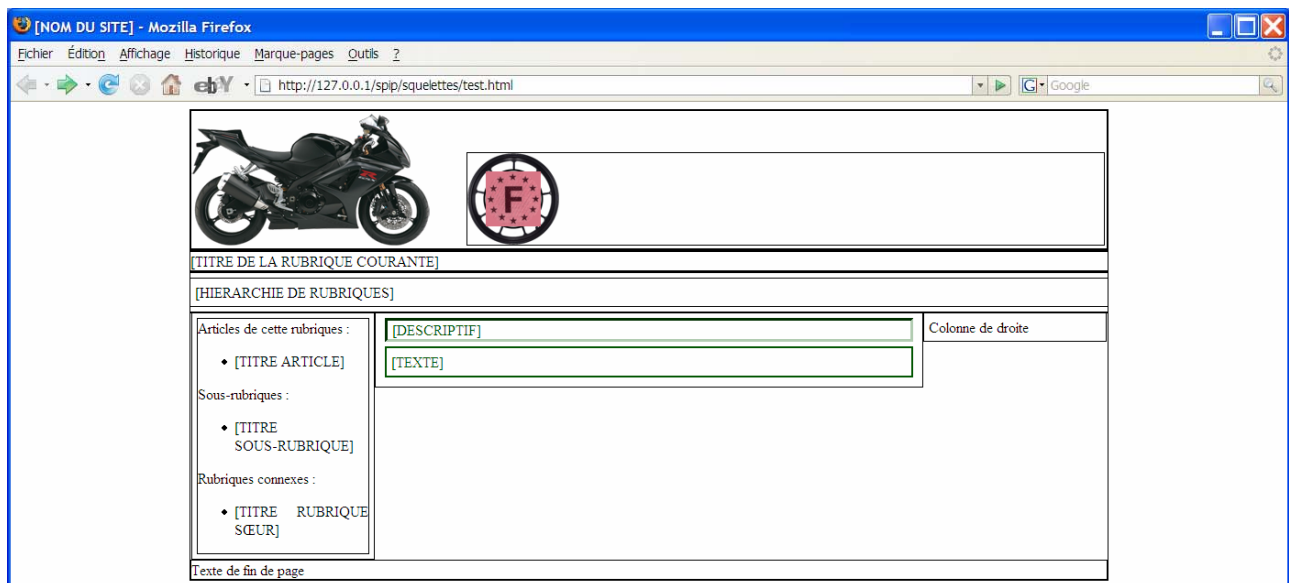
```
<div id="BANDEAU">
  <div id="MENUSECTEUR">
    <div class="CADRELOGO">
      <a href="[(#URL_RUBRIQUE)]">
        <div class="LOGORUBRIQUE"></div>
      </a>
    </div>
  </div>
</div>
```

Nous avons inséré dans la boîte MENUSECTEUR deux boîtes imbriquées CADRELOGO et LOGORUBRIQUE, lesquelles vont contenir respectivement le cadre-image en forme de roue, et le logo de la rubrique. La feuille de style recevra les descriptions suivantes, dans un premier sans gérer la profondeur des éléments :

```
.CADRELOGO a {      position:relative;top:0px;left:0px;width:100px;height:100px;
                     background-image:url(IMAGES/roue_i.gif);float:left;}
.LOGORUBRIQUE img{  position:relative;top:20px;left:20px;width:60px;height:60px;
                     border-style:none;}
```

Nous utilisons ici deux modes de positionnement relatifs. Le premier, pour CADRELOGO, avec un décalage de 0 pixel, est nécessaire pour autoriser le positionnement de LOGORUBRIQUE à l'intérieur de celui-ci. Les décalages de 20 pixels pour le second permettent de positionner exactement la boîte LOGORUBRIQUE de 60 pixels de côté au centre de la boîte CADRELOGO qui elle fait 100 pixels de côté. Pour ces éléments-ci, nous utilisons ici des classes (class="...") et non des identifiants (id="...") parce que nous aurons plusieurs boîtes CADRELOGO identiques, contenant chacune une boîte LOGORUBRIQUE, alors que les identifiants sont censés désigner des éléments uniques. Les cadres contenant les logos sont définis comme des boîtes flottantes (float:left;) et seront donc positionnés dans l'ordre de leur apparition dans le flux HTML.

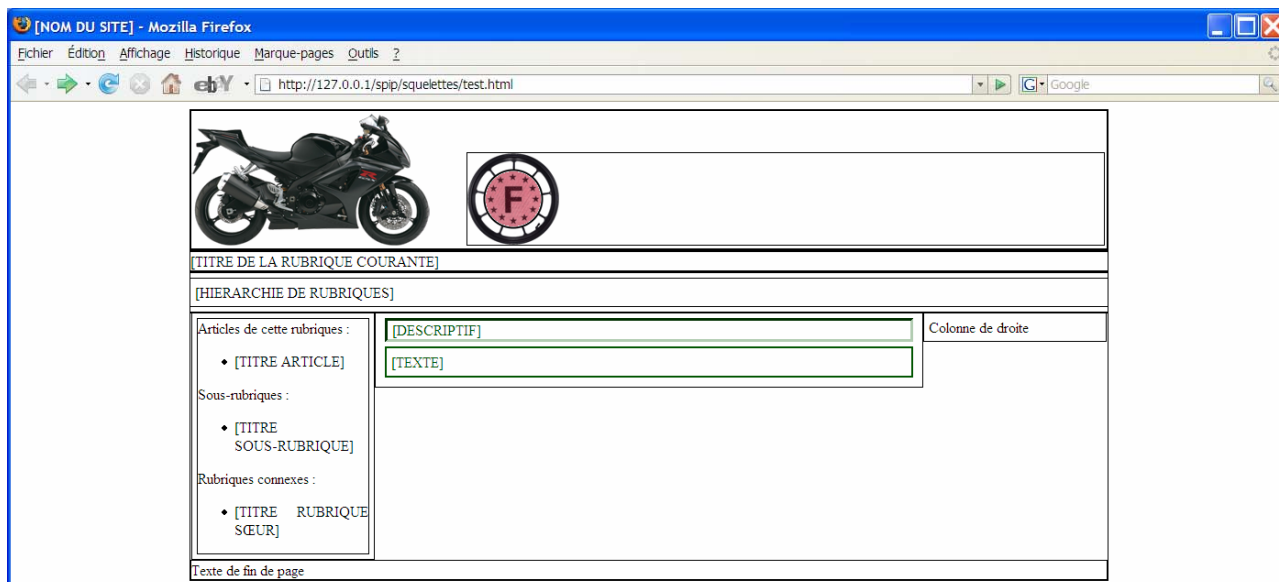
Le résultat obtenu est le suivant :



Nous constatons que le logo de la rubrique apparaît au-dessus du cadre-image, ce qui est le comportement normal des éléments CSS en l'absence de propriété z-index. Modifions donc la feuille de style de façon à fournir la propriété z-index:0; à la boîte CADRELOGO et z-index:-1; à la boîte LOGORUBRIQUE. Cette dernière apparaîtra alors sous la précédente. Le code CSS devient :

```
.CADRELOGO a {      position:relative;top:0px;left:0px;width:100px;height:100px;
                     background-image:url(IMAGES/roue_i.gif);float:left;z-index:0;}
.LOGORUBRIQUE img{  position:relative;top:20px;left:20px;width:60px;height:60px;
                     border-style:none;z-index:-1;}
```

Le résultat obtenu est présenté page suivante.



Afin d'animer un peu ce menu « secteurs », nous allons faire tourner la roue qui encadre le logo de la rubrique lorsque le curseur de la souris se positionnera dessus. Pour ce faire, nous exploiterons la propriété CSS `a:hover`.

```
.CADRELOGO a { position:relative;top:0px;left:0px;width:100px;height:100px;
background-image:url(IMAGES/roue_i.gif);float:left;z-index:0;}
.CADRELOGO a:hover { background-image:url(IMAGES/roue_m.gif);}
.LOGORUBRIQUE img{ position:relative;top:20px;left:20px;width:60px;height:60px;
border-style:none; z-index:-1;}
```

La propriété `a:hover` définit donc une image de fond différente pour `CADRELOGO` lorsque le curseur de la souris est présent sur le lien (donc sur l'image). Cette image `roue_m.gif` est une image animée donnant l'impression de mouvement, conçue à cette intention à partir de `roue_i.gif`. Toutes les autres propriétés de `.CADRELOGO a:hover` sont héritées de `.CADRELOGO a` (même taille, même mode de placement, etc.). Constatez à l'écran le comportement du logo lorsque le curseur de la souris passe dessus.

6.4.3. Les menus de navigation

Nous allons apporter la dernière correction graphique majeure à notre site en donnant au menu de navigation (colonne de gauche) un aspect un peu plus agréable. Ici aussi nous exploiterons les propriétés `a:hover`, entre autres.

Modifions le fichier source HTML de manière à créer les liens. La partie `COLONNEGAUCHE` de ce fichier devient :

```
<div id="COLONNEGAUCHE">
  <div id="MENURUBRIQUES" class="ELEMENTMENU">
    <div class="TITREMENU">Articles de cette rubrique :</div>
    <ul><li><a href="[(#URL_ARTICLE)]"> [(#TITRE)]</a></li></ul>
    <div class="TITREMENU">Sous-rubriques :</div>
    <ul><li><a href="[(#URL_RUBRIQUE)]"> [(#TITRE)]</a></li></ul>
    <div class="TITREMENU">Rubriques connexes :</div>
    <ul><li><a href="[(#URL_RUBRIQUE)]"> [(#TITRE)]</a></li></ul>
  </div>
</div>
```

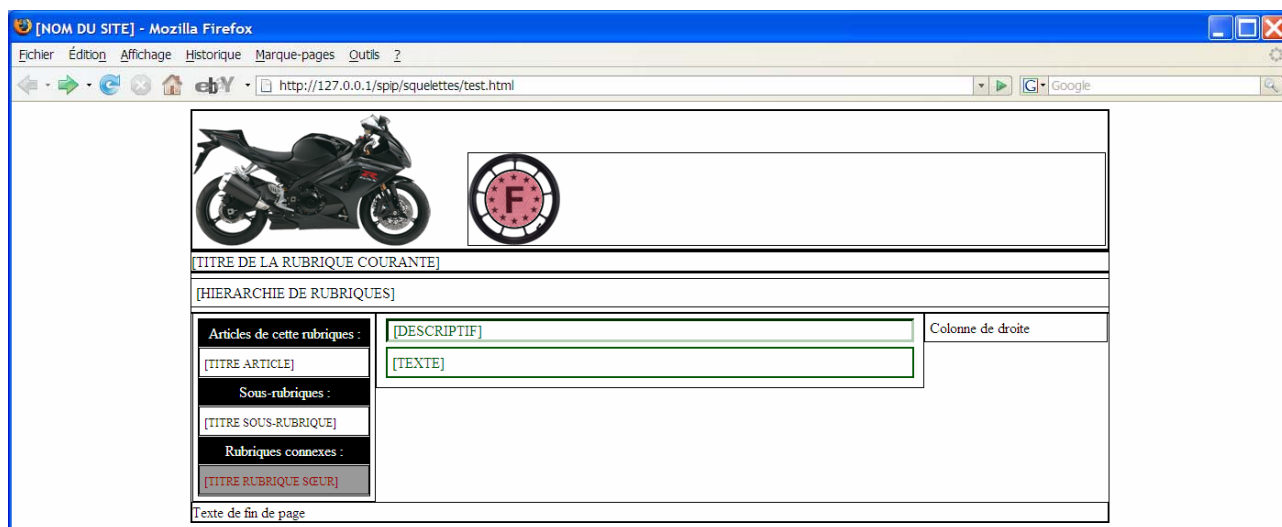
Il nous faut donc définir les classes `TITREMENU` et `ELEMENTMENU`. Ce ne sont pas des identifiants pour les mêmes raisons que précédemment : de nombreux éléments vont partager cette même définition de style.


```

.TITREMENU {
background-color:black;color:white;
text-align:center;padding:5px;margin-bottom:1px;}
.ELEMENTMENU {
margin-bottom:0px;padding:0px;}
.ELEMENTMENU:hover {
background-color:#999999;}
.ELEMENTMENU ul {
list-style:none;margin:0px;padding:5px;
border-style:solid;border-width:1px;margin-bottom:1px;}
.ELEMENTMENU ul:hover {
background-color:#999999;}
.ELEMENTMENU a {
text-decoration:none;color:black;font-size:8pt;}
.ELEMENTMENU a:hover {
color:#990000;}
.ELEMENTMENU a:visited {
color:#550000;}

