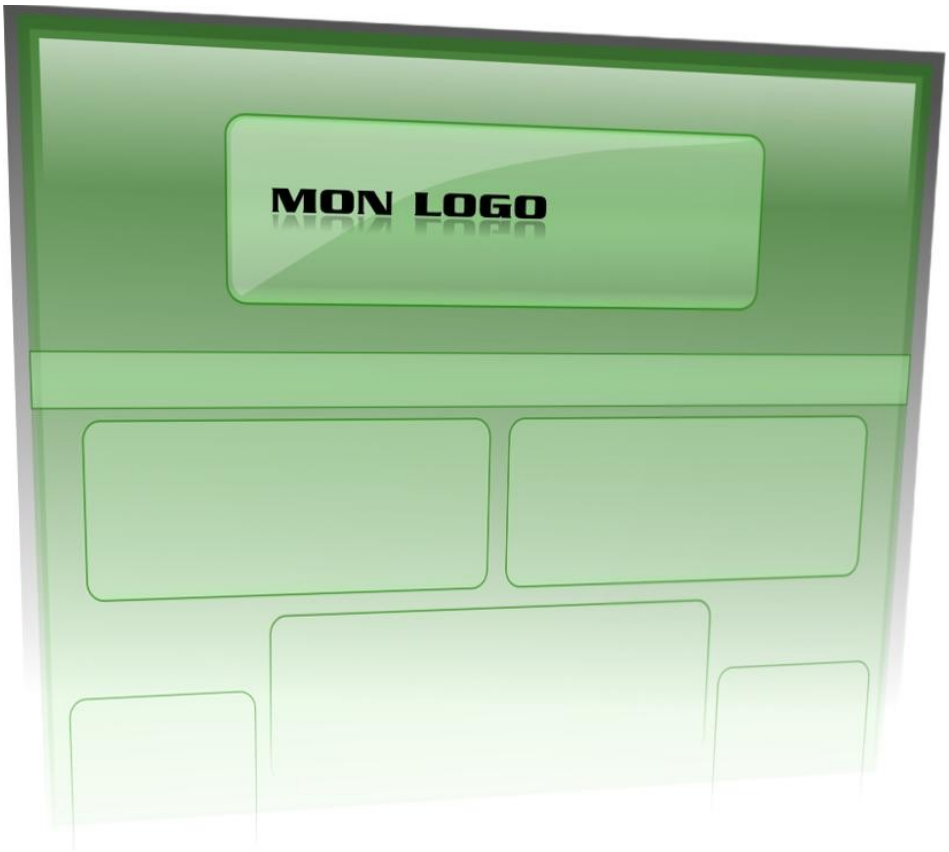
Tutoriel



Créer son template pour Joomla! 1.5

Date : 13/10/08

Auteur : Ced http://www.joomlack.fr.nf

Table des matières

Les annexes sont répertoriées dans un document séparé.

Versions

Béta1 (16/10/08) : première sortie du tuto...

Béta 2 (20/10/08) :
- ajout de couleur pour le code
- rectification de quelques erreurs et ajout d'informations
- ajout des annexes

V1.0 (21/10/08) :
- ajout paramètres RTL
- hacks pour les navigateurs
- mise à jour des annexes

V1.1 (09/11/08) :
- mise à jour de liens dans les annexes
- ajout du récap' pour les nuls
- mise à jour des hacks et div imbriqués
- mise à jour du code de la partie body (il manquait le header et le topmenu)
- ajout de fonctionnalités pour le topmenu

Contributeurs

Sources et liens : Millenniums http://www.annuaire2tutos.com/
Merci aussi à ghazal, cbia, et tous les autres qui me donnent un retour sur mon travail...

Avant de commencer

Je suppose que avez toutes les connaissances nécessaires en HTML, PHP et CSS. Si ce n'est pas le cas, prenez d'abord quelques jours pour apprendre tout ça sur internet...ben oui faites comme moi !

Tout d'abord certains d'entre vous jugeront ce tutoriel simpliste ou incomplet. Dites-vous qu'il s'adresse à ceux qui veulent créer leur template, mais qu'il n'aborde certainement pas tous les aspects de ce sujet assez vaste.

