

Exemple de conception d'un site dynamique

par [Pierre-Baptiste Naigeon](#)

Date de publication : 18 Mai 2006

Dernière mise à jour : 18 Mai 2006

Le but de ce tutorial est de vous montrer un exemple de conception de site dynamique. Bien entendu, rien ne vous empêche de ne pas suivre cette structure, et de coder votre site autrement.

- I - Définition de la structure du site
 - I-A - Arborescence
 - I-B - Les contraintes
- II - Base de données
 - II-A - Structure de la base
 - II-B - Remplissage de la base
- III - PHP
 - III-A - Fonctions de connexion à la base
 - III-B - Création du template
 - III-C - Détail des fonctions de récupération et d'affichage des données
 - III-D - Gestion de la feuille de style
- IV - Conclusion

I - Définition de la structure du site

I-A - Arborescence

Voici l'arborescence que nous souhaitons créer :

- Accueil
 - Mon corps de rêve
 - Mes abdos Kro
 - Mes mains
 - Mon profil Grec
 - Mes vacances
 - Au ski
 - A la mer
 - Mer du Nord
 - Mer Morte
 - Mes loisirs
 - Me contacter

Bien entendu, ce n'est qu'un exemple, vous pouvez adapter à volonté.

I-B - Les contraintes

Dans le cadre de cet exemple, nous allons définir notre besoin.

- Présentation identique des différentes pages (un seul template).
- Menus générés à la volée.
- Chaque page aura un identifiant unique
- Chaque page aura un titre
- Chaque page aura des mots-clés
- Chaque page aura une description
- Chaque page aura un contenu
- Chaque page aura un et un seul parent

II - Base de données

Passons maintenant à la création de la base de données.

II-A - Structure de la base

Commencez par créer une base nommée BASE_TEST.

Nous allons ensuite créer la table `pages`, en fonction des besoins énoncés ci-dessus :

- Id_page : int auto_increment (clé primaire): identifiant unique
- Titre : varchar(255) : titre
- Mots_cles : varchar(255) : mots-clés
- Description : varchar(255) : description
- Contenu : text : contenu
- Id_parent : int : Identifiant de la page parent

Afin de définir la page 'racine' du site, nous allons déclarer que son Id_parent vaudra 0. Il n'y aura donc qu'une seule page à posséder cet Id_parent.

Code SQL correspondant

```
CREATE TABLE `pages` (  
  `Id_page` INT NOT NULL AUTO_INCREMENT ,  
  `Titre` VARCHAR( 255 ) NOT NULL ,  
  `Mots_cles` VARCHAR( 255 ) NOT NULL ,  
  `Description` VARCHAR( 255 ) NOT NULL ,  
  `Contenu` TEXT NOT NULL ,  
  `Id_parent` INT DEFAULT '1' NOT NULL ,  
  PRIMARY KEY ( `Id_page` )  
);
```

Pour le champ Id_parent, j'ai choisi 1 comme valeur par défaut, afin d'éviter que par mégarde plusieurs pages ne se retrouvent avec soit une valeur nulle soit 0 comme parent.

Au pire donc, pour toutes les pages dont l'Id_parent ne sera pas spécifié, elles seront considérées comme enfant de l'accueil.

Bon, un peu plus d'explications concernant l'Id_parent. Comme un dessin vaut souvent mieux qu'une longue explication, voici notre arborescence représentée avec les Id de chaque page :

Nous allons insérer dans la base les pages par 'niveau' (et de gauche à droite) dans l'arborescence.

Arborescence du site

Comme vous pouvez le constater, chaque page n'a qu'un et un seul parent (exception faite de la page d'accueil), et 0 à n enfants.

Prenons maintenant un exemple pour bien comprendre le système de parenté : "Mer Morte" a pour Id 12, et comme parent "A la mer" qui elle a pour Id 10. L'Id_parent de "Mer Morte" sera donc l'Id de "A la mer" soit 10. En remontant encore d'un cran, "A la mer" a pour parent "Mes vacances" (Id=3). L'Id_parent de "A la mer" est donc 3. Et faisons de même pour le dernier niveau : "Mes vacances" a pour parent "Accueil" (Id=1). L'Id_parent de "Mes vacances" est donc 1.

La page d'accueil est la seule en orange, car elle subira un traitement particulier, du fait qu'elle est la seule à ne pas avoir de parent.

Les pages en vert sont les pages 'feuilles', n'ayant pas d'enfant. Les pages en bleu et en rose représentent les différents niveau de profondeur.