```

Le menu de gauche ressemble maintenant à ça :



6.4.4. Résumé

Afin de délester la page de toutes ses bordures inutiles et d'aérer la présentation, nous n'allons conserver que le strict minimum. La feuille de style finale – à ce stade - devient :

```

#GLOBAL { width:1000px;border-style:solid;border-width:1px;text-align:justify;}
#BANDEAU { width:998px;height:150px;background-image:url(IMGES/bandeau_moto.png);}
#CONTENU { width:998px;margin-top:5px;}
#FIN { border-style:solid;border-width:0px;border-top-width:1px;width:1000px;
clear:both;}

#MENUSECTEUR { position:relative;top:45px;left:300px;width:694px;height:100px;}
.CADRELOGO a { position:relative;top:0px;left:0px;width:100px;height:100px;float:left;
background-image:url(IMGES/roue_i.gif);z-index:0;}
.CADRELOGO a:hover { background-image:url(IMGES/roue_m.gif);}
.LOGORUBRIQUE img{ position:relative;top:20px;left:20px;width:60px;height:60px;
border-style:none;z-index:-1;}

#TITRERUBRIQUE {border-top-width:3px;border-top-style:solid;
border-bottom-width:3px;border-bottom-style:solid;padding:5px;}
#HIERARCHIE { border-top-width:1px;border-top-style:solid;
border-bottom-width:1px;border-bottom-style:solid;
margin-top:5px;margin-bottom:5px;padding:5px;}

#COLONNEGAUCHE {width:188px;height:auto;float:left;padding:5px;}
#COLONNECENTRE {width:586px;height:auto;float:left;padding:5px;}
#COLONNEDROITE {width:188px;height:auto;float:left;padding:5px;
border-style:solid;border-width:1px;}

```

```
#MENURUBRIQUES { }
.TITREMENU { background-color:black;color:white;text-align:center;padding:5px;margin-bottom:1px;}
.ELEMENTMENU { margin-bottom:0px;padding:0px;}
.ELEMENTMENU:hover { background-color:#999999;}
.ELEMENTMENU ul { list-style:none;margin:0px;padding:5px;border-style:solid;border-width:1px;margin-bottom:1px;}
.ELEMENTMENU ul:hover { background-color:#999999;}
.ELEMENTMENU a { text-decoration:none;color:black;font-size:8pt;}
.ELEMENTMENU a:hover { color:#990000;}
.ELEMENTMENU a:visited {color:#550000;}

#DESCRIPTIF {margin:5px;margin-top:0px; padding-left:5px;padding-right:5px;
border-color:#005500;border-width:3px;border-style:inset;
background:white;color:#005500;}
#TEXTE { margin:5px;margin-top:0px;z-index:1;padding:5px;
border-color:#005500;border-width:2px;border-style:solid;
background:white;color:#005500; }
```

Pour finir nous ajouterons la ligne suivante dans la partie d'en-tête du fichier HTML :

```
<link rel="shortcut icon" href="IMAGES/tiny_moto.gif">
```

Cette ligne définit uniquement l'icône de la barre d'adresse du site, celle qui est stockée dans les signets en même temps que l'adresse.

Le code source HTML complet est le suivant :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>[NOM DU SITE]</title>
<link href="moto.css" rel="stylesheet" type="text/css">
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<link rel="shortcut icon" href="IMAGES/tiny_moto.gif">
</head>
<body>
<center>
<div id="GLOBAL">
<div id="BANDEAU">
<div id="MENSECTEUR">
<div class="CADRELOGO">
<a href="[(#URL_RUBRIQUE)]">
<div class="LOGORUBRIQUE">

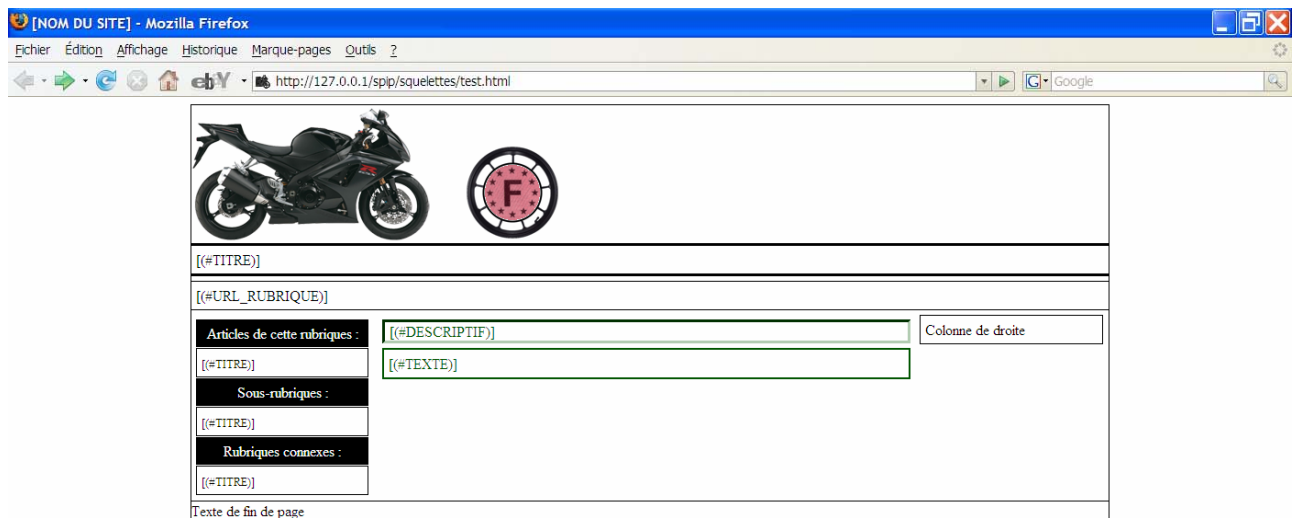
</div>
</a>
</div>
</div>
</div>
<div id="TITRERUBRIQUE">[(#URL_TITRE)]</div>
<div id="HIERARCHIE">[(#URL_RUBRIQUE)]</div>
<div id="CONTENU">
<div id="COLONNEGAUCHE">
<div id="MENURUBRIQUES" class="ELEMENTMENU">
<div class="TITREMENU">Articles de cette rubrique :</div>
<ul><li><a href="[(#URL_ARTICLE)]">[(#TITRE)]</a></li></ul>
<div class="TITREMENU">Sous-rubriques :</div>
<ul><li>
<a href="[(#URL_RUBRIQUE)]">[(#TITRE)]</a>
</li></ul>
<div class="TITREMENU">Rubriques connexes :</div>
<ul><li>
<a href="[(#URL_RUBRIQUE)]">[(#TITRE)]</a>
</li></ul>
</div>
</div>
```

```

        <div id="COLONNECENTRE">
            <div id="DESCRIPTIF">[(#DESCRIPTIF)]</div>
            <div id="TEXTE">[(#TEXTE)]</div>
        </div>
        <div id="COLONNEDROITE"> Colonne de droite </div>
    </div>
    <div id="FIN">Texte de fin de page</div>
</div>
</center>
</body>
</html>

```

Et notre site ressemble maintenant à ceci :



6.4.5. Et ensuite ?

La présentation de notre site étant maintenant à peu près définie, il ne nous reste plus qu'à remplir la page avec le contenu de la base de données. C'est ici qu'interviennent les « boucles SPIP » qui font l'objet de la section suivante.

7.Principe des boucles SPIP

Il est impossible de vous décrire ici toutes les fonctionnalités offertes par SPIP : nous vous encourageons vivement à consulter la documentation en ligne très complète pour en apprendre davantage. Nous allons ici simplement aborder la notion de boucles au travers de quelques exemples simples et concrets mettant en œuvre ce qu'il est indispensable de connaître pour débiter dans l'utilisation avancée de SPIP. Nous continuerons naturellement à agrémenter notre feuille de style de façon à améliorer la présentation du site.

7.1. Création des fichiers squelettes

Tous les fichiers squelettes doivent se situer dans le répertoire « squelettes » situé à la racine du site. Nous allons commencer par l'écriture du fichier rubrique.html. À ce fichier, ainsi qu'à tous les autres fichiers squelettes, nous allons associer une feuille de style moto.css.

Avant de récupérer le contenu du fichier base.html, nous allons créer un nouveau fichier rubrique.html. Tous les fichiers squelettes étant en premier lieu des fichiers HTML, ils doivent contenir toutes les informations indispensables à tout fichier HTML. Nous lui ajouterons dans un premier temps quelques mots-clefs de SPIP

7.1.1. Mots-clefs SPIP

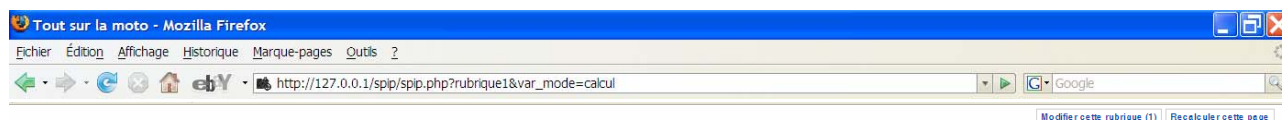
Les mots-clefs de SPIP commencent tous par le symbole # et sont tous écrits en lettres capitales. Certains mots-clefs sont « globaux », en ce sens qu'il ne dépendent d'aucune boucle. C'est le cas des mots-clefs utilisés ci-après. D'autres par contre dépendent de la boucle dans laquelle ils se situent : ils correspondent au(x) résultat(s) retourné(s) par la boucle.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>#NOM_SITE_SPIP</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    <link href="#URL_SITE_SPIP/squelettes/moto.css" rel="stylesheet" type="text/css">
    <link rel="shortcut icon" href="#URL_SITE_SPIP/squelettes/IMAGES/tiny_moto.gif">
  </head>
  <body>
    ...
  </body>
</html>
```

Le code ci-dessus montre la base de notre fichier de travail : c'est un fichier HTML pratiquement vide, contenant la seule en-tête enrichie de quelques mots-clefs SPIP :

- Un mot-clef indiquant le nom du site SPIP : **#NOM_SITE_SPIP**
- Un mot-clef indiquant l'adresse du site SPIP : **#URL_SITE_SPIP**

Voici ce que ça donne :



Nous pouvons constater que sur la barre de titre #NOM_SITE_SPIP a été remplacé par « Tout sur la moto ». De même, les adresses de la feuille de style et de l'icône de raccourci ont bien été complétées avec l'adresse de base du site (http://127.0.0.1/spip/squelettes/moto.css et http://127.0.0.1/spip/squelettes/IMAGES/tiny_moto.gif, respectivement). Vous pouvez vérifier le code source affiché par votre navigateur pour le comparer avec celui que vous avez écrit.

Mais au-delà des mots-clefs, SPIP ne dévoile toute sa puissance qu'avec l'utilisation des boucles. Ce sont elles qui offrent à ce logiciel la capacité de créer un site dynamique et sur mesure à tout un chacun.

7.1.2. Les boucles SPIP – syntaxe

Voici la syntaxe commune à toutes les boucles SPIP :

```
<BOUCLEnom(type_de_boucle){paramètre}>
...
</BOUCLEnom>
```

- **nom** est le nom que vous donnez à la boucle : il peut être constitué de n'importe quel caractère alphanumérique. Ici le nom est **_rubrique_principale**. Comme on peut s'y attendre, cette boucle fait donc référence à la rubrique principale à laquelle la page affichée fait référence.
- **type_de_boucle** fait référence au type de boucle (quel type d'information cette boucle va traiter). Il y a un nombre fixe de types possibles. Pour l'instant, seuls deux types nous intéressent : ARTICLES et RUBRIQUES. Ici, on a donc une boucle de type

RUBRIQUES, ce qui signifie que tous les mots clefs utilisés à l'intérieur de cette boucle feront référence à la rubrique sélectionnée.