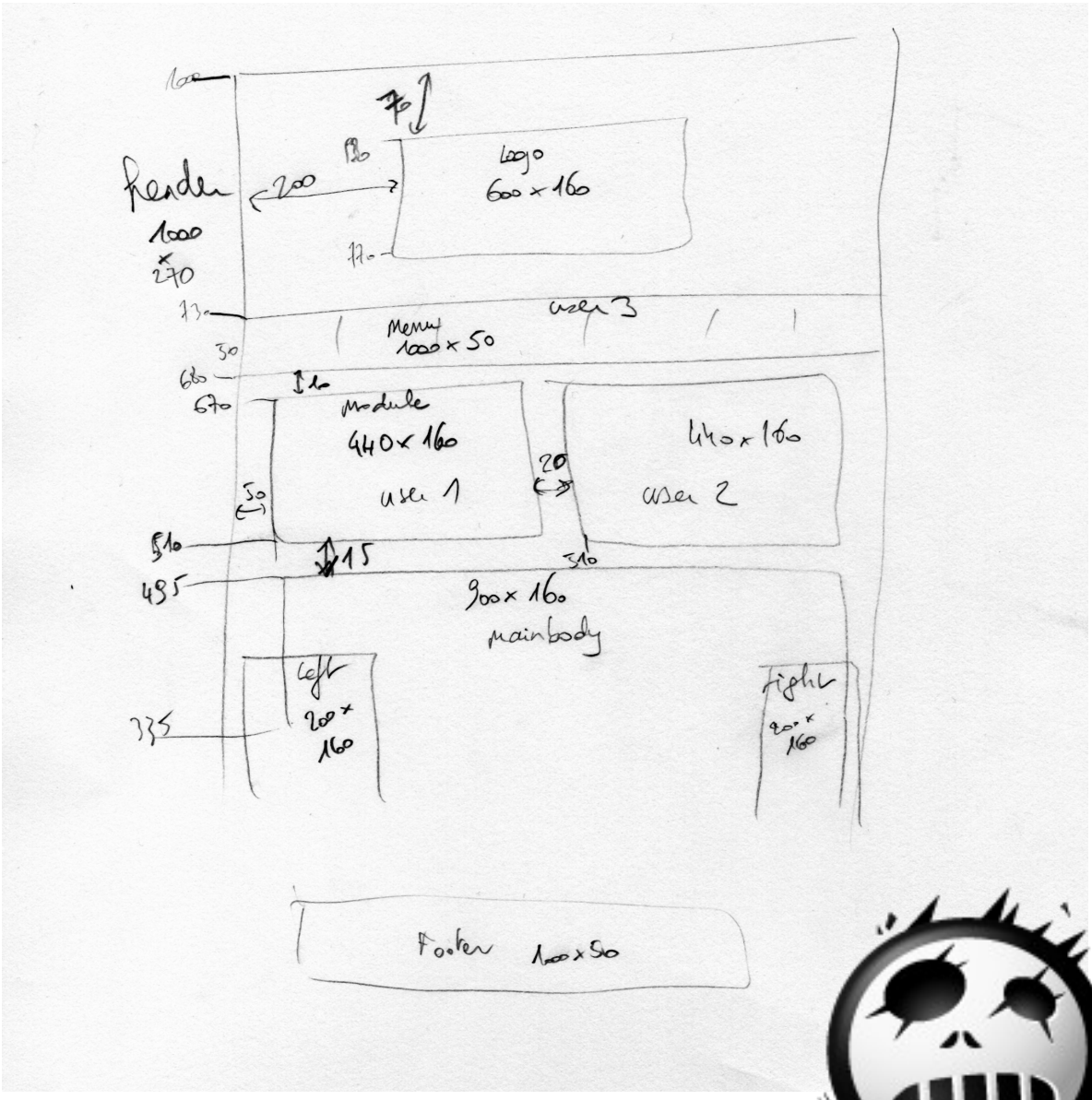
Je vais toutefois essayer de vous donner toutes les billes pour arriver à faire quelque chose de plus complexe. Dans les annexes sont ajoutées toutes les sources que j'ai trouvées et utilisées.

Alors allons-y !!

Créer son modèle : Architecture et design

L'idée

La première chose à faire avant de commencer son template est de trouver l'idée. C'est cette idée qui va nous guider tout au long de la construction. Une petite ébauche au crayon sur une feuille de papier est la bienvenue !



Voilà! Nous avons notre idée ! Vous la trouvez pas belle mon ébauche?
À partir de là commence le travail long et fastidieux de la création graphique. C'est très important de faire un site de bonne qualité, agréable à regarder et pas trop compliqué. Pour y arriver il existe des règles de design à respecter, mais ce n'est pas le but de l'exercice ici.

Concentrons nous sur notre projet....Nous voulons donc obtenir quelque chose de simple, avec

- 2 modules au dessus du corps pour y insérer des news, des statistiques, les derniers articles...etc
- 1 partie sur toute la largeur qui permet d'afficher le logo ou la bannière (sans module)

- 1 module pour afficher le top menu
- 1 module à gauche
- 1 module à droite
- 1 partie centrale pour le corps du site
- 1 module en bas de page pour le pied de page

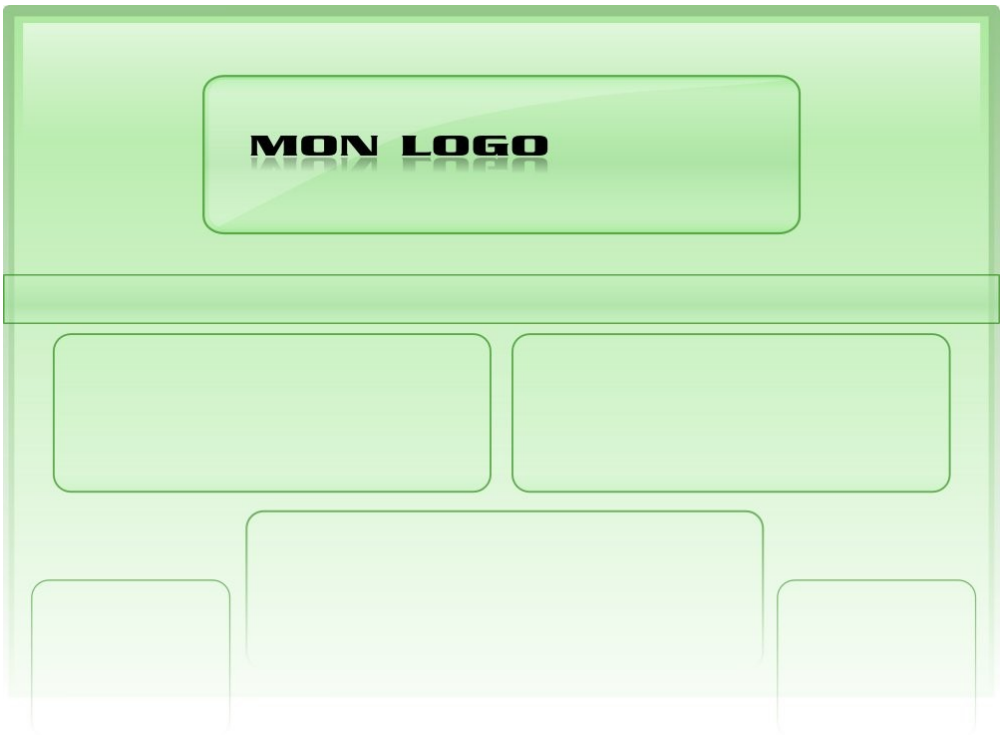
L'idée principale du projet c'est de créer un template très simple avec un minimum de fonctionnalités. Il nous servira à comprendre comment ça fonctionne pour acquérir l'expérience nécessaire à l'élaboration de projets beaucoup plus ambitieux.