Pour le champ Contenu, Deux solutions s'offrent à nous pour le remplissage :

- Définir ce champ en varchar, et stocker uniquement l'adresse physique sur le serveur de la page à inclure. A ce moment là, il faudra remplacer l'affichage du contenu en provenance de la base de données par un include du chemin extrait du contenu de la DB.
- Laisser ce champ en text (comme c'est le cas ici), et inclure directement le code HTML dans la base de données.

Quelle que soit la solution choisie, il ne sera pas utile de remettre les entêtes HTML dans le contenu. Le seul contenu de la balise BODY nous interesse ici (en effet, les entêtes sont déjà définies dans index.php).

Dans le second cas, il va falloir faire très attention à la manière dont les données seront insérées dans la base. Les apostrophes et les guillemets dans notre contenu risquent de nous poser problème, il faudra donc les échapper à l'aide de la fonction [mysql_real_escape_string](#).

II-B - Remplissage de la base

Pour le remplissage de la base, je ne me suis pas embêté : le titre, les mots clés, la description et le contenu seront identiques.

Voici le SQL généré par l'ajout de notre arborescence :

SQL de remplissage de la table

```
INSERT INTO `pages` VALUES (1, 'Accueil', 'Accueil', 'Accueil', 'Accueil', 0);
INSERT INTO `pages` VALUES (2, 'Mon corps de rêve', 'Mon corps de rêve', 'Mon corps de rêve', 'Mon corps de rêve', 1);
INSERT INTO `pages` VALUES (3, 'Mes vacances', 'Mes vacances', 'Mes vacances', 'Mes vacances', 1);
INSERT INTO `pages` VALUES (4, 'Mes loisirs', 'Mes loisirs', 'Mes loisirs', 'Mes loisirs', 1);
INSERT INTO `pages` VALUES (5, 'Me contacter', 'Me contacter', 'Me contacter', 'Me contacter', 1);
INSERT INTO `pages` VALUES (6, 'Mes abdos Kro', 'Mes abdos Kro', 'Mes abdos Kro', 'Mes abdos Kro', 2);
INSERT INTO `pages` VALUES (7, 'Mes mains', 'Mes mains', 'Mes mains', 'Mes mains', 2);
INSERT INTO `pages` VALUES (8, 'Mon profil Grec', 'Mon profil Grec', 'Mon profil Grec', 'Mon profil Grec', 2);
INSERT INTO `pages` VALUES (9, 'Au ski', 'Au ski', 'Au ski', 'Au ski', 3);
INSERT INTO `pages` VALUES (10, 'A la mer', 'A la mer', 'A la mer', 'A la mer', 3);
INSERT INTO `pages` VALUES (11, 'Mer du Nord', 'Mer du Nord', 'Mer du Nord', 'Mer du Nord', 10);
INSERT INTO `pages` VALUES (12, 'Mer Morte', 'Mer Morte', 'Mer Morte', 'Mer Morte', 10);
```

Remarquez bien ici que seul l'accueil à comme Id_parent 0.

III - PHP

III-A - Fonctions de connexion à la base

Créons tout d'abord un fichier nommé mes_fonctions.php. Ce fichier contiendra les différentes fonctions permettant de se connecter et d'interroger la base (et d'autres par la suite) :

mes_fonctions.php

```
<?php

// Se connecte à la DB
// Paramètres : nom de la base -> $name_DB
function connexion_DB($name_DB) {
// Déclaration des paramètres de connexion
    $host = "localhost";
    $user = "root";
    $bdd = $name_DB;
    $passwd = "";
// Connexion au serveur
    mysql_connect($host, $user,$passwd) or die("erreur de connexion au serveur");
    mysql_select_db($bdd) or die("erreur de connexion a la base de donnees");
}

//
// -----
// Deconnection de la DB
function deconnexion_DB() {
    mysql_close();
}

//
// -----
// Exécute une requête SQL. Si la requête ne passe pas, renvoyer le message d'erreur MySQL
// Paramètres : chaîne SQL -> $strSQL
// Renvoie : enregistrements correspondants -> $result
function requete_SQL($strSQL) {
    $result = mysql_query($strSQL);
    if (!$result) {
        $message = 'Erreur SQL : ' . mysql_error() . "<br>\n";
        $message .= 'SQL string : ' . $strSQL . "<br>\n";
        $message .= "Merci d'envoyer ce message au webmaster";
        die($message);
    }
    return $result;
}

?>
```

Pas besoin de plus de détails sur ces fonctions, contentons-nous de les utiliser.

Juste une petite explication sur l'utilité de ce fichier 'mes_fonctions.php'. Ce fichier va être inclus dans notre page 'index.php'. Cela revient au même que si vous aviez codé directement dans le fichier 'index.php'.