- **paramètre** est un critère de sélection d'un ou plusieurs éléments parmi l'ensemble des éléments possibles. Il est possible de cumuler plusieurs paramètres en les juxtaposant : {paramètre1}{paramètre2}{paramètre3}...

7.1.3. Fonctionnement d'une boucle

Une boucle SPIP retourne **tous les éléments qui répondent aux critères de la boucle**, et **recopie autant de fois que nécessaire le code HTML** inscrit à l'intérieur des balises de boucle **en remplaçant les mots-clefs SPIP par les valeurs correspondantes** à l'élément retourné.

Reprenons l'exemple de boucle extrait du code ci-dessus, en laissant le body vide. Ajoutons deux mots-clefs supplémentaires dans le corps du fichier HTML, et encadrons l'ensemble du code HTML par une boucle de type RUBRIQUES, avec comme seul paramètre id_rubrique :

```
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>#NOM_SITE_SPIP</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    <link href="#URL_SITE_SPIP/squelettes/moto.css" rel="stylesheet" type="text/css">
    <link rel="shortcut icon" href="#URL_SITE_SPIP/squelettes/IMAGES/tiny_moto.gif">
  </head>
  <body>
    #URL_SITE_SPIP/#URL_RUBRIQUE <br />
    #TITRE
  </body>
</html>
</BOUCLE_rubrique_principale>
```

Cette boucle contient les éléments suivants :

- Une boucle SPIP :
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
...
</BOUCLE_rubrique_principale>
- Deux mots-clefs globaux identiques au cas précédent :
 - o **#URL_SITE_SPIP**
 - o **#NOM_SITE_SPIP**
- Deux mots-clefs spécifiques à la boucle _rubrique_principale :
 - o **#URL_RUBRIQUE**
 - o **#TITRE**

Cette boucle va de permettre de sélectionner la rubrique (type de boucle : RUBRIQUES) dont l'identifiant est précisé par le paramètre id_rubrique, et va afficher l'URL de la rubrique correspondant au critère id_rubrique (mot-clef #URL_RUBRIQUE) ainsi que le titre de la rubrique (mot-clef #TITRE).

Notons que les mots-clefs #URL_SITE_SPIP ou #NOM_SITE_SPIP ne dépendent pas de la boucle et pourraient tout à fait être sortis de la boucle en offrant la même fonctionnalité (cf cas précédent). Le mot-clef #TITRE fait par contre directement référence au titre de la rubrique sélectionnée par le paramètre id_rubrique.

Avant d'aller plus loin, il nous faut maintenant comprendre l'intérêt de cette boucle « _rubrique_principale », ainsi que la manière dont le paramètre id_rubrique sera exploité.

L'accès à une page d'un site SPIP se fait via une adresse du type suivant :

http://127.0.0.1/spip/spip.php?rubrique1

La partie de l'adresse en caractères gras constitue l'adresse de base du site. La partie suivante (**spip.php?rubrique1**) fait référence à la page demandée : il s'agit d'une rubrique dont l'identifiant est « 1 ». C'est cet identifiant qui est transmis au moteur de SPIP, lequel va remplacer dans le code HTML la variable `id_rubrique` par la valeur « 1 ». En résumé, cette URL fait apparaître plusieurs éléments :

- **http://127.0.0.1/spip/** correspond à `#URL_SITE_SPIP`
- **spip.php?rubrique1** correspond à `#URL_RUBRIQUE`
- **1** correspond à `id_rubrique`

On peut imaginer le même type d'adresse pour un article au lieu d'une rubrique. L'adresse serait **http://127.0.0.1/spip/spip.php?article1** et :

- **http://127.0.0.1/spip/** correspond à `#URL_SITE_SPIP`
- **spip.php?article1** correspond à `#URL_ARTICLE`
- **1** correspond à `id_article`

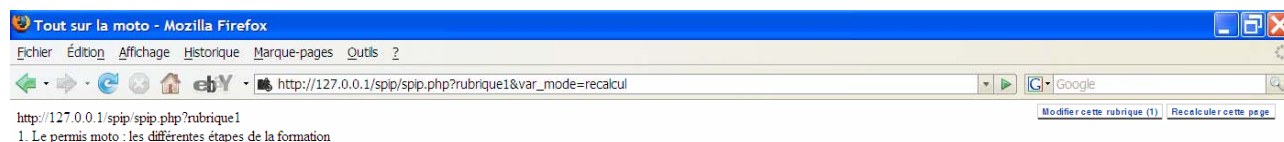
Ainsi, dans tout le code de la boucle précédente, les mots-clefs relatifs à la rubrique sélectionnée seront ceux de la rubrique « 1 ». La boucle précédente peut donc être comprise comme suit :

```
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique=1}>  
  http://127.0.0.1/spip/spip.php?rubrique1  
  TITRE DE LA RUBRIQUE 1  
</BOUCLE_rubrique_principale>
```

Seuls les éléments à l'intérieur de la boucle (en caractères gras) seront donc affichés par la page Web. Ici, le titre de la rubrique 1 est «**1. Le permis moto : les différentes étapes de la formation**».

Dans cet exemple simple, le paramètre `{id_rubrique}` ne sélectionnera donc qu'une seule rubrique (celle demandée dans l'adresse), ce qui est bien l'effet voulu : on veut afficher la page correspondant à cette rubrique uniquement. Par contre, nous verrons ci-après comment sélectionner plusieurs rubriques ou plusieurs articles, par exemple pour afficher la liste de toutes les rubriques ou de tous les articles contenus dans la rubrique courante.

Le résultat est présenté dans la capture d'écran ci-dessous :



7.2. Exemple : fichier *rubrique.html*

7.2.1. La boucle RUBRIQUES principale

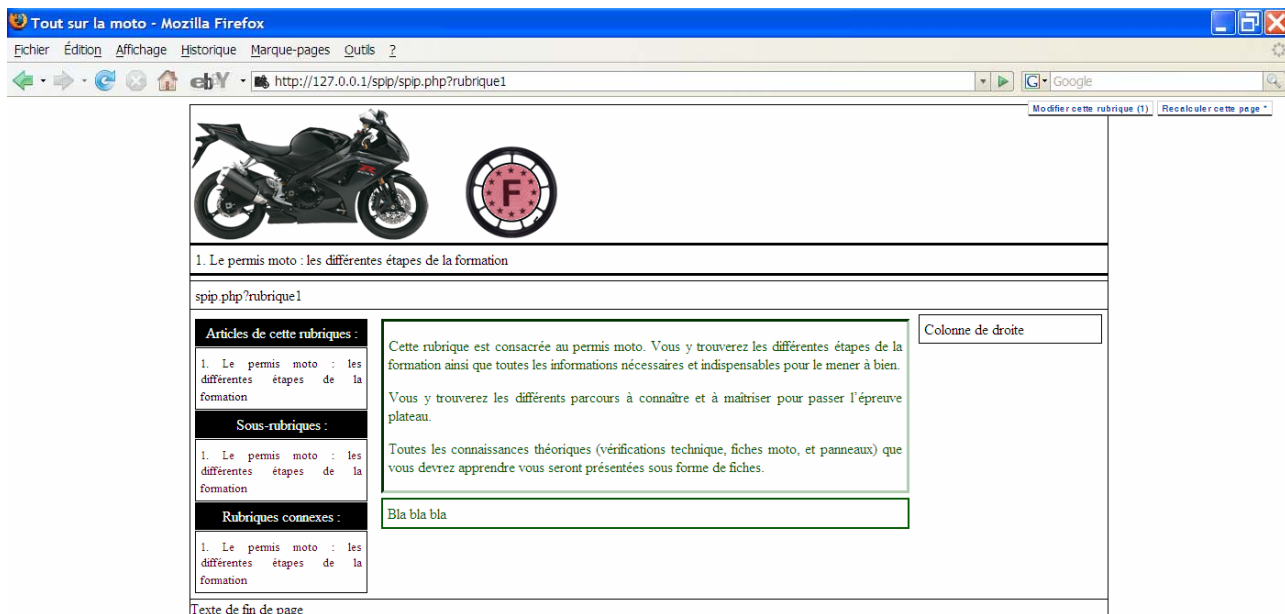
Nous pouvons maintenant appliquer ce même principe à notre fichier HTML. Récupérez le contenu du fichier `base.html` et copiez-le à l'intérieur de la boucle `rubrique_principale` dans le fichier `rubrique.html`. Ce code contient déjà les mots-clefs SPIP, mais ceux-ci n'étaient pas interprétés puisque le nom de fichier `base.html` n'est pas reconnu par SPIP.

Le code source complet de rubrique.html est détaillé page suivante, les éléments à modifier par rapport au fichier base.html étant mis en évidence. Testez ensuite le fonctionnement du site en accédant à l'adresse suivante :

<http://127.0.0.1/spip/spip.php?rubrique1>

Le résultat est présenté sur la capture d'écran page suivante.

```
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>#NOM_SITE_SPIP</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    <link href="#URL_SITE_SPIP/squelettes/moto.css" rel="stylesheet" type="text/css">
    <link rel="shortcut icon" href="#URL_SITE_SPIP/squelettes/IMAGES/tiny_moto.gif">
  </head>
  <body>
    <center>
      <div id="GLOBAL">
        <div id="BANDEAU">
          <div id="MENSECTEUR">
            <div class="CADRELOGO">
              <a href="[#URL_RUBRIQUE]">
                <div class="LOGORUBRIQUE">
                  
                </div>
              </a>
            </div>
          </div>
        </div>
      </div>
      <div id="TITRERUBRIQUE">[#TITRE] </div>
      <div id="HIERARCHIE">[#URL_RUBRIQUE] </div>
      <div id="CONTENU">
        <div id="COLONNEGAUCHE">
          <div id="MENURUBRIQUES" class="ELEMENTMENU">
            <div class="TITREMENU">Articles de cette rubriques </div>
            <ul><li><a href="[#URL_ARTICLE]">[#TITRE]</a></li></ul>
            <div class="TITREMENU">Sous-rubriques </div>
            <ul><li><a href="[#URL_RUBRIQUE]">[#TITRE]</a></li></ul>
            <div class="TITREMENU">Rubriques connexes </div>
            <ul><li><a href="[#URL_RUBRIQUE]">[#TITRE]</a></li></ul>
          </div>
          <div id="COLONNECENTRE">
            <div id="DESCRIPTIF">[#DESCRIPTIF]</div>
            <div id="TEXTE">[#TEXTE]</div>
          </div>
          <div id="COLONNEDROITE"> Colonne de droite </div>
        </div>
        <div id="FIN">Texte de fin de page</div>
      </div>
    </center>
  </body>
</html>
</BOUCLE_rubrique_principale>
```

Nous constatons que tous les `[ (#TITRE) ]` ont été remplacés par le titre de la rubrique n°1, de même que tous les `[ (#URL_RUBRIQUE) ]` ont été remplacés par l'adresse de la rubrique n°1, à savoir `spip.php?rubrique1`, même si ça n'apparaît pas visuellement.

Vous pouvez essayer de cliquer sur les différents liens, tous vous conduiront à cette même page. Pas très utile ...

Donc, il nous faut maintenant développer notre fichier `squelette rubrique.html` en insérant de nouvelles boucles permettant

- Pour la « barre de titre » :
 - o D'afficher tous les logos des secteurs, dans la boîte `MENUSECTEUR` ;
 - o D'afficher la hiérarchie de rubriques qui ont conduit à celle-ci ;
- Pour le menu de la colonne gauche :
 - o D'afficher tous les articles contenus dans cette rubrique ;
 - o D'afficher toutes les rubriques contenues dans cette rubrique ;
 - o D'afficher toutes les rubriques voisines (ou rubriques sœurs, celles qui dépendent de la même rubrique que la rubrique courante).