La création graphique

Pour construire le modèle, il nous faut un outil de création graphique. Le mieux, c'est d'utiliser un outil vectoriel. Et comme nous faisons partie de la communauté du libre, utilisons Inkscape qui est très pratique.

On commence par une feuille vierge de largeur 1000 pixels et de longueur 1500 pixels. Maintenant il va falloir donner tout ce qu'on a pour dessiner notre site !

Voici le résultat :



Les tailles choisies sont les suivantes :

- Logo : 600x160 px
- topmenu : 1000x50 px --> user3
- les 2 modules : 440x160 px --> user1 + user2
- le corps : 900x160 px
- module gauche et droite : 200x160 px --> left + right
- pied de page : 1000 x50 px --> footer

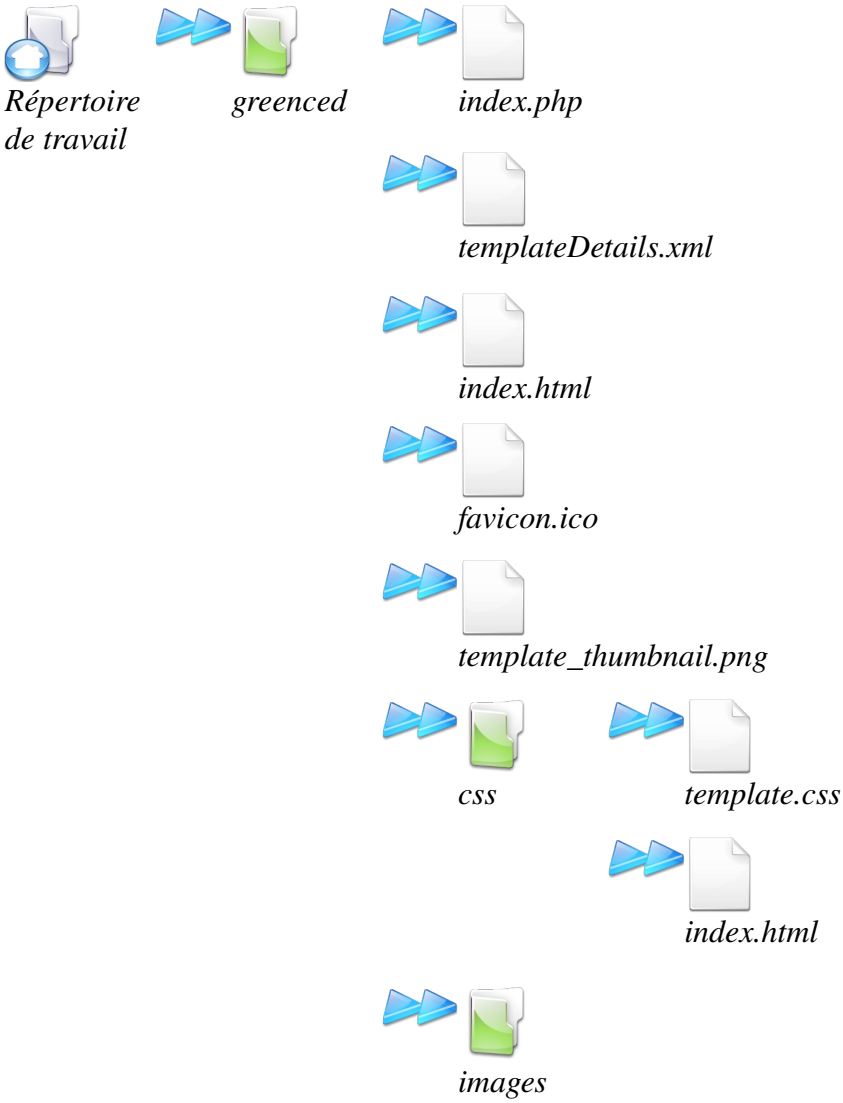
Bien sûr, mon graphisme n'est pas au top et ne sera probablement jamais utilisé.... c'est pas grave je veux juste vous montrer comment intégrer ce truc vert sur notre site !

Les modules user1 et user2

Pour les deux modules on va un corser la chose, histoire de rendre notre template un peu plus fonctionnel.
Si les deux modules ont quelque chose à afficher, on prend deux encadrés de 440x160 px comme définis juste avant.
Par contre si un seul des deux modules affiche quelque chose, on mettra l'autre en pleine page, donc avec une image de 900x160 px.

Préparer la structure

Commençons tout de suite par donner un peu d'ordre dans notre projet. Nous allons créer et classer les fichiers qui vont nous servir :



Expliquons un peu l'utilité de chacun d'eux :

- index.php : c'est le fichier dans lequel nous allons créer la structure du site
- templateDetails.xml : il sert à l'installation du template et aux paramètres (pour les templates paramétrables)
- index.html : feuille vierge qui sert de support à la génération de la page
- favicon.ico : icône du site (dans la barre des tâches)
- template_thumbnail.png : image miniature de prévisualisation du site (140x90 px)
- template.css : fichier qui va piloter tous les styles CSS du site

Voilà, maintenant nous allons pouvoir commencer à travailler.

Création des fichiers

templateDetails.xml

Comme nous l'avons vu juste avant, il sert à l'installation et au paramétrage du template.
Nous allons donc retrouver à l'intérieur les sections suivantes :

- Description du template
- Fichiers à copier lors de l'installation
- Paramètres du template (ici nous n'en aurons pas)
- Positions à charger dans le template

Note importante ! N'oubliez pas d'écire le « D » en majuscules (templateDetails.xml).

Nous avons vu au dessus les fichiers dont nous aurons besoin. Il nous suffit maintenant de les inscrire dans le fichier. Voilà ce que ça donne :

```
<?xml version="1.0" encoding="utf-8"?>
<install version="1.5" type="template">
	<name>greenced</name>
	<version>1.0.0</version>
	<creationDate>12/10/08</creationDate>
	<author>CEd</author>
	<authorEmail>monemail@mondomaine.com</authorEmail>
	<authorUrl>www.monsite.com</authorUrl>
	<copyright></copyright>
	<license>GNU/GPL</license>
	<description>Un design simple de couleur verte pour nous apprendre à créer un
template</description>
<files>
	<folder>images</folder>
	<folder>css</folder>
	<filename>index.php</filename>
	<filename>templateDetails.xml</filename>
	<filename>template_thumbnail.png</filename>
</files>
	<positions>
		<position>user1</position>
		<position>user2</position>
		<position>user3</position>
		<position>left</position>
		<position>right</position>
		<position>footer</position>
	</positions>
</install>
```

Bon...pas besoin de vous faire un schéma, c'est assez clair, non?