J'ai choisi de séparer les fonctions du reste pour plusieurs raisons :

- Fonctions plus faciles à maintenir. On voit clairement où elles sont (et rien n'empêche de créer un fichier par 'type' de fonctions... connexion_DB.php, affichage_header.php, ...).
- Certaines fonctions peuvent être nécessaires dans plusieurs pages du site. Les inclure dans un fichier spécifique permet donc de ne pas avoir à les recopier dans chaque page (et donc une seule fonction à maintenir, CQFD).

Les fonctions définies dans ce fichier seront donc appelées depuis notre page principale, sans que cela ne pose le moindre problème.

III-B - Création du template

L'intérêt de ce tutorial étant tout autre, nous n'allons pas nous étaler sur la création du template.

Définissons ici simplement nos besoins en terme d'affichage :

- Un menu 'racine' horizontal en permanence accessible en haut de la page
- Un 'chemin de fer', permettant de savoir en permanence l'endroit où l'on se situe dans le site
- Un menu vertical à gauche correspondant à la position actuelle
- Le contenu de la page.
- Un pied de page commun à l'ensemble des pages du site

L'idée va simplement être de créer une seule et unique page PHP qui chargera dynamiquement les différents éléments dont elle a besoin pour son affichage.

Nommons cette page index.php :

```
index.php
<?php
    // Active tout les warning. Utile en phase de développement
    // En phase de production, remplacer E_ALL par 0
    error_reporting(E_ALL);

    // Inclus le fichier contenant les fonctions personnalisées
    include_once 'mes_fonctions.php';

    // Fonction de connexion à la base de données
    connexion_DB('BASE_TEST');

    // Définit l'Id de la page d'accueil (1 dans cet exemple)
    // Pensez à le modifier si ce n'est pas le cas chez vous.
    $id_page_accueil = 1;

    // Récupère l'id de la page courante passée par l'URL
    // Si non défini, on considère que la page est la page d'accueil
    if (isset($_GET['id_page'])) {
        $_ENV['id_page'] = intval($_GET['id_page']);
    } else {
        $_ENV['id_page'] = $id_page_accueil;
    }

    // Extrait les informations correspondantes à la page en cours de la DB
    extraction_infos_DB();
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<!-- Insère les mots-clés extraits de la DB dans les meta -->
<META NAME="keywords" lang="fr" CONTENT="<?php echo $_ENV['mots_cles']; ?>">
<!-- Insère la description extraite de la DB dans les meta -->
<META NAME="Description" CONTENT="<?php echo $_ENV['description']; ?>">
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- Insère le titre extrait de la DB dans la balise correspondante -->
<title><?php echo $_ENV['titre']; ?></title>
<link rel="stylesheet" type="text/css" href="styles.css">
</head>
<body>
    <div id="menu_horizontal">
        <?php
            // Affiche le menu 'racine' => id de la page = id de la page d'accueil = 1
            echo affiche_menu($id_page_accueil);
        ?>
    </div>
    <div id="chemin_fer">
        <?php
            // Affiche le chemin de fer
            echo 'Vous êtes ici : '.affiche_chemin_fer($_ENV['id_page']);
        ?>
    </div>
</body>
</html>
```

index.php

```

        ?>
    </div>
    <div id="bloc_central">
        <div id="menu_vertical">
            <?php
cours.
                // Affiche le menu en cours => id de la page = id de la page en
                echo affiche_menu($_ENV['id_page']);
            ?>
        </div>
        <div id="contenu">
            <?php
                // Affiche le contenu de la page en cours
                echo $_ENV['contenu'];
            ?>
        </div>
    </div>
    <div id="pied_page">
        <?php include("pied_page.html"); ?>
    </div>
    <?php deconnexion_DB(); ?>
</body>
</html>

```

Et voici le CSS lié :

styles.css

```

/* propriétés générales de la page */
body {
    background-color:#DDE3EF;
}
/* propriétés du menu horizontal */
#menu_horizontal {
    text-align:center;
}
/* propriétés du chemin de fer */
#chemin_fer {
}
/* bloc central uniquement crée pour uniformiser les hauteurs des deux blocs contenus (menu_vertical
et contenu) */
#bloc_central {
}
/* propriétés du menu vertical */
#menu_vertical {
    float:left;
    width:20%;
    height:300px;
}
/* propriétés du contenu */
#contenu {
    float:left;
    width:80%;
    height:300px;
    font-size:28px;
    text-align:center;
}
/* propriétés du pied de page */
#pied_page {
    clear:both;
    text-align:center;
}

```

Et enfin le fichier pied_page.html

pied_page.html

```

<hr>
Ceci est le pied de page...