7.2.2. Les boucles imbriquées

Nous allons commencer par la description des boucles permettant de rendre notre menu de navigation fonctionnel. Ce menu contient les articles de la rubrique, les rubriques filles, et les rubriques sœurs. Le but des boucles que nous souhaitons écrire est de toutes les lister.

Reprenons le code HTML du menu :

```
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
...
  <div id="MENURUBRIQUES" class="ELEMENTMENU">
    <div class="TITREMENU">Articles de cette rubrique :</div>
    <ul><li><a href="[(#URL_ARTICLE)]">[(#TITRE)]</a></li></ul>
    <div class="TITREMENU">Sous-rubriques :</div>
    <ul><li><a href="[(#URL_RUBRIQUE)]"> [(#TITRE)]</a></li></ul>
    <div class="TITREMENU">Rubriques connexes :</div>
    <ul><li><a href="[(#URL_RUBRIQUE)]"> [(#TITRE)]</a></li></ul>
  </div>
  ...
</BOUCLE_rubrique_principale>
```


Ainsi, ce code récupère l'identifiant via l'URL de la page : rubrique1. La boucle `_rubrique_principale` retourne donc tous les paramètres de la rubrique correspondant à `id_rubrique=1`. Tous les mots-clefs utilisés dans cette boucle se réfèrent donc à la rubrique n°1. Nous allons voir maintenant comment explorer la base de données de notre site à l'aide d'autres boucles imbriquées dans celle-ci.

a. Boucle articles

Nous voulons récupérer tous les articles dans cette rubrique ; en d'autres termes, tous les articles dont la rubrique est celle-ci. Pour ce faire nous utilisons la boucle `ARTICLES` :

```
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
    ...
    <BOUCLE_articles(ARTICLES){id_rubrique}{tous}>
    ...
</BOUCLE_rubrique_principale>
```

Cette boucle retournera donc **tous** les articles (paramètre `tous`) de la rubrique en cours (le paramètre `id_rubrique` de la boucle `ARTICLES` fait référence à la rubrique retournée par la boucle `RUBRIQUES` dans laquelle elle se situe). Il ne restera plus qu'à afficher les adresses des articles récupérés et leurs titres à l'aide des mots-clefs `#URL_ARTICLE` et `#TITRE` qui cette fois-ci s'appliquera bien à la boucle `_articles` et non plus à la boucle `_rubrique_principale`.

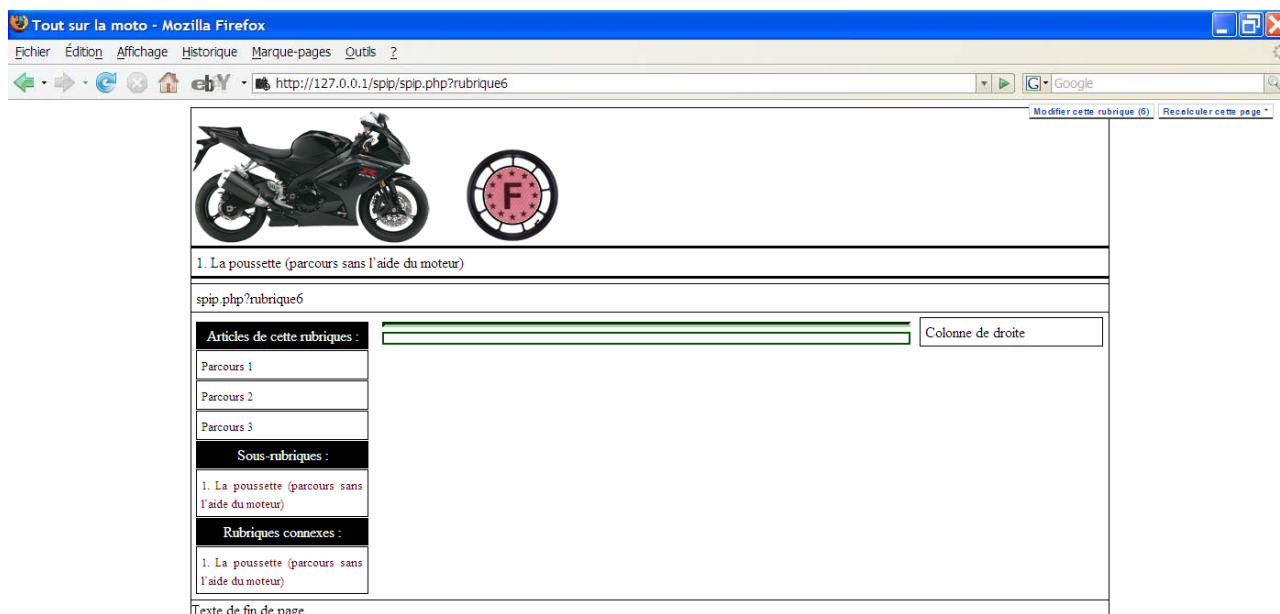
Modifions le contenu HTML en conséquence :

```
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
    ...
    <div id="MENURUBRIQUES" class="ELEMENTMENU">
    <div class="TITREMENU">Articles de cette rubrique :</div>
    <BOUCLE_articles(ARTICLES){id_rubrique}{tous}>
    <ul><li><a href="[ (#URL_ARTICLE)]">[ (#TITRE)]</a></li></ul>
    </BOUCLE_articles>
    <div class="TITREMENU">Sous-rubriques :</div>
    <ul><li><a href="[ (#URL_RUBRIQUE)]">[ (#TITRE)]</a></li></ul>
    <div class="TITREMENU">Rubriques connexes :</div>
    <ul><li><a href="[ (#URL_RUBRIQUE)]">[ (#TITRE)]</a></li></ul>
    </div>
</BOUCLE_rubrique_principale>
```

Voyons le résultat, et comparons-le avec le résultat précédent :



On constate, dans le menu de gauche, que sous **Articles de cette rubrique :** il n'y a plus de titre. Ceci est dû au fait que la rubrique dans laquelle nous nous situons ne contient aucun article. Un test avec une autre rubrique (rubrique6, ici, donc l'adresse est <http://127.0.0.1/spip/spip.php?rubrique1>) nous montre bien la liste des articles (cf. capture d'écran ci-dessous).



Si nous observons le code source HTML de cette page, nous constatons que :

```
<BOUCLE_articles(ARTICLES){id_rubrique}{tous}>
  <ul><li><a href="[ (#URL_ARTICLE)]">[ (#TITRE)]</a></li></ul>
</BOUCLE_articles>
```

a été remplacé par :

```
<ul><li><a href="spip.php?article1">Parcours 1</a></li></ul>
<ul><li><a href="spip.php?article2">Parcours 2</a></li></ul>
<ul><li><a href="spip.php?article3">Parcours 3</a></li></ul>
```

On y retrouve le même code HTML répété trois fois, à chaque fois en remplaçant l'adresse du lien [(#URL_ARTICLE)] et le texte de ce lien [(#TITRE)] par les valeurs trouvées en base de données.

Donc, lorsque la boucle retourne des résultats la présentation est correcte. Mais lorsque la boucle ne retourne aucun résultat, on affiche un titre de menu sans élément sous celui-ci. Un comportement logique serait donc, lorsque la rubrique courante ne contient aucun article, de ne pas en afficher la liste et de supprimer le titre **Articles de cette rubrique :**. Ceci est rendu possible par les boucles conditionnelles.

b. Les boucles conditionnelles

La ligne ci-dessous met en évidence ce que nous souhaitons afficher uniquement lorsque des articles sont présents, en d'autres termes, uniquement lorsque la boucle ARTICLES retourne des résultats :

```
<div class="TITREMENU">Articles de cette rubriques :</div>
```

Modifions ce code pour y insérer les balises de boucle conditionnelle :

```

<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
...
  <div id="MENURUBRIQUES" class="ELEMENTMENU">
    <B_articles>
      <div class="TITREMENU">Articles de cette rubriques :</div>
      <BOUCLE_articles(ARTICLES){id_rubrique}{tous}>
        <ul><li><a href="[ (#URL_ARTICLE)]">[ (#TITRE)]</a></li></ul>
      </BOUCLE_articles>
    </B_articles>
  </B_articles>
  <div class="TITREMENU">Sous-rubriques :</div>
  <ul><li><a href="[ (#URL_RUBRIQUE)]"> [ (#TITRE)]</a></li></ul>
  <div class="TITREMENU">Rubriques connexes :</div>
  <ul><li><a href="[ (#URL_RUBRIQUE)]"> [ (#TITRE)]</a></li></ul>
</div>
</BOUCLE_rubrique_principale>

```

Nous avons ajouté deux nouvelles balises `<B_articles>` et `</B_articles>`. Notez que le nom après B doit être le même que celui après BOUCLE dans `<BOUCLE_articles>` pour que cela fonctionne.

Tout le code HTML compris entre `<B_articles>` et `</B_articles>` sera affiché une fois, **et une fois seulement**, mais uniquement si la boucle `_articles` retourne un ou plusieurs résultats. Si aucun résultat n'est retourné, c'est alors le code compris entre `</B_articles>` et `</B_articles>` qui sera affiché, une fois seulement également ; ici, rien ne sera affiché, mais nous aurions pu y mettre une phrase du genre *Aucun article dans cette rubrique*, par exemple. Dans notre cas, la balise `</B_articles>` est optionnelle puisqu'aucun code ne la précède.

Voyons le résultat :



Nous constatons que le menu ne fait plus apparaître la liste des articles.

c. Boucles de sous-rubriques et des rubriques sœurs

Nous allons appliquer exactement le même principe de boucles imbriquées aux sous-rubriques et des rubriques sœurs. Pour la boucle `_articles`, il nous suffisait de préciser `id_rubrique` comme critère de sélection.

Dans le cas des sous-rubriques, il existe un critère de sélection pour les rubriques filles qui se nomme `id_parent` :

```

<BOUCLE_sousrubriques(RUBRIQUES){id_parent}{par num titre}{tous}>

```

Voyons le code complet de la boucle :

```
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
...
<div id="MENURUBRIQUES" class="ELEMENTMENU">
...
  <B_autresrubriques>
    <div class="TITREMENU"> Rubriques connexes :</div>
    <BOUCLE_autresrubriques(RUBRIQUES){meme_parent}{par num titre}{tous}>
      <ul><li><a href="[URL_RUBRIQUE]"> [(#TITRE)]</a></li></ul>
    </BOUCLE_autresrubriques>
  </B_autresrubriques>
</B_autresrubriques>
...
</div>
...
</BOUCLE_rubrique_principale>
```

Ce code est à comprendre de la façon suivante : trouver toutes les rubriques dont la rubrique mère (critère `id_parent` de la boucle `_sousrubriques`) correspond à la rubrique courante (critère `id_rubrique` de la boucle `_rubrique_principale`).

Autrement dit : trouver les rubriques pour lesquelles `id_parent=id_rubrique`.

On découvre par la même occasion un nouveau critère `{par num titre}`. Ce critère n'est pas un critère de sélection, mais un critère de classement : les résultats seront retournés par l'ordre des numéros qui précèdent leur titre.

Dans le cas des rubriques soeurs, il existe un critère de sélection qui permet justement ceci ; il se nomme `meme_parent` :

```
<BOUCLE_autresrubriques(RUBRIQUES){meme_parent}{par titre}{tous}{exclus}>
```

Voyons le code complet de la boucle :

```
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
...
<div id="MENURUBRIQUES" class="ELEMENTMENU">
...
  <B_sousrubriques>
    <div class="TITREMENU">Sous-rubriques :</div>
    <BOUCLE_sousrubriques(RUBRIQUES){id_parent}{par num titre}{tous}>
      <ul><li><a href="[URL_RUBRIQUE]"> [(#TITRE)]</a></li></ul>
    </BOUCLE_sousrubriques>
  </B_sousrubriques>
</B_sousrubriques>
...
</div>
...
</BOUCLE_rubrique_principale>
```

Le dernier critère `{exclus}` permet d'exclure des résultats la rubrique (ou l'article, etc.) dans laquelle on se situe déjà. En l'occurrence, sans ce critère, cette boucle nous retournerait la rubrique courante puisque celle-ci fait bien partie des rubriques « sœurs » (tautologie : la rubrique courante appartient à l'ensemble des rubriques dont la rubrique mère est la mère de la rubrique courante). Il serait curieux de trouver sur une page un lien qui nous conduirait à cette même page, et c'est le rôle de ce critère `{exclus}` que d'empêcher ce genre de comportement des boucles SPIP.