Bon allez, je vous fait un petit récap' ?
De la première ligne jusqu'à </description> vous comprendrez qu'il suffit de remplir vos informations (nom, prénom, surnom, adresse, blabla...).
Ensuite on attaque le chargement des fichiers, on va indiquer lesquels doivent être copiés pendant l'installation pour faire fonctionner notre chef d'oeuvre. Ensuite on chargera les positions.

- La balise <files></files> définit le début et la fin des fichiers à copier.
- La balise <folder> indique simplement le dossier à copier avec tous ses fichiers à l'intérieur. Très pratique si on a un max de fichiers et qu'on ne veut pas tous les lister! Comme pour les images.
- La balise <filename> pemet de spécifier un fichier unique à copier.
- La balise <positions></positions> définit le début et la fin des positions du template à charger. Ce sont celles-ci qui seront disponibles dans l'administration pour y placer les modules par exemple. On peut y mettre ce qu'on veut, pourvu que ça corresponde à quelque chose qu'on utilise ensuite dans notre index.php.
- La balise <position> indique la position qu'on choisit d'ajouter.

index.php

Définir son contenu

Définir son contenu....ben oui, avant de mettre quoi que ce soit à l'intérieur du fichier, il faut d'abord définir ce que l'on veut y mettre !

On va y retrouver plusieurs parties :

- la partie HEAD
  - le copyright
  - l'appel aux références externes (autres fichiers php, css, javascript)
  - la définition de certains paramètres (largeur des modules, suivant s'ils sont chargés ou pas...c'est pas très clair je sais, mais on verra ça avec les positions user1 + user2)
- la partie BODY
  - la structure html du site

Partie HEAD

Nous allons y mettre le copyright de la licence GNU/GPL pour partager notre travail avec la communauté. Ensuite nous allons charger les références externes, ici on ira chercher le fichier template.css. Pour finir on mettra quelques petites lignes de php qui définiront la largeur des modules user1 et user2. Je m'explique (on en a déjà un peu parlé dans la partie graphisme...) :

- si on charge un module dans user1 ET un autre dans user2,

  alors chaque module aura une largeur de 440 px

- si on charge un module dans user1 OU un autre dans user2,

  alors le module aura toute la largeur, soit 900 px.

Et là vous me dites mais comment on fait? Regardez ça :

```
<?php
// on compte le nombre de modules à charger pour leur donner la largeur
$moduser1 = $this->countModules( 'user1' );
$moduser2 = $this->countModules( 'user2' );
if ( $moduser1 && $moduser2 ) {
        $divid = '-deux';
        } elseif ( $moduser1 ) {
        $divid = '-un';
        } elseif ( $moduser2 ) {
        $divid = '-un';
        } else {
        $divid = '';
 }
?>
```

Pour résumer, les variables moduser1 et 2 contiennent le nombre de modules à charger dans chacun d'eux. Ensuite on teste, y a-t-il quelque chose dans les deux? Si oui on ajoute le suffixe -deux, si seulement un des deux on ajoute -un, et sinon rien.

Alors là on se demande...ça sert à quoi les suffixes? On va tout simplement les utiliser pour identifier les objets avec le CSS. Vous regarderez plus tard le fichier CSS de notre tutoriel et vous verrez qu'on y retrouve nos suffixes !

CODE :

```
<?php
/**
 * @copyright Copyright (C) 2005 - 2007 Open Source Matters. All rights reserved.
 * @license GNU/GPL
 * Joomla! is free software. This version may have been modified pursuant
 * to the GNU General Public License, and as distributed it includes or
 * is derivative of works licensed under the GNU General Public License or
 * other free or open source software licenses.
 */

// no direct access
defined( '_JEXEC' ) or die( 'Restricted access' );
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

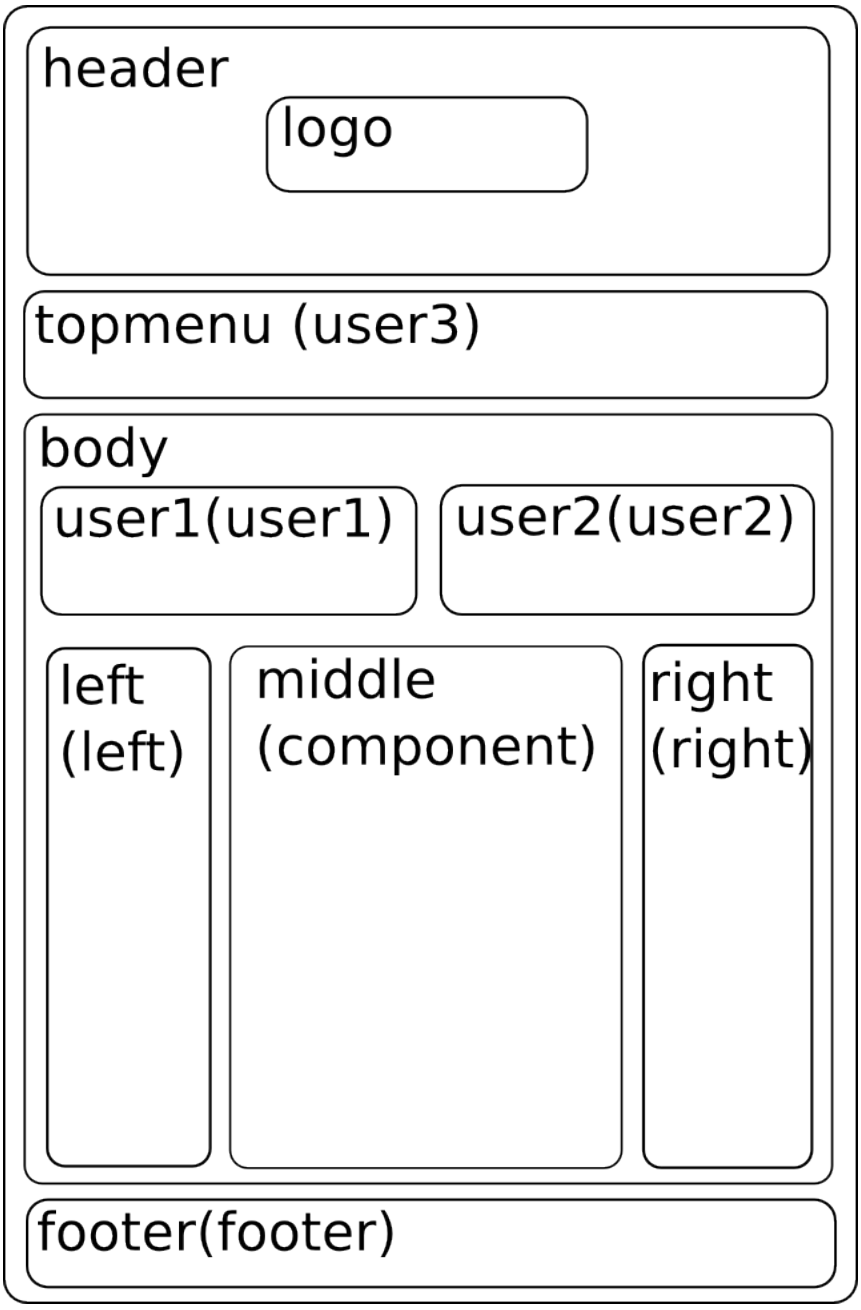
```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="<?php echo $this->language; ?>"
lang="<?php echo $this->language; ?>"  dir="<?php echo $this->direction; ?>" >