```

Le CSS sera amené à évoluer au fil du temps, le PHP restera en l'état, toutes les fonctions appelées seront ajoutées dans mes_fonctions.php.

Nous avons défini ici trois fonctions : `extraction_infos_DB()`, `affiche_menu()`, et `affiche_chemin_fer()`. En les construisant bien, rien de plus ne sera nécessaire.

III-C - Détail des fonctions de récupération et d'affichage des données

Nous allons détailler ici les différentes fonctions auxquelles nous faisons appel dans notre `index.php`.

Ces fonctions seront à ajouter dans le fichier "`mes_fonctions.php`"

Commençons par la première, `extraction_infos_DB()` :

```
extraction_infos_DB()
// Récupère les informations de la page concernée
function extraction_infos_DB() {
    $strSQL = 'SELECT * FROM `pages` WHERE `Id_page` = ' . $_ENV['id_page'];
    $resultat = requete_SQL($strSQL);
    $tabl_result = mysql_fetch_array($resultat);
    $_ENV['mots_cles'] = $tabl_result['Mots_cles'];
    $_ENV['description'] = $tabl_result['Description'];
    $_ENV['titre'] = $tabl_result['Titre'];
    $_ENV['contenu'] = $tabl_result['Contenu'];
    $_ENV['id_parent'] = $tabl_result['Id_parent'];
}
```

Nous nous contentons simplement ici de récupérer les différentes informations de la base de données pour la page en cours.

On colle toutes ces informations dans des variables d'environnement histoire d'être tranquilles, et hop !

Rien de bien sorcier quoi ;)

Mais pourquoi ce `intval` lors de la récupération de la valeur passée par l'URL ? Tout simplement pour s'assurer que la valeur récupérée est bien un nombre, et ne risque pas de détourner la requête de son but initial.

Dans cet exemple, ce n'est pas flagrant, mais documentez-vous sur l'injection SQL pour vous rendre compte des risques.

Pensez systématiquement à contrôler toutes les saisies de l'utilisateur (ou valeurs qu'il pourrait modifier, telles les variables passées par l'URL). Par défaut, ne JAMAIS faire confiance à l'utilisateur.

Passons maintenant à un peu plus difficile... L'affichage du chemin de fer. Commençons par le code, l'explication viendra après :

```
affiche_chemin_fer($idpage)
// Affiche le chemin de fer.
// Paramètres : id de la page en cours -> $idpage
// Renvoie : chemin complet -> $chemin_complet
function affiche_chemin_fer($idpage) {
    // on définit la variable pour éviter le warning
    $chemin_complet = "";
    // Si l'id de la page en cours est différent de 0
    // (0 = page parente de la page racine = inexistante)
    if ($idpage != 0) {
        // on récupère les informations de la page en cours dans la DB
        $strSQL = 'SELECT `Titre`, `Id_parent` FROM `pages` WHERE `Id_page` = ' . $idpage;
        $resultat = requete_SQL($strSQL);
        $tabl_result = mysql_fetch_array($resultat);

        $titrepage = $tabl_result['Titre'];
```

```

affiche_chemin_fer($idpage)
    $idparent = $tabl_result['Id_parent'];

    // création du lien vers la page en cours
    $chemin_page_en_cours = ' -> <a
href="index.php?id_page='.$idpage.'">'.$titrepage.'</a>';

    // Concaténation du lien de la page N-1 et
    // du lien de la page en cours
    $chemin_complet = affiche_chemin_fer($idparent).$chemin_page_en_cours;
}
// renvoie le chemin complet
return $chemin_complet;
}

```

N'ayez pas peur, ce n'est pas si compliqué que ça y paraît ...

Il s'agit simplement d'une fonction récursive (qui se rappelle elle-même). Elle récupère les informations de la page en cours, puis elle crée le lien correspondant et se rappelle elle-même. Elle prend en paramètre l'ID de la page parent, tant que cet ID n'est pas égal à 0 (id_parent = 0 -> page racine du site).

Reste la dernière fonction, celle chargée d'afficher le menu...