Et voyons le résultat dans la capture d'écran page suivante, sur lequel nous constatons que les différents menus nous retournent les titres des deux sous-rubriques et des deux rubriques de même niveau (rubriques sœurs) :



Le squelette rubrique est presque finalisé. Il nous reste à définir la boucle **HIERARCHIE** avant de finir par le menu des secteurs avec les logos.

7.2.3. La boucle **HIERARCHIE**

Pour l'instant, la boîte « hiérarchie » ne contient que la ligne suivante :

```
<div id="HIERARCHIE">[ (#URL_RUBRIQUE) ]</div>
```

La boucle **HIERARCHIE** consiste à afficher toute l'arborescence de rubriques ayant conduit jusqu'à la rubrique ou à l'article en cours. Elle se présente de la façon suivante :

```
<BOUCLE_hierarchie(HIERARCHIE){id_rubrique}{" - ">
...
</BOUCLE_hierarchie>
```

Le paramètre `id_rubrique` permet de retourner l'arborescence ayant conduit jusqu'à la rubrique identifiée par ce paramètre. Dans le cas d'un article, on le remplacera simplement par le paramètre `id_article` (naturellement, la boucle hiérarchie devra dans ce cas être imbriquée dans une boucle **ARTICLES**).

Le second paramètre `{" - "}`, optionnel, précise le séparateur à insérer entre deux rubriques successives retournées par la boucle.

Ainsi, le code suivant nous permettra d'afficher, sur une seule et même ligne :

Vous êtes dans : rubrique1 - rubrique2 - ... - rubrique_courante

Chacun des titres des rubriques parentes sera associé à un lien vers la rubrique concernée.

```
<div id="HIERARCHIE">
  Vous êtes dans :
  <BOUCLE_hierarchie(HIERARCHIE){id_rubrique}{" - ">
    [ <a href="#URL_RUBRIQUE">(#TITRE|supprimer_numero)</a> ]
  </BOUCLE_hierarchie>
  [ - (#TITRE|supprimer_numero) ]
</div>
```

Le résultat obtenu est présenté page suivante. On devine sur cette capture d'écran que le titre « Le permis moto : les différentes étapes de la formation » n'est pas un lien. Ceci est dû au fait que la rubrique courante est un secteur : elle ne possède pas de rubrique mère.

Le titre est affiché de la façon suivante :

```
[ - (#TITRE|supprimer_numero)]
```

L'option `supprimer_numero` permet, comme son nom l'indique, de supprimer le numéro en début de titre.

Ainsi, le titre « 1.Le permis moto : ... », sera affiché « Le permis moto : ... ».

Notez que le numéro est suivi d'un point. L'intérêt de mettre un numéro devant le titre, pour ensuite le supprimer lors de l'affichage, permet simplement de pouvoir classer les rubriques selon un ordre défini par le concepteur Web en utilisant le paramètre `{par num titre}` lors de l'affichage, plutôt que par un ordre alphabétique « incontrôlé » (non).



Dans le cas de la rubrique 6, par exemple, nous obtenons le résultat suivant :



On constate ici que la rubrique courante (« La poussette ») est précédée de ses deux rubriques parentes (« Le plateau » et « Le permis moto : ... »), lesquelles sont affichées sous forme de lien.

7.3. La gestion des logos

Notre menu secteurs est bien pauvre : un seul logo est présent dont l'image est codée de façon statique dans le fichier source HTML, et dont le lien ramène à la rubrique courante. Nous allons modifier cela en faisant en sorte d'afficher le logo correspondant à chaque rubrique.

Pour l'instant le code correspondant à notre « bandeau » de titre est le suivant :


```

<div id="BANDEAU">
  <div id="MENSECTEUR">
    <div class="CADRELOGO">
      <a href="[#URL_RUBRIQUE]">
        <div class="LOGORUBRIQUE">
          
        </div>
      </a>
    </div>
  </div>
</div>

```

Nous voulons remplacer l'image IMAGES/F2.gif par le logo de chaque rubrique, et le lien [(#URL_RUBRIQUE)] par celui conduisant à la rubrique concernée. Il nous faut donc introduire une nouvelle boucle RUBRIQUES qui va nous retourner toutes les rubriques de plus haut niveau, que l'on appelle « secteurs ».

Deux méthodes sont possibles.

1. La première méthode consiste à exploiter une boucle RUBRIQUES classique en prenant en compte la particularité des secteurs : ce sont des rubriques qui n'ont pas de rubrique mère, puisqu'elles sont toutes à la racine du site. Il suffit donc d'utiliser comme critère de sélection {id_parent=0} (rubrique dont la rubrique mère est la rubrique n°0, donc n'existant pas).
2. La seconde méthode, introduite par la suite lors des évolutions de SPIP, utilise un nouveau critère de sélection conçu justement pour cet usage, {racine}, dont le nom est explicite. C'est naturellement cette seconde solution que nous allons préférer :

```

<div id="BANDEAU">
  <div id="MENSECTEUR">
    <div class="CADRELOGO">
      <BOUCLE_secteurs(RUBRIQUES){racine}{par num titre}{tout}>
        <a class="LOGORUBRIQUE" href="#URL_RUBRIQUE">#LOGO_RUBRIQUE</a>
      </BOUCLE_secteurs>
    </div>
  </div>
</div>

```

Nous sélectionnons **toutes** les rubriques à la **racine**, les classons selon l'**ordre du numéro** qui précède leur titre, et affichons leur logo (mot-clef #LOGO_RUBRIQUE), dans un lien vers l'adresse de la rubrique (mot-clef #URL_RUBRIQUE). Voici le résultat obtenu :



7.4. Listing complet de rubrique.html

```
<BOUCLE_rubrique_principale(RUBRIQUES){id_rubrique}>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>#NOM_SITE_SPIP</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    <link href="#URL_SITE_SPIP/squelettes/moto4.css" rel="stylesheet"
type="text/css">
    <link rel="shortcut icon" href="#URL_SITE_SPIP/squelettes/IMAGES/tiny_moto.gif">
  </head>
  <body>
    <center>
      <div id="GLOBAL">
        <div id="BANDEAU">
          <div id="MENSECTEUR">
            <div class="CADRELOGO">
              <BOUCLE_secteurs(RUBRIQUES){racine}{par num titre}{tout}>
                <a class="LOGORUBRIQUE" href="#URL_RUBRIQUE">#LOGO_RUBRIQUE</a>
              </BOUCLE_secteurs>
            </div>
          </div>
        </div>
      </div>
      <div id="TITRERUBRIQUE">[ (#TITRE)]</div>
      <div id="HIERARCHIE">
        Vous êtes dans :
        <BOUCLE_hierarchie(HIERARCHIE){id_rubrique}{" - ">
          [<a href="#URL_RUBRIQUE"> (#TITRE|supprimer_numero)</a>]
        </BOUCLE_hierarchie>
        [ - (#TITRE|supprimer_numero)]
      </div>
      <div id="CONTENU">
        <div id="COLONNEGAUCHE">
          <div id="MENURUBRIQUES" class="ELEMENTMENU">
            <B_listearticles>
              <div class="TITREMENU">Articles de cette rubriques :</div>
              <BOUCLE_listearticles(ARTICLES){id_rubrique}{par titre}{tous}>
                [<ul><li><a
href="#URL_ARTICLE"> (#TITRE|supprimer_numero)</a></li></ul>]
              </BOUCLE_listearticles>
            </B_listearticles>
            <B_sousrubriques>
              <div class="TITREMENU">Sous-rubriques :</div>
              <BOUCLE_sousrubriques(RUBRIQUES){id_parent}{par num titre}{tous}>
                [<ul><li><a
href="#URL_RUBRIQUE"> (#TITRE|supprimer_numero)</a></li></ul>]
              </BOUCLE_sousrubriques>
            </B_sousrubriques>
            <B_autresrubriques>
              <div class="TITREMENU">Rubriques connexes :</div>
              <BOUCLE_autresrubriques(RUBRIQUES){meme_parent}{par num
titre}{tous}{exclus}>
                [<ul><li><a
href="#URL_RUBRIQUE"> (#TITRE|supprimer_numero)</a></li></ul>]
              </BOUCLE_autresrubriques>
            </B_autresrubriques>
          </div>
        </div>
        <div id="COLONNECENTRE">
          <div id="DESCRIPTIF">[ (#DESCRIPTIF)]</div>
          <div id="TEXTE">[ (#TEXTE)]</div>
        </div>
        <div id="COLONNEDROITE"> Colonne de droite </div>
      </div>
      <div id="FIN">Texte de fin de page</div>
    </div>
  </center>
</body>
</html>
```


8. Questions

Exercice 1 : La balise `<center></center>` utilisée pour centrer la boîte GLOBAL est en cours d'abandon : dans le futur, elle ne sera plus reconnue. Proposer une solution permettant de centrer la boîte GLOBAL sans avoir recours à cette balise.

Indice : utilisez les paramètres CSS `position`, `left` et `margin-left`.

Exercice 2 : Le squelette des rubriques étant finalisé, proposer un squelette `article.html` pour les articles en utilisant la même présentation.

Exercice 3 : Le fichier `sommaire.html` est un squelette particulier : c'est celui qui est utilisé lorsque que l'on accède au site sans faire référence à une rubrique ou à un article particulier (adresse `http://127.0.0.1/spip/`, dans notre exemple).

Proposez un squelette `sommaire.html` qui fasse la liste des 3 derniers articles publiés sur le site (aidez vous de l'annexe B pour ce faire).

Indice : utilisez le critère SPIP `{par date}`, et un critère d'affichage d'une partie des résultats.

Annexe A – Liste d'éléments XHTML

Balises de structuration du texte

Balise	Type	Description
<p>	Block	Paragraphe
<hr />	Block	Crée une ligne de séparation horizontale
<address>	Block	Permet d'indiquer une adresse, ou éventuellement l'auteur d'un document. Le texte est généralement mis en italique.
	Inline	Permet d'indiquer un texte qui a été supprimé. Le texte est généralement barré.
<ins>	Inline	Permet d'indiquer un texte qui a été inséré. Le texte est généralement souligné.
<dfn>	Inline	Permet d'indiquer une définition.
<kbd>	Inline	Permet d'indiquer un code que doit taper le visiteur.
<pre>	Block	Le texte à l'intérieur de la balise <pre> sera affiché tel qu'il a été tapé dans le code (espaces et entrées compris). Une police de taille fixe est utilisée.
 	Inline	Retour à la ligne
<a>	Inline	Lien hypertexte. Indiquez l'url de destination grâce à l'attribut href : Rendez-vous sur l'autre page
	Inline	Insère une image. Utilisez les attributs src (pour indiquer l'adresse de l'image) et alt (pour indiquer un texte de remplacement). Ces 2 attributs sont obligatoires. Exemple :
<h1>	Block	Titre de niveau 1
<h2>	Block	Titre de niveau 2
<h3>	Block	Titre de niveau 3
<h4>	Block	Titre de niveau 4
<h5>	Block	Titre de niveau 5
<h6>	Block	Titre de niveau 6
	Inline	Mise en valeur (faible). Le texte est généralement mis en italique.
	Inline	Mise en valeur (forte). Le texte est généralement mis en gras.
<sub>	Inline	Mise en indice
<sup>	Inline	Mise en exposant
<q>	Inline	Citation (courte)
<cite>	Inline	Citation (moyenne)
<blockquote>	Block	Citation (longue). Vous devez obligatoirement mettre une balise de paragraphe à l'intérieur du blockquote. Par exemple : <blockquote> Texte de la citation </blockquote>
<acronym>	Inline	Sert à définir des acronymes, comme C.I.A. On utilise généralement l'attribut title pour donner la définition de l'acronyme quand on pointe dessus : <acronym title="Central Intelligence Agency">C.I.A.</acronym>