<?php
// on compte le nombre de modules à charger pour leur donner la largeur
$moduser1 = $this->countModules( 'user1' );
$moduser2 = $this->countModules( 'user2' );

if ( $moduser1 && $moduser2 ) {
        $divid = '-deux';
        } elseif ( $moduser1 ) {
        $divid = '-un';
        } elseif ( $moduser2 ) {
        $divid = '-un';
        } else {
        $divid = '';
}
?>

<head>
<jdoc:include type="head" />

<!-- --------- on charge les références externes (d'abord les fichiers systèmes par défaut) -------- -->
<link rel="stylesheet" href="<?php echo $this->baseurl ?>/templates/system/css/system.css"
type="text/css" />
<link rel="stylesheet" href="<?php echo $this->baseurl ?>/templates/system/css/general.css"
type="text/css" />

<link rel="stylesheet" href="<?php echo $this->baseurl ?>/templates/<?php echo $this->template?
>/css/template.css" type="text/css" />

</head>
```

Partie BODY

Pour appeler les fonctionnalités du site, c'est à dire les modules à charger dans les différents endroits de notre template, nous utiliserons la fonction suivante :
<jdoc:include type="modules" name="user3" style="xhtml" />

Elle permet d'appeler (dans notre exemple) un module qui s'appelle user3 (en référence à la position que nous avons créée) qui utilisera un style xhtml. Ce style signifie simplement que le module sera encapsulé dans 1 div. Pour plus d'informations, jetez un coup d'oeil à l'annexe !

Définissons la manière dont on va afficher les positions :



on pourra maintenant créer nos DIV avec la bonne structure sans problème:
div header
	div logo
div topmenu
div body
	div user1
	div user2
	div modleft
	div bodymiddle
	div modright
div modfooter

En gros, voici donc l'idée. Il va tout simplement falloir écrire tout ça en php et html ! Mais ça, ça ne devrait pas vous poser de problème...

Comment on fait?

On va utiliser les principes suivants :

1) Appliquer un id aux DIV pour leur appliquer un style CSS par la suite.

Exemple :
On identifiera le DIV avec #monid dans le fichier CSS.
On peut aussi appliquer une classe
Exemple :
On identifiera le DIV avec .maclasse dans le fichier CSS.

La différence entre les deux c'est qu'un ID est unique, alors qu'on peut appliquer la même classe à plusieurs objet (DIV). Ca peut servir si on veut par exemple formater plusieurs DIV de la même manière.
On peut très bien ajouter un id **ET** une classe à un élément html.

2) Vérifier la présence du module pour charger sa position

Si on met en place la position DIV alors que le module est vide, on va se retrouver avec un template assez moche... voilà donc l'astuce :

```
<?php if($this->countModules('user1')) : ?>
		<div id="user1"><jdoc:include type="modules" name="user1" style="xhtml"
/></div>
<?php endif; ?>
```

En clair, y a-t-il quelque chose à charger dans user1? Si oui créer la position, sinon rien.

3) On finit par appliquer un style à nos modules. Un style ça sert à quoi? Ben ça va tout simplement définir la manière dont le module va être structuré. Consulter l'annexe pour plus de détails.

CODE :

```
<body>
<div id="container">

<!-- début du header -->
<div id="header">
```

```
    <div id="logo"></div>
</div>
<!-- fin du header-->

<!--début du topmenu-->
<div id="topmenu"><jdoc:include type="modules" name="user3" style="xhtml" /></div>
<!--fin du topmenu-->

<!--début du body-->
<div id="body">

        <!--si user1 ou user2, on les affiche-->
        <?php if($this->countModules('user1')) : ?>
                <div id="user1<?php echo $divid; ?>"><jdoc:include type="modules" name="user1"
style="rounded" /></div>
        <?php endif; ?>
        <?php if($this->countModules('user2')) : ?>
                <div id="user2<?php echo $divid; ?>"><jdoc:include type="modules" name="user2"
style="rounded" /></div>
        <?php endif; ?>