L'idée est la suivante :

- La page en cours a des filles, auquel cas, le menu sera composé de liens vers ces filles.
- La page en cours est une page feuille, auquel cas le menu sera composé de liens vers ses soeurs.

```

affiche_menu($idpage)
// Affiche les menus.
// Paramètres : id de la page -> $idpage
// (id de la page en cours pour le menu vertical, id de la page racine (1) pour le menu horizontal)
// Renvoie : le menu sous forme de liste -> $menu_retour
function affiche_menu($idpage) {
    // Sélectionne toutes les pages filles de la page en cours
    $strSQL = 'SELECT `Id_page`, `Titre` FROM `pages` WHERE `Id_parent` = '.$idpage;
    $resultat = requete_SQL($strSQL);
    // Si la page n'a pas de page fille, alors on modifie la requête pour obtenir ses pages
    soeurs.
    if (mysql_num_rows($resultat) == 0) {
        $strSQL = 'SELECT `Id_page`, `Titre` FROM `pages` WHERE `Id_parent` =
        '.$_ENV['id_parent'];
        $resultat = requete_SQL($strSQL);
    }
    $menu_retour = '<ul>';
    while ($tabl_result = mysql_fetch_array($resultat)) {
        $menu_retour .= '<li>';
        $menu_retour .= '<a href="index.php?id_page='.$tabl_result['Id_page'].'">';
        $menu_retour .= $tabl_result['Titre'];
        $menu_retour .= '</a>';
        $menu_retour .= '</li>';
    }
    $menu_retour .= '</ul>';
    return $menu_retour;
}

```

Rien de bien compliqué ici.

On sélectionne toutes les pages filles de la page passée en paramètre. Si il n'y en a pas, on sélectionne toutes les pages filles de la page parent.

Il ne reste plus ensuite qu'à générer notre liste, laissons les CSS gérer l'affichage :)

III-D - Gestion de la feuille de style

Le plus important d'abord... Afficher notre liste 'en ligne'. Rien de bien compliqué, rassurez-vous.

Rajoutons le code suivant à notre fichier "styles.css" :

affichage en ligne d'une liste

```
/* comportement de la liste dans le bloc menu_horizontal */
#menu_horizontal li {
    margin-right:40px;
    list-style-type: none;
    display: inline;
}
```

Ce simple petit bout de code suffit à préciser l'écartement entre nos différents éléments, l'affichage ou non d'une puce, ainsi que l'affichage en ligne.

Le #menu_horizontal li signifie que ce style s'appliquera à tout les éléments de type liste contenus dans le bloc ayant pour id menu_horizontal.

Rajoutons encore quelques styles pour rendre notre affichage plus agréable :

styles complémentaires

```
/* propriétés des liens (de base et déjà visités) contenus dans le bloc menu_horizontal */
#menu_horizontal a:link, #menu_horizontal a:visited {
    color:#339999;
    font-weight:bold;
    text-decoration:none;
}

/* propriétés des liens (au dessus et actifs) contenus dans le bloc menu_horizontal */
#menu_horizontal a:hover, #menu_horizontal a:active {
    text-decoration:underline overline;
}

/* propriétés des liens (de base et déjà visités) contenus dans le bloc chemin_fer */
#chemin_fer a:link, #chemin_fer a:visited {
    color:#FF9933;
    font-weight:bold;
    text-decoration:none;
}

/* propriétés des liens (au dessus et actifs) contenus dans le bloc chemin_fer */
#chemin_fer a:hover, #chemin_fer a:active {
    text-decoration:underline overline;
}

/* propriétés des liens (de base et déjà visités) contenus dans le bloc menu_vertical */
#menu_vertical a:link, #menu_vertical a:visited {
    color:#CC66CC;
    font-size:16px;
    text-decoration:none;
}

/* propriétés des liens (au dessus et actifs) contenus dans le bloc menu_gauche */
#menu_vertical a:hover, #menu_vertical a:active {
    text-decoration:underline;
}
```

Voici le résultat que vous devriez obtenir :

Résultat final

IV - Conclusion

Et voilà, vous disposez à présent d'un site entièrement dynamique. Vous pouvez rajouter autant de pages que vous le souhaitez, en supprimer à la volée, les menus seront toujours à jour, de même que les liens.

Finalement, avec un tout petit effort de conception à la base, il est facile d'être fainéant après. Plus besoin désormais de modifier 42 pages, si vous souhaitez déplacer le menu, une simple édition du fichier index.html suffira.

Si vous souhaitez aller plus avant dans la conception de ce site (classes d'abstractions, système de template), je vous conseille la lecture du tutoriel de [Guillaume Rossolini](#) : [Tutoriel de site dynamique - Classes d'abstraction](#).

Un très gros merci à l'équipe web de developpez.com, ainsi qu'à mon bêta testeur qui a bien voulu se dévouer pour éponger les plâtres : [SpaceFrog](#).

[Télécharger les codes source de l'article](#)