Balises de liste

Balise	Type	Description
<code></code>	Block	Liste à puces non numérotée. Vous devez mettre un <code></code> par élément de la liste. Exemple : <pre> Un élément Un autre élément </pre>
<code></code>	Block	Liste à puces numérotée. Vous devez mettre un <code></code> par élément de la liste. Exemple : <pre> Elément n°1 Elément n°2 </pre>
<code></code>	list-item	Permet de créer un élément de liste. Le type de la balise est particulier car elle n'est ni block ni inline. On dit qu'elle est de type list-item.
<code><dl></code>	Block	Liste de définitions. Vous devez alterner chaque terme <code><dt></code> par sa définition <code><dd></code> . Exemple : <pre><dl> <dt>Porte</dt> <dd>Ouverture dans un mur permettant d'entrer et de sortir</dd> <dt>Théâtre</dt> <dd>Lieu où l'on représente des ouvrages dramatiques</dd> </dl></pre>
<code><dt></code>	Block	Terme à définir
<code><dd></code>	Block	Définition du terme

Balises de tableau

Balise	Type	Description
<code><table></code>	Block	Délimite un tableau.
<code><caption></code>	-	Permet de donner un titre au tableau
<code><tr></code>	-	Ligne de tableau
<code><th></code>	-	Cellule d'en-tête du tableau (généralement mise en gras)
<code><td></code>	-	Cellule du tableau
<code><thead></code>	-	Balise non obligatoire permettant d'insérer l'en-tête du tableau. Si vous choisissez d'utiliser <code><thead></code> , <code><tfoot></code> et <code><tbody></code> , vous devez les mettre dans l'ordre suivant dans votre code source : 1. <code><thead></code> 2. <code><tfoot></code> 3. <code><tbody></code>
<code><tbody></code>	-	Balise non obligatoire permettant d'insérer le corps du tableau
<code><tfoot></code>	-	Balise non obligatoire permettant d'insérer le pied du tableau

Balises de formulaire

Balise	Type	Description
<form>	Block	Délimite un formulaire. Vous devrez généralement donner 2 attributs à la balise <form> method : indique la méthode d'envoi du formulaire (get ou post). Si vous ne savez pas quoi utiliser, mettez post. action : la page vers laquelle le visiteur doit être redirigé lorsqu'il a validé votre formulaire.
<fieldset>	Block	Permet de regrouper plusieurs éléments d'un formulaire. On l'utilise généralement dans de grands formulaires. Pour donner un titre à votre groupe, utilisez la balise <legend>
<legend>	Inline	Titre d'un groupe dans un formulaire. À utiliser à l'intérieur d'un <fieldset>
<label>	Inline	Titre d'un élément de formulaire. Généralement, vous devrez mettre l'attribut for sur cette balise pour indiquer l'ID de l'élément auquel correspond le label.
<input />	Block	Champ de formulaire. Il existe de nombreux types de champs différents. Vous choisissez le type de champ que vous désirez grâce à l'attribut type : <pre> <!-- Zone de texte d'une ligne --> <input type="text" /> <!-- Mot de passe (le texte est caché) --> <input type="password" /> <!-- Envoi de fichier --> <input type="file" /> <!-- Case à cocher --> <input type="checkbox" /> <!-- Bouton d'option --> <input type="radio" /> <!-- Bouton d'envoi --> <input type="submit" /> <!-- Bouton de remise à zéro --> <input type="reset" /> <!-- Champ caché --> <input type="hidden" /> </pre> Pensez à donner un nom à vos champs grâce à l'attribut name
<textarea>	Block	Zone de saisie multiligne. Vous pouvez définir sa taille grâce aux attributs rows et cols (nombre de lignes et colonnes) ou bien le faire en CSS grâce aux propriétés width et height.
<select>	Block	Liste déroulante. Utilisez la balise <option> pour créer chaque élément de la liste : <pre> <select name="pays"> <option value="france">France</option> <option value="espagne">Espagne</option> <option value="italie">Italie</option> </select> </pre>
<option>	Block	Élément d'une liste déroulante
<optgroup>	Block	Groupe d'éléments d'une liste déroulante. À utiliser dans le cas d'une grande liste déroulante.

Balises génériques

Les balises génériques sont des balises qui n'ont pas de sens sémantique. En effet, toutes les autres balises XHTML ont un **sens** : <p> signifie "Paragraphe", <h2> signifie "Sous-titre" etc. Parfois, on a besoin d'utiliser des balises génériques (aussi appelées balises universelles) car aucune des autres balises ne convient. On utilise le plus souvent des balises génériques pour construire son design.

Il y a 2 balises génériques : l'une est `inline`, l'autre est `block`.

Balise	Type	Description
	Inline	Balise générique de type inline
<div>	Block	Balise générique de type block

Ces balises ont un intérêt uniquement si vous leur donnez un attribut `class`, `id` ou `style` :

- **class** : indique le nom de la classe CSS à utiliser.
- **id** : donne un nom à la balise. Ce nom doit être unique sur toute la page car il permet d'identifier la balise. Vous pouvez vous servir de l'ID pour de nombreuses choses, comme par exemple pour un lien vers une ancre, pour un style CSS de type ID, pour des manipulations en Javascript etc.
- **style** : cette balise vous permet d'indiquer directement le code CSS à appliquer. Vous n'êtes donc pas obligés d'avoir une feuille de style à part, vous pouvez juste mettre directement les attributs CSS. Notez qu'il est préférable de ne pas utiliser cet attribut et de passer à la place par une feuille de style externe car cela rend votre site plus facile à mettre à jour par la suite.

Ces 3 attributs ne sont pas réservés aux balises génériques : vous pouvez aussi les mettre sur la plupart des autres balises sans aucun problème

Annexe B – Liste des propriétés CSS

Police, taille et décorations

Type	Nom	Valeurs possibles
Nom de police	font-family	Indiquer les noms de polices possibles par ordre de préférence : font-family:police1, police2, police3; Si le visiteur a la police 1, il l'utilisera. Sinon, il regarde s'il a la police 2, puis la police 3, etc. Utilisez des guillemets si le nom de la police comporte des espaces. Essayez de toujours mettre comme dernière police possible "serif" ou "sans-serif".
Taille du texte	font-size	Indiquez la taille du texte. Plusieurs unités sont possibles : • px (pixels) • % (pourcentage, 100% = normal) • em (taille relative, 1.0 = normal) • ex (taille relative à la hauteur de la lettre "x". 1.0 = normal) • nom de taille : ○ xx-small : très très petit ○ x-small : très petit ○ small : petit ○ medium : moyen ○ large : grand ○ x-large : très grand ○ xx-large : très très grand
Gras	font-weight	bold : gras bolder : plus gras lighter : plus fin normal : pas gras (par défaut)
Italique	font-style	italic : italique oblique : autre façon de mettre en italique normal : normal (par défaut)
Décoration	text-decoration	underline : souligné overline : ligne au-dessus line-through : barré blink : clignotant none : normal (par défaut)
Petites capitales	font-variant	small-caps : petites capitales normal : normal (par défaut)
Capitales	text-transform	uppercase : tout mettre en majuscules lowercase : tout mettre en minuscules capitalize : début des mots en majuscules none : normal (par défaut)
Méga-propriété de police	font	Indiquez dans n'importe quel ordre des valeurs possibles pour font-weight, font-style, font-size, font-variant, font-family. Attention exception : le nom de la police (font-family) doit être placé en dernier dans la liste dans tous les cas. Vous n'êtes pas obligés de mettre une valeur de chacune de ces propriétés. Exemple : font: bold, 16px, Arial; Cela mettra votre texte en gras, 16 pixels, Arial.

Alignement

Type	Nom	Valeurs possibles
Alignement horizontal	text-align	left : à gauche (par défaut) center : centré right : à droite justify : texte justifié (prend toute la largeur de la page)
Alignement vertical	vertical-align	À utiliser dans des cellules de tableau, ou dans des éléments inline eux-mêmes contenus dans un élément inline. top : en haut middle : au milieu bottom : en bas

Hauteur de ligne	line-height	Indiquer une valeur en pixels (px) ou en pourcentage (%)
Alinéa	text-indent	Indiquez une valeur en pixels (px) pour définir l'alinéa de vos paragraphes. Vos paragraphes commenceront avec le retrait que vous avez indiqué.
Césure	white-space	normal : le passage à la ligne est automatique (par défaut) nowrap : pas de passage à la ligne automatique, à moins qu'une balise xHTML comme <code>
</code> ne soit présente. pre : le passage à la ligne se fait tel que le texte a été saisi dans le code source (comme la balise xHTML <code><pre></code>)

Propriétés de couleur et de fond

Couleur

Type	Nom	Valeurs possibles
Couleur de texte	color	Indiquer une couleur avec l'une des méthodes suivantes : - En tapant le nom de la couleur en anglais (black, blue, green, white, red...). - En indiquant la couleur en hexadécimal (#CC48A1) - En indiquant la couleur en RGB : rgb (128, 255, 0)
Couleur de fond	background-color	Même fonctionnement que color. Cela définit cette fois la couleur de fond du texte

Image de fond

Type	Nom	Valeurs possibles
Image de fond	background-image	Indiquer l'url de l'image (notation absolue ou relative) background-image:url("images/fond.png"); /* Notation relative */ background-image:url("http://www.monsite.com/images/fond.png"); /* Notation absolue */
Fond fixé	background-attachment	fixed : le fond reste fixe quand on descend plus bas sur la page scroll : le fond défile avec le texte (par défaut)
Répétition du fond	background-repeat	repeat : le fond se répète (par défaut) repeat-x : le fond ne se répète que sur une ligne, horizontalement repeat-y : le fond ne se répète que sur une colonne, verticalement no-repeat : le fond ne se répète pas, il n'est affiché qu'une fois
Position du fond	background-position	2 façons de faire : - En notant une distance en px ou %, par rapport au coin en haut à gauche. background-position:50px 200px; /* 50 px à droite, 200px en bas */ - En utilisant des valeurs prédéfinies, une pour la verticale et une pour l'horizontale : top : en haut, verticalement center : au milieu, verticalement bottom : en bas, verticalement left : à gauche, horizontalement center : au centre, horizontalement right : à droite, horizontalement background-position : bottom right; /* en bas à droite */
Méga-propriété de fond	background	Indiquer une ou plusieurs valeurs issues des propriétés background-image, background-repeat, background-attachment, background-position. L'ordre des valeurs n'a pas d'importance et vous n'êtes pas obligés de mettre toutes les valeurs de ces propriétés (au moins une suffit) /* Le fond fond.png reste affiché en haut à droite de l'écran et n'est pas répété. */ background:url("images/fond.png") no-repeat fixed top right;

Propriétés des boîtes

Dimensions

Type	Nom	Valeurs possibles
Largeur	width	Valeur en px, %, ou encore "auto" (valeur par défaut, la largeur dépendra du texte à l'intérieur)
Hauteur	height	Idem
Largeur minimale	min-width	Indiquer une valeur, en pixels par exemple.
Largeur maximale	max-width	Idem

Hauteur minimale	min-height	Idem
Hauteur maximale	max-height	Idem

Marges extérieures

Type	Nom	Valeurs possibles
Marge en haut	margin-top	Indiquer une valeur comme 20px, 1.5em...
Marge à gauche	margin-left	Idem
Marge à droite	margin-right	Idem
Marge en bas	margin-bottom	Idem
Méga-propriété de marge	margin	Indiquez de 1 à 4 valeurs à la suite. Selon le nombre de valeurs que vous mettez, la signification change : 1 valeur : ce sera la marge pour le haut, le bas, la gauche et la droite 2 valeurs : la première correspond à la marge pour le haut et le bas, la seconde pour la gauche et la droite 3 valeurs : la première correspond à la marge du haut, la seconde aux marges à gauche et à droite, la troisième à la marge du bas 4 valeurs : respectivement la marge du haut, de la droite, du bas, de la gauche. Par exemple, si je mets 2 valeurs : <code>margin:20px 5px; /* 20px de marge en haut et en bas, 5px à gauche et à droite */</code>

Marges intérieures

Type	Nom	Valeurs possibles
Marge intérieure en haut	padding-top	Indiquer une valeur comme 20px, 1.5em...
Marge intérieure à gauche	padding-left	Idem
Marge intérieure à droite	padding-right	Idem
Marge intérieure en bas	padding-bottom	Idem
Méga-propriété de marge intérieure	padding	Indiquez de 1 à 4 valeurs à la suite. Selon le nombre de valeurs que vous mettez, la signification change : 1 valeur : ce sera la marge pour le haut, le bas, la gauche et la droite 2 valeurs : la première correspond à la marge pour le haut et le bas, la seconde pour la gauche et la droite 3 valeurs : la première correspond à la marge du haut, la seconde aux marges à gauche et à droite, la troisième à la marge du bas 4 valeurs : respectivement la marge du haut, de la droite, du bas, de la gauche.