        <!--si left on l'affiche-->
        <?php if($this->countModules('left')) : ?>
                <div id="modleft"><jdoc:include type="modules" name="left" style="xhtml"
/></div>
        <?php endif; ?>

        <!-- on affiche le corps du site -->
        <div id="bodymiddle">
                <div id="message"><jdoc:include type="message" /></div>
                <div id="component"><jdoc:include type="component" style="xhtml" /></div>
        </div>

        <!--si right on l'affiche-->
        <?php if($this->countModules('right')) : ?>
                <div id="modright"><jdoc:include type="modules" name="right" style="xhtml" /></
div>
        <?php endif; ?>
</div>
<!--fin du body-->

<!--début du footer-->
<div id="modfooter"><jdoc:include type="modules" name="footer" style="none" /> Designed by
<a href="http://www.monsite.com" title="CEd" target="_blank">CEd</a></div>
<!--fin du footer-->

</div>
</body>
<!-- fin du body -->
</html>
```

template.css

Haaa... enfin. On va pouvoir créer les styles CSS qui vont piloter le graphisme ! C'est maintenant que tout va se jouer.
Tout d'abord on retourne faire un petit tour sous Inscape, notre logiciel de dessin. On va découper nos images pour les insérer dans le site. Voici comment j'ai découpé les miennes :

header.png	1000x730 px	c'est le fond dégradé du site
logo.png	600x160 px	facile ça...c'est le logo
menu.png	1000x50 px	le fond du topmenu

les 3 images qui forment les coins arrondis des modules user1 et user2 si les deux sont chargés :

moduletoptop440.png	440x130 px	c'est la tête de notre module
moduletopmiddle440.png	440x1 px (c'est une image qui sera répétée)	corps du module
moduletopbottom440.png	440x29 px	c'est le pied de notre module

NOTE : la hauteur de l'image de tête devrait être de 20 ou 30 px (au lieu de 130 px) à confirmer, car je n'ai pas encore eu le temps de tester.

les 3 images qui forment les coins arrondis des modules user1 et user2 si un des deux est chargé

moduletoptop900.png	900x130 px	c'est la tête de notre module
moduletopmiddle900.png	900x1 px (image répétée encore une fois)	corps du module
et moduletopbottom900.png	900x29 px	c'est le pied de notre module

module.png	200x160 px	c'est le bord des modules du site
body500.png	520x160 px	c'est le bord du corps

Allez, je vous donne le code. Essayez d'y jeter un petit coup d'oeil, vous verrez c'est très simple.
Bien sur on peut largement le compléter pour définir l'allure des textes et tout un tas de choses.
Si vous cherchez la liste des styles CSS, regardez encore un peu dans l'annexe.

CODE :

```
*
{
	margin: 0;
	padding: 0;
}

body
{
	background: none;
	color: #000000;
	font-family: arial, helvetica, sans-serif;
	font-size: 100.1%;
	padding: 0px;
}
```

```
#container {
	width:1000px;
	height:730px;
	background: url(../images/header.png) no-repeat center;
	margin-left:auto;
	margin-right:auto;
	margin-top:0px;
}

#logo {
	background: url(../images/logo.png) no-repeat center;
	margin-left:200px;
	padding-top:130px;
	height:160px;
	width:600px;
}

#topmenu {
	background: url(../images/menu.png) no-repeat center;
	margin:0px;
	height:50px;
}

#user1-deux div,#user2-deux div{
	background: url(../images/moduletoptop440.png) no-repeat top;
	padding:0;
	margin-top:10px;
	margin-left:40px;
	padding-top:20px;
	width:440px;
	float:left;
}

#user1-deux div div,#user2-deux div div{
	background: none;
	margin:0;
	padding:0;
}

#user1-deux div div div,#user2-deux div div div{
	background: url(../images/moduletopbottom440.png) no-repeat bottom;
	margin-top:0px;
	margin-bottom:0px;
	padding-bottom:15px;
}

#user1-deux div div div div,#user2-deux div div div div{
	background: url(../images/moduletopmiddle440.png) left;
	padding-left:25px;
```

```
	padding-right:15px;
	margin-top:0px;
	width:400px;
	height:auto;
}

#user1-un div,#user2-un div{
	background: url(../images/moduletoptop900.png) no-repeat top;
	padding:0;
	margin-top:10px;
	margin-left:40px;
	padding-top:20px;
	width:900px;
	float:left;

}

#user1-un div div,#user2-un div div{
	background: none;
	margin:0;
	padding:0;
}

#user1-un div div div,#user2-un div div div{
	background: url(../images/moduletopbottom900.png) no-repeat bottom;
	margin-top:0px;
	margin-bottom:0px;
	padding-bottom:15px;
}

#user1-un div div div div,#user2-un div div div div {
	background: url(../images/moduletopmiddle900.png) left;
	padding-left:25px;
	padding-right:15px;
	margin-top:0px;
	width:860px;
	height:auto;
}

div#topmenu div.moduletable_topmenu table tbody tr td {
	padding-top:10px;
	padding-left:25px;
}

div#topmenu div.moduletable_topmenu table tbody tr td a.mainlevel{
	padding-left:10px;
}

#bodymiddle {
	background: url(../images/body500.png) no-repeat top;
```

```
	width:480px;
	padding-left:20px;
	padding-right:20px;
	padding-top:15px;
	float:left;
	margin-top:10px;
	margin-left:10px;
}

#modleft {
	width:200px;
	clear:both;
	float:left;
	padding-left:30px;
	padding-top:10px;
	margin-top:50px;
}

#modright {
	width:200px;
	float:right;
	padding-right:30px;
	padding-top:10px;
	margin-top:50px;
}

#modfooter {
	width:100%;
	float:left;
	clear:both;
}

div.moduletable,div.moduletable_text,div.moduletable_toto2,div.moduletable_menu {
	background: url(../images/module.png) no-repeat top;
	padding-top:10px;
	padding-left:25px;
	padding-right:5px;
	padding-bottom:10px;
	margin-bottom:15px;
}
```

Finalisation

Bon voilà, c'est fini pour aujourd'hui. Nous avons créé tous les fichiers nécessaires pour faire fonctionner notre template. Il reste les petits détails, c'est à dire la miniature template_thumbnail.png et favicon.ico.
Je vous laisser faire ça, une capture d'écran pour la miniature, et pour l'icône, libre à votre imagination !

Une fois que tout ça est fait, on va clore le sujet et créer une archive .zip dans lequel on met tous nos fichiers. Et c'est cette archive qu'on va utiliser pour l'installation du template.

Allez, bon courage pour vos futures réalisations, n'oubliez pas qu'il faut un peu de patience pour arriver au résultat.

CEd

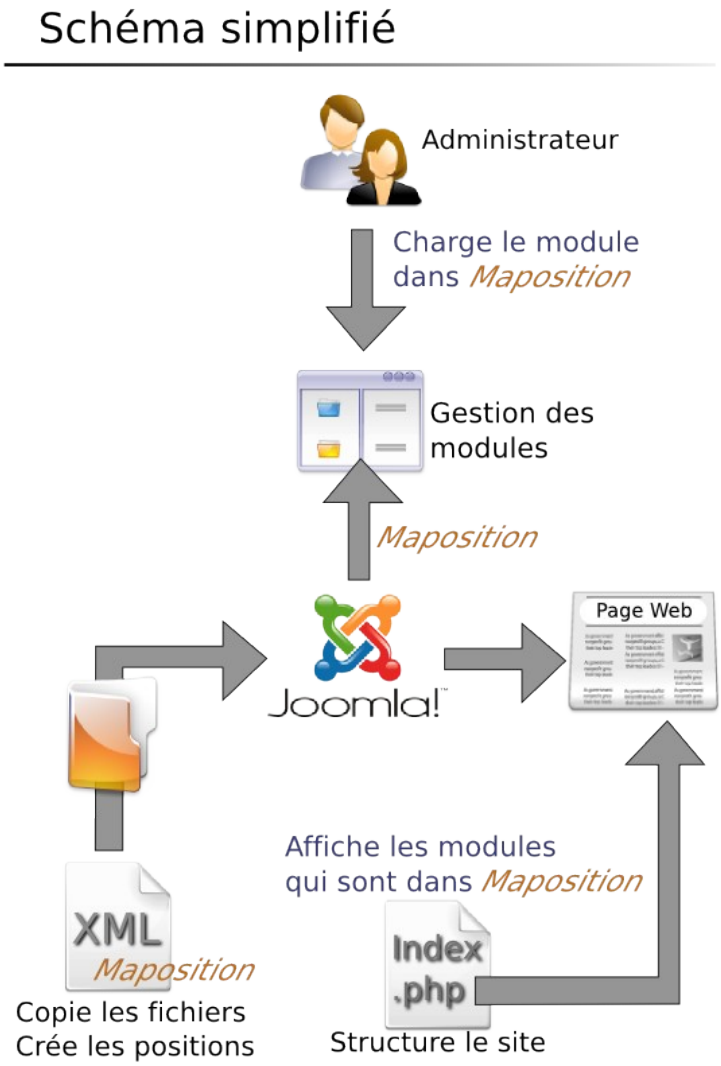
Le récap' pour les nuls

FICHIER XML
Crée les positions qui seront ensuite utilisées dans la gestion des modules de la partie administration de Joomla!. Exemple, on crée une position *MAPOSITION*

INDEX.PHP
Crée l'architecture html définit une zone dans laquelle charger le module *MAPOSITION*
Exemple: inlcude...blabla *MAPOSITION*

TEMPLATE.CSS
Formate l'élément html pour lui donner l'allure et la position, mais aussi le comportement (fixe, variable, ...)
exemple: #montest {margin-left:100px;width:300px; ....etc}

DANS JOOMLA!
Gestion des modules, tu charges un module dans la position *MAPOSITION* que tu as créée et qui est appelée dans le DIV et qui a l'allure définie dans le CSS



Mode Avancé

Template RTL (Right To Left)

Ben oui, tout le monde ne lit pas de gauche à droite. Pour ça aussi on peut prévoir un fichier CSS spécifique. On insère le code suivant :
d'abord on définit la variable direction (déjà présent dans le head de mon index.php)

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="<?php echo $this->language; ?>"
lang="<?php echo $this->language; ?>"  dir="<?php echo $this->direction; ?>" >
```

ensuite on va charger le fichier CSS approprié si c'est du RTL

```
<?php if($this->direction == 'rtl') : ?>
        <link rel="stylesheet" href="<?php echo $this->baseurl ?>/templates/<?php echo $this-
>template ?>/css/template_rtl.css" type="text/css" />
<?php endif; ?>
```

A charger à la fin des autres fichiers CSS.
Pour modifier la direction du texte dans le fichier CSS, utilisons la commande suivante :

```
direction: rtl;
```

Il faut aussi changer l'alignement des modules, de la page en général. Ce qui était à gauche passe à droite, et vice-versa. C'est en fait un effet miroir. Il faut le faire si vous destinez votre site à l'utilisation internationale. L'information de direction vient directement du navigateur de la personne qui visite le site.

« Hacks » pour les navigateurs

Hou la la! Pas d'inquiétudes, on ne va faire de mal à personne! On va juste adapter notre template pour que le site soit lisible sur tous les navigateurs. Il existe plusieurs méthodes

- des hacks
- des méthodes conditionnelles pour traiter les infos css dans un fichier spécifique
- les div imbriqués

Méthode conditionnelle

On va lui dire, si le navigateur est IE7 (par exemple), charge moi le template approprié.
Voilà comment on fait ça :

```
<!--[if IE 7]>
                <link href="<?php echo $this->baseurl ?>/templates/<?php echo $this->template?
>/css/ie7only.css" rel="stylesheet" type="text/css" />
<![endif]-->
```