Bordures

Type	Nom	Valeurs possibles
Épaisseur de la bordure	border-width	Indiquer une valeur en px.
Couleur de la bordure	border-color	Indiquer une valeur de couleur.
Type de bordure	border-style	none : pas de bordure (par défaut) hidden : bordure cachée solid : ligne pleine double : ligne double (taille de bordure de 3px minimum) dashed : en tirets dotted : en pointillés inset : effet 3D "enfoncé" outset : effet 3D "surélevé" ridge : autre effet 3D
Bordure à gauche	border-left	Indiquer la couleur, l'épaisseur et le type de bordure pour la bordure gauche. L'ordre n'a pas d'importance. Exemple : <code>border-left: 2px inset blue; /* Bordure bleue de 2px avec effet 3D "enfoncé" */</code>
Bordure en haut	border-top	Idem
Bordure à droite	border-right	Idem
Bordure en bas	border-bottom	Idem
Méga-propriété de bordure	border	Indique l'apparence des bordures en haut, à droite, en bas et à gauche.

Propriétés de positionnement et d'affichage

Affichage

Type	Nom	Valeurs possibles
Type d'élément	display	none : l'élément ne sera pas affiché block : l'élément devient de type "block" (bloc, comme <p>) inline : l'élément devient de type "inline" (en ligne, comme) list-item : l'élément devient de type "élément de liste à puce" (comme)
Affichage	visibility	hidden : masqué visible : visible (par défaut) display:none; fait complètement disparaître l'élément, tandis que visibility:hidden; masque l'élément, qui continue quand même à prendre de la place sur l'écran.
Afficher seulement une partie	clip	Indiquer 4 valeurs comme ceci : clip: rect(valeur1, valeur2, valeur3, valeur4); Cela permet de n'afficher qu'une partie d'un élément. rect() permet d'indiquer les coordonnées du rectangle qui sera affiché. Les valeurs 1 à 4 correspondent respectivement aux coins haut, droite, bas et gauche du rectangle.
Limiter les dimensions	overflow	visible : tout l'élément sera affiché (par défaut). hidden : l'élément sera coupé s'il dépasse les limites définies par height et width. On ne pourra pas voir la partie du texte coupée. scroll : tout comme hidden, l'élément sera coupé s'il dépasse les limites. Toutefois, le navigateur ajoutera des barres de défilement pour qu'on puisse voir la suite du texte. auto : c'est le navigateur qui décide d'ajouter des barres de défilement ou pas en fonction des cas. Bien souvent, utiliser cette valeur revient à utiliser la valeur "scroll".

Positionnement

Type	Nom	Valeurs possibles
Flottant	float	left : flottant à gauche right : flottant à droite none : pas de flottant (par défaut)
Stopper un flottant	clear	left : supprime l'effet d'un flottant à gauche précédent right : supprime l'effet d'un flottant à droite précédent both : supprime l'effet d'un flottant précédent, qu'il soit à gauche ou à droite none : pas de suppression de l'effet du flottant (par défaut)
Type de positionnement	position	absolute : position absolue par rapport au coin en haut à gauche fixed : position fixe (fonctionne comme la position absolue). L'élément reste à sa position même quand on descend plus bas dans la page. relative : position relative, par rapport à la position "normale" de l'élément static : positionnement normal (par défaut)
Position par rapport au haut	top	Valeur en px, %, em... A utiliser pour un positionnement absolu, fixe ou relatif.
Position par rapport au bas	bottom	Idem
Position par rapport à gauche	left	Idem
Position par rapport à droite	right	Idem
Ordre d'affichage	z-index	En cas de positionnement absolu par exemple, si 2 éléments se chevauchent, z-index permet d'indiquer quel élément doit être affiché au-dessus de l'autre. Indiquez un nombre. Plus ce nombre est élevé, plus l'élément sera affiché en avant. Par exemple, si vous avez 2 éléments positionnés en absolus avec un z-index de 10 pour l'un et de 20 pour l'autre, c'est celui qui a un z-index de 20 qui sera affiché par-dessus.

Propriétés des listes

Type	Nom	Valeurs possibles
Type de liste	list-style-type	- Pour les listes non ordonnées () : disc : un disque noir (par défaut). circle : un cercle. square : un carré. none : aucune puce ne sera utilisée. - Pour les listes ordonnées () : decimal : des nombres 1, 2, 3, 4, 5... (par défaut)

		O decimal-leading-zero : des nombres commençant par zéro (01, 02, 03, 04, 05...). Ne fonctionne pas sur Internet Explorer O upper-roman : numérotation romaine majuscules (I, II, III, ...) O lower-roman : numérotation romaine minuscules (i, ii, iii,...) O upper-alpha : numérotation alphabétique majuscules (A, B, C, ...) O lower-alpha : numérotation alphabétique minuscules (a, b, c, ...) O lower-greek : numérotation grecque. Ne fonctionne pas sur Internet Explorer
Position en retrait	list-style-position	inside : sans retrait outside : avec retrait (par défaut)
Puce personnalisée	list-style-image	Indiquer l'url de l'image qui servira de puce. Exemple : list-style-image: url("images/puce.png");
Méga-propriété de liste	list-style	Vous pouvez réunir les valeurs de list-style-type, list-style-position et list-style-image. Vous n'êtes pas obligés de mettre toutes les valeurs, et l'ordre n'a pas d'importance. Exemple : list-style: inside square;

Propriétés des tableaux

Type	Nom	Valeurs possibles
Type de bordure	border-collapse	collapse : les bordures du tableau et des cellules sont mélangées. separate : les bordures du tableau et des cellules sont séparées (par défaut).
Cellules vides	empty-cells	show : les bordures des cellules vides sont affichées. collapse : les cellules vides sont masquées (par défaut).
Position du titre	caption-side	Indique la position du titre du tableau, défini via la balise <caption> top : en haut du tableau bottom : en bas du tableau left : à gauche du tableau right : à droite du tableau

Autres propriétés

Type	Nom	Valeurs possibles
Curseur de souris	cursor	auto : curseur automatique (par défaut) default : curseur standard pointer : curseur en forme de main, comme quand on pointe sur un lien text : curseur utilisé quand on pointe sur du texte wait : curseur utilisé pour indiquer une attente (sablier) progress : curseur utilisé pour indiquer une tâche de fond (curseur avec sablier) help : curseur en forme de point d'interrogation, indiquant une aide move : curseur en forme de croix, indiquant un déplacement possible n-resize : flèche vers le nord ne-resize : flèche vers le nord-est e-resize : flèche vers l'est se-resize : flèche vers le sud-est s-resize : flèche vers le sud sw-resize : flèche vers le sud-ouest w-resize : flèche vers l'ouest nw-resize : flèche vers le nord-ouest url : curseur personnalisé, de type .cur ou .ani. Exemple : cursor: url("images/curseur.cur"); Vous devez utiliser un logiciel dédié à la création de curseurs pour créer des .cur et des .ani

Annexe C – Liste des critères communs aux boucles SPIP

Classer les résultats

`{par critère_de_classement}` indique l'ordre de présentation des résultats. Ce critère de classement correspond à l'une des balises tirées de la base de données pour chaque type de boucle. Par exemple, on pourra classer les articles `{par date}`, `{par date_redac}` ou `{par titre}`. (Notez que, si les balises sont en majuscules, les critères de classement sont en minuscules.)

Cas particulier : `{par hasard}` permet d'obtenir une liste présentée dans un ordre aléatoire.

Inverser le classement. De plus, `{inverse}` provoque l'affichage du classement inversé. Par exemple `{par date}` commence par les articles les plus anciens ; avec `{par date}{inverse}` on commence la liste avec les articles les plus récents.

Depuis [SPIP 1.9](#), le critère *inverse* peut prendre en paramètre n'importe quelle balise pour varier dynamiquement le sens du tri. Par exemple, il est possible d'écrire : `<BOUCLE_exemple(ARTICLES){par #ENV{tri}}{inverse #ENV{senstri}}>`, ce qui permet de choisir la colonne de tri et le sens du tri par l'url (`&senstri=1` ou `&senstri=0`)

Classer par numéro. [\[SPIP 1.3\]](#) Lorsqu'on réalise le classement selon un élément de texte (par exemple le *titre*), le classement est réalisé par ordre *alphabétique*. Cependant, pour forcer un ordre d'affichage, on peut indiquer un numéro devant le titre, par exemple : « 1. Mon premier article », « 2. Deuxième article », « 3. Troisième... », etc ; avec un classement alphabétique, le classement de ces éléments donnerait la série « 1, 10, 11, 2, 3... ». Pour rétablir le classement selon les numéros, on peut utiliser le critère :

`{par num critère}`

Par exemple :

```
<BOUCLE_articles(ARTICLES){id_rubrique}{par date}{inverse}>
```

affiche les articles d'une rubrique classés selon l'ordre chronologique inversé (les plus récents au début, les plus anciens à la fin), et :

```
<BOUCLE_articles(ARTICLES){id_rubrique}{par titre}>
```

les affiche selon l'ordre alphabétique de leur titre ; enfin :

```
<BOUCLE_articles(ARTICLES){id_rubrique}{par num titre}>
```

les affiche selon l'ordre du numéro de leur titre (remarque : l'option `{par num titre}` ne fonctionne pas pour les plus anciennes versions de MySQL, antérieures à la version 3.23).

```
<BOUCLE_articles(ARTICLES){id_rubrique}{par multi titre}>
```

Dans le cadre d'un site multilingue le critère `{par multi critère}` permet de trier par ordre alphabétique dans chaque langue. Sans l'ajout de "multi" la boucle renvoie le même classement pour chaque langue.

Classer selon plusieurs critères A partir de [SPIP 1.8](#), [SPIP 1.8.1](#), on peut classer selon plusieurs critères : `{par critère1, critère2}`. On indique ainsi des ordres de classement consécutifs. Les résultats seront d'abord triés selon le *critère1*, puis le *critère2* pour les résultats ayant le même *critère1*. On peut spécifier autant de critères que nécessaire.

Par exemple `{par date, titre}` triera les résultats par *date* puis les résultats ayant la même *date* seront triés par *titre*.

Avec [\[SPIP 1.8.2\]](#) on peut spécifier plusieurs critères `{par ...}` pour une boucle pour arriver au même résultat. Par exemple : `{par date} {par titre}` est équivalent à l'exemple précédent.

Remarque : Quand on utilise plusieurs critères de tri, le critère `{inverse}` ne s'applique qu'au critère de tri placé juste avant.

C'est pourquoi [\[SPIP 1.8.2\]](#) introduit la notation `{!par ...}` qui inverse un critère de tri en particulier. Par exemple : `{!par date} {par num titre}` tri par *date* décroissantes puis par numéros croissants dans le *titre* pour les résultats ayant la même *date*.

Comparaisons, égalités

`{critère < valeur}` Comparaison avec une valeur fixée (on peut utiliser « > », « < », « = », « >= », « <= »). Tous les *critères de classement* (tels que tirés de la base de données) peuvent également être utilisés pour limiter le nombre de résultats.