En clair on définit un fichier CSS par navigateur que l'on veut gérer. Faut juste penser à le mettre après le template.css de base pour qu'il soit prioritaire (le dernier fichier lu est celui qui a raison en cas de conflit).
On peut encore aller plus loin et lui dire de l'appliquer pour tous les navigateurs de versions

supérieure ou inférieure à celle citée. Par exemple :

```
<!--[if lte IE 6]>
                <link href="<?php echo $this->baseurl ?>/templates/<?php echo $this->template?
>/css/ieonly.css" rel="stylesheet" type="text/css" />
<![endif]-->
```

Ici on lui demande si le navigateur est inférieur ou égal à IE6.
On retrouve toutes les possibilités dans le tableau suivant :

Opérateur	**Description**
!	N'est pas égale
lt	Inférieure
lte	Inférieure ou égale
gt	Supérieure
gte	Supérieure ou égale

Autres exemples

```
<!--[if IE]>
<div>C'est internet explorer</div>
<![endif]-->

<!--[if !IE]><!-->
<div>C'est pas internet explorer</div>
<!--<![endif]-->
```

Hacks

Oui je sais, les hacks c'est PAS BIEN. Malheureusement on peut parfois en avoir besoin pour se sortir d'une situation catastrophe, et vaux mieux les avoir sous la main. Après si on peut s'en passer, c'est avec plaisir !
Je ne vous cache pas que j'ai tiré tous ces exemples du web, je les introduit dans le tuto pour qu'on ait la ressource sous les yeux.
Voyons donc un peu ce que ça donne :

IE6 et inférieur :

```
#bloc {border: green;}
*html #bloc {border: red;} /* only <=ie6 */
```

IE7 :

```
#bloc {border: green;}
*+html #bloc {border: red;} /* only ie7 */
```

Gecko (Firefox, Netscape) :

```
#bloc {border: green;}
html:not([lang*=""]) #bloc {border: red;} /* only gecko */
```

Opera :

```
#bloc {border: green;}
html:first-child #bloc {border: red;} /* only opera */
```
Attention n'est plus valable sur Opéra >8 et est reconnu par Safari 3.

Safari (Webkit) :
```
#bloc {
color: green;
}
/*\*/
html*#bloc {
color:red;/*only safari*/
_color:green;/*only IE6 PC*/
}/**/
```
Attention, dans ce dernier cas il faut repréciser le style pour IE6

Un deuxième hack plus simple pour cibler Safari 2 (valide CSS3) :
```
body:last-child:not(:root:root) #bloc {
color: red; }/* only Safari 2 */
```

et enfin pour Safari 3 et les nouvelles versions de Webkit (valide CSS3):
(On doit utiliser un filtre @media)
```
@media all and (min-width: 0px) {
body:not(:root:root) #bloc {
color: red; }/* only webkit */
}
```

Forcer la valeur pour IE :
Quand IE ne veut pas comprendre, on peut aussi tenter de lui forcer l'application du style CSS en faisant de la sorte :
```
#bloc {border: green !important;}
```

L'underscore :
Firefox ne reconnait plus l'underscore, on peut donc l'utiliser pour IE.
```
margin-right: 100px;
_margin-right: 150px;
```
donnera un retrait de 100 pixels sous Firefox et un de 150 pixels sous Explorer. A utiliser avec prudence...

Voilà, nous avons vu une bonne partie de ce que l'on peut utiliser. Si vous rencontrez des diffcultés ou que vous voulez compléter vos connaissances, je vous invite tout simplement à faire un tour sur le web. Mais attention à toujours respecter les règles de validation au maximum!

DIV imbriqués

Une méthode pour éviter de trop jouer avec les fameux hacks, consiste à utiliser des DIV imbriqués. En effet le gros problème que l'on rencontre , c'est que
avec Firefox : la largeur dune div EXCLUT padding et border
avec IE : la largeur d'un block INCLUT padding et border (de la div)

Désolé, mais je n'ai pas plus d'informations à donner pour l'instant, ce chapitre sera complété prochainement. En attendant vous pouvez retrouver la discussion originale sur le sujet sur le forum joomla.fr.
http://forum.joomla.fr/showthread.php?t=76005&p=355774

Le topmenu

Pour éviter d'avoir la bordure des modules qui apparaît sur le topmenu, il faut renommer le suffixe de classe de module dans la gestion des modules de la partie administration de Joomla!.
On peut simplement renommer « _menu » en « _topmenu ».
Si vous ne comprenez pas à quoi ça sert, faites le test et vous verrez !

Dans le fichier CSS on retrouve

```
div.moduletable, div.moduletable_text, div.moduletable_toto2, div.moduletable_menu {
	background: url(../images/module.png) no-repeat top;
}
```

qui attribue une image de fond que nous ne voulons pas pour notre top menu. Du coup on donne un suffixe différent à notre module de topmenu et on peut le piloter en appelant

```
div.moduletable_topmenu {
}
```

Par exemple, pour rendre tout ça un peu plus joli on peut mettre :

```
div.moduletable_topmenu li {
	display:block;
	float:left;
	padding-left:10px;
}
```

Ce qui nous permet d'afficher les titre en lignes avec un espacement de 10 pixels.

Petit ajout de dernière minute, le menu déroulant... Là je sais que plus d'un cherche à en mettre un sur son site et galère pour y arriver. Une solution très simple tirée des tutoriels sur les dropdowns suckerfish. Lorsque l'on passe la souris sur le menu parent, les enfants viennent s'afficher. C'est magique... ben non c'est du CSS !

Voici comment on fait pour appliquer l'idée à notre topmenu :

```
div.moduletable_topmenu ul li ul {
	position:absolute;
	left:-999em;
}

div.moduletable_topmenu ul li:hover ul {
	left:auto;
}
```

Je vois déjà les yeux qui s'agrandissent, « Quoi? C'est tout ? », ben oui. Une micro explication s'impose.
En temps normal, le menu enfant est ramené totalement hors de la fenêtre, et la position absolute permet de le désolidariser du reste du menu.
Lorsque la souris passe sur le parent, on ramène le menu enfant à sa position d'origine automatiquement.
Pfff, suffisait d'y penser ! Et tout de suite notre menu rajoute une touche assez classe à notre page web.