La valeur à droite de l'opérateur peut être :

- Une valeur constante fixée dans le squelette. Par exemple :

```
<BOUCLE_art(ARTICLES){id_article=5}>
```

affiche l'article dont le numéro est 5 (utile pour mettre en vedette un article précis sur la page d'accueil).

```
<BOUCLE_art(ARTICLES){id_secteur=2}>
```

affiche les articles du secteur numéro 2.

- A partir de [\[SPIP 1.8\]](#), une balise disponible dans le contexte de la boucle. Par exemple :

```
<BOUCLE_art(ARTICLES){id_article=5}>
<BOUCLE_titre(ARTICLES) {titre=#TITRE}>
...
</BOUCLE_titre>
</BOUCLE_art>
```

sert à trouver les articles qui ont le même titre que l'article 5.

Attention : On ne peut utiliser qu'une balise simple. Il n'est pas permis de la filtrer ou de mettre du code optionnel.

Spécialement, si on veut utiliser la balise `#ENV` — ou tout autre balise prenant des paramètres —, on doit utiliser la notation : `{titre = #ENV{titre}}` et **pas** : `{titre = [#ENV{titre}]}`.

Expressions régulières :

Très puissant (mais nettement plus complexe à manipuler), le terme de comparaison « == » introduit

une comparaison selon une expression régulière. Par exemple :

```
<BOUCLE_art(ARTICLES){titre==^[aA]}>
```

sélectionne les articles dont le titre commence par « a » ou « A ».

Négation :

A partir de [SPIP 1.2] On peut utiliser la notation {xxx != yyy} et {xxx !== yyy}, le ! correspondant à la négation (opérateur logique NOT).

```
<BOUCLE_art(ARTICLES){id_secteur != 2}>
```

sélectionne les articles qui n'appartiennent pas au secteur numéro 2.

```
<BOUCLE_art(ARTICLES){titre!==^[aA]}>
```

sélectionne les articles dont le titre ne commence pas par « a » ou « A ».

Affichage en fonction de la date

Pour faciliter l'utilisation des comparaisons sur les dates, on a ajouté des critères :

- age et age_redac correspondent respectivement à l'ancienneté de la publication et de la première publication d'un article, en jours : {age<30} sélectionne les éléments publiés depuis un mois ;
- les critères mois, mois_redac, annee, annee_redac permettent de comparer avec des valeurs fixes ({annee<=2000}) pour les éléments publiés avant la fin de l'année 2000).

On peut combiner plusieurs de ces critères pour effectuer des sélections très précises. Par exemple :

```
<BOUCLE_art(ARTICLES){id_secteur=2}{id_rubrique !=3}{age<30}>
```

affiche les articles du secteur 2, à l'exclusion de ceux de la rubrique 3, et publiés depuis moins de 30 jours.

Astuce. Le critère age est très pratique pour afficher les articles ou les brèves dont la date est située « dans le futur », avec des valeurs négatives (à condition d'avoir sélectionné, dans la Configuration précise du site, l'option « Publier les articles post-datés »). Par exemple, ce critère permet de mettre en valeur des événements futurs. {age<0} sélectionne les articles ou les brèves dont la date est située dans le futur (« après » aujourd'hui)...

[SPIP 1.3] *Âge par rapport à une date fixée.* Le critère age est calculé par rapport à la date d'aujourd'hui (ainsi {age<30} correspond aux articles publiés depuis moins d'un mois par rapport à aujourd'hui). Le critère **age_relatif** compare la date d'un article ou d'une brève à une date « courante » ; par exemple, à l'intérieur d'une boucle ARTICLES, on connaît déjà une date pour chaque résultat de la boucle, on peut donc sélectionner par rapport à cette date (et non plus par rapport à aujourd'hui).

Par exemple :

```
<BOUCLE_article_principal(ARTICLES){id_article}>
```

```
<h1>#TITRE</h1>
```

```
<BOUCLE_suivant(ARTICLES){id_rubrique}{age_relatif<=0}{exclus}{par date}{0,1}>
Article suivant: #TITRE
</BOUCLE_suivant>
```

```
</BOUCLE_article_principal>
```

la BOUCLE_suivant affiche un seul article de la même rubrique, classé par date, dont la date de publication est inférieure ou égale à la date de l'« article_principal » ; c'est-à-dire l'article de la même rubrique publié après l'article principal.

De plus amples informations sur l'utilisation des dates se trouvent dans l'article sur « [La gestion des dates](#) ».

Affichage d'une partie des résultats

– {branche} A partir de [SPIP 1.8.2], limite les résultats — pour des boucles ayant un #ID_RUBRIQUE — à la branche actuelle (la rubrique actuelle et ses sous-rubriques). Par exemple :

```
<BOUCLE_articles(ARTICLES) {branche}> retournera
tous les articles de la rubrique actuelle et de ces sous-
rubriques,
```

```
<BOUCLE_articles(ARTICLES) {!branche}> retournera
tous les articles qui ne sont pas dans la rubrique actuelle
ou ses sous-rubriques,
```

On peut utiliser le critère {branche?} optionnel pour ne l'appliquer que si une rubrique est sélectionnée dans le contexte (une boucle englobante ou l'url fournie un id_rubrique). Par exemple :

```
<BOUCLE_articles(ARTICLES) {branche?}> retournera
tous les articles de la rubrique actuelle et de ces sous-
rubriques si il y a un id_rubrique dans le contexte,
sinon, tous les articles du site.
```

– {doublons} ou {unique} (ces deux critères sont rigoureusement identiques) permettent d'interdire l'affichage des résultats déjà affichés dans d'autres boucles utilisant ce critère.

historique : A partir de [SPIP 1.2] et jusqu'à [SPIP 1.7.2], seules les boucles **ARTICLES**, **RUBRIQUES**, **DOCUMENTS** et **SITES** acceptaient ce critère.

– {doublons xxxx} à partir de [SPIP 1.8], on peut avoir plusieurs jeux de critères {doublons} indépendants. Les boucles ayant {doublons rouge} n'auront aucune incidence sur les boucles ayant {doublons bleu} comme critère.

– {exclus} permet d'exclure du résultat l'élément (article, brève, rubrique, etc.) dans lequel on se trouve déjà. Par exemple, lorsque l'on affiche les articles contenus dans la même rubrique, on ne veut pas afficher un lien vers l'article dans lequel on se trouve déjà.

– {xxxx IN a,b,c,d} à partir de [SPIP 1.8], limite l'affichage aux résultats ayant le critère xxxx égal à a, b, c ou d. Les résultats sont triés dans l'ordre indiqué (sauf demande explicite d'un autre critère de tri). Il est aussi possible de sélectionner des chaînes de caractères, par exemple avec {titre IN 'Chine', 'Japon'}.

Avec [SPIP 1.9](#), les balises sont admises dans les arguments de IN, et notamment la balise ENV, à laquelle sont appliqués les filtres d'analyse pour assurer que la requête SQL sera bien écrite. De manière dérogatoire, SPIP testera si l'argument de ENV désigne un tableau (venant par exemple de saisies de formulaire dont l'attribut name se termine par []). Si c'est le cas, et si les filtres d'analyse ont été désactivés en suffixant cette balise par une double étoile, alors chaque élément du tableau sera considéré comme argument de IN, SPIP appliquant les filtres de sécurité sur chacun d'eux.

Le squelette standard formulaire_forum_previsu fournit un exemple d'utilisation avec une boucle MOTS ayant le critère {id_mot IN #ENV**{ajouter_mot}} : cette boucle sélectionne seulement les mots-clés appartenant à un ensemble indiqué dynamiquement. Ici, cet ensemble aura été construit par le formulaire du squelette standard choix_mots, qui utilise des attributs name=ajouter_mot[].

– {a,b} où a et b sont des chiffres. Ce critère permet de limiter le nombre de résultats. a indique le résultat à partir duquel on commence l'affichage (attention, le premier résultat est numéroté 0 - zéro) ; b indique le nombre de résultats affichés.

Par exemple {0,10} affiche les dix premiers résultats ; {4,2} affiche les deux résultats à partir du cinquième (inclus).

– {debut_xxx,b} est une variante très élaborée de la précédente. Elle permet de faire commencer la limitation des résultats par une variable passée dans l'URL (cette variable remplace ainsi le a que l'on indiquait précédemment). C'est un fonctionnement un peu compliqué, que fort heureusement on n'a pas besoin d'utiliser trop souvent.

La variable passée dans l'URL commence forcément par debut_xxx (où xxx est un mot choisi par le webmestre). Ainsi, pour une page dont l'URL est :

```
spip.php?page=petition&id_article=13&debut_signatures=200
```

avec un squelette (petition.html) contenant par exemple :

```
<BOUCLE_signatures(SIGNATURES){id_article}{debut_signatures,100}>
```

on obtiendra la liste des 100 signatures à partir de la 201-ième [rappel]. Avec l'URL :

```
spip.php?page=petition&id_article=13&debut_signatures=300
```

on obtient la liste des 100 signatures à partir de la 301-ième [rappel].

– {a,n-b} à partir de [\[SPIP 1.8\]](#), est une variante de {a,b} qui limite l'affichage en fonction du nombre de résultats dans la boucle. a est le résultat à partir duquel commencer à faire l'affichage ; b indique le nombre de résultats à **ne pas afficher** à la fin de la boucle.

{0,n-10} affichera tous les résultats de la boucle sauf les 10 derniers.

– {n-a,b} à partir de [\[SPIP 1.8\]](#), est le pendant de {a,n-b}. On limite à b résultats en commençant l'affichage au a^e résultat avant la fin de la boucle.

Par exemple : {n-20,10} affichera au 10 résultats en partant du 20^e résultat avant la fin de la boucle.

– {a/b} où a et b sont des chiffres. Ce critère permet d'afficher une partie a (proportionnellement) des résultats en fonction d'un nombre de « tranches » b.

Par exemple : {1/3} affiche le premier tiers des résultats. Ce critère est surtout utile pour présenter des listes sur plusieurs colonnes. Pour obtenir un affichage sur deux colonnes, il suffit de créer une première boucle, affichée dans une case de tableau, avec le critère {1/2} (la première moitié des résultats), puis une seconde boucle dans une seconde case, avec le critère {2/2} (la seconde moitié des résultats).

Attention. L'utilisation du [critère {doublons}](#) avec ce critère est périlleuse. Par exemple :

```
<BOUCLE_prem(ARTICLES){id_rubrique}{1/2}{doublons}>
  <li> #TITRE
</BOUCLE_prem>
<BOUCLE_deux(ARTICLES){id_rubrique}{2/2}{doublons}>
  <li> #TITRE
</BOUCLE_deux>
```

n'affichera pas tous les articles de la rubrique ! Imaginons par exemple qu'il y ait au total 20 articles dans notre rubrique. La BOUCLE_prem va afficher la première moitié des articles, c'est-à-dire les 10 premiers, et interdire (à cause de {doublons}) de les réutiliser. La BOUCLE_deux, elle, va récupérer la deuxième moitié des articles de cette rubrique *qui n'ont pas encore été affichés* par la BOUCLE_prem ; donc, la moitié des 10 articles suivants, c'est-à-dire les 5 derniers articles de la rubrique. Vous avez donc « perdu » 5 articles dans l'opération...

Affichage entre les résultats

{ "inter" } permet d'indiquer un code HTML (ici, *inter*) inséré *entre* les résultats de la boucle. Par exemple, pour séparer une liste d'auteurs par une virgule, on indiquera :

```
<BOUCLE_auteurs(AUTEURS){id_article}{", "}>
```

Divers

{logo} permet de ne sélectionner que les articles (ou rubriques, etc) qui disposent d'un logo. Il fonctionne aussi dans la boucle (HIERARCHIE). Le critère inverse {!logo} liste les objets qui n'ont pas de logo.

Notes

[rappel] : le premier résultat est numéroté 0, donc le 200^e résultat représente réellement la 201^e signature

