Site internet pour ligues
et clubs de sports d'équipe

Manuel du développeur

PhpMySport vl.2
Sorti le 08/03/2007

Table des matières

Introduction

A qui s'adresse ce manuel ?

Ce manuel s'adresse à tous les **bricoleurs**, **webmasters** et autres **développeurs** qui souhaitent **comprendre le fonctionnement de phpMySport** afin de le **personnaliser**, lui apporter des modifications, ou lui ajouter des fonctionnalités supplémentaires. Il vous permettra de découvrir les **méthodes de développement** choisies, d'acquérir les notions utilisées et de connaître les spécificités du logiciel. Vous aurez ainsi **toutes les clés** en main pour comprendre la structure du logiciel et l'adapter à vos besoins. Mais avant toute chose, savez-vous à quoi sert phpMySport ?

PhpMySport

PhpMySport est un logiciel web destiné aux **clubs de sport et aux ligues sportives** qui désirent créer facilement leur **site internet**. A la fois complet et simple d'utilisation, il s'adapte à la plupart des **sports d'équipe** : football, handball, volley, basket, hockey, waterpolo, etc.

Il permet de gérer les **membres**, les **matchs**, les **compositions d'équipe**, les **championnats**, les **saisons** et bien plus encore. Le logiciel propose en outre une gestion des **actualités,** une section **pages web** à modifier librement ainsi qu'un **forum de discussion.**

Réalisé en PHP et couplé à une base de données **MySQL**, phpMySport fait partie des « Content Management Systems » ou **CMS** ; c'est un système de gestion de contenu permettant de concevoir et de **gérer simplement un site internet.** Il se différencie par ses aspects sportifs mais conserve ce qui fait les plus des CMS : **personnalisation du design**, **éditeur de texte WYSIWYG**, **gestion simplifiée et intuitive**, gestion **multi-utilisateurs**, etc.

En d'autres termes, phpMySport offre un site web complet qui s'adapte à votre sport ainsi qu'aux couleurs de votre club ou ligue !

Voici les principales fonctionnalités intégrées à phpMySport, selon les versions :

La version 1.0

La version 1.0 de phpMySport inclut les fonctionnalités suivantes :

- Gestion des actualités
- Gestion de pages web d'information
- Gestion des membres, des joueurs, des entraîneurs et des dirigeants
- Gestion des clubs et des stades
- Gestion des équipes et de leur composition
- Gestion des matchs, des saisons et des compétitions
- Forums de discussion
- Espace membre
- Espace administration

La version 1.1

Parmi les nouvelles fonctionnalités :

- Gestion des plugins
- Ajout d'un plugin : Gestion de liens web

La version 1.2

Parmi les nouvelles fonctionnalités :

- Classement par points et statistiques des équipes
- Statistiques des joueurs
- Amélioration de la gestion des compétitions : système par tour
- Gestionnaire de fichiers et d'images
- Introduction de l'italien et du portugais

Avant de commencer

Avant de commencer, il vous faudra tout d'abord **télécharger la dernière version** du logiciel et l'**installer** en suivant les indications du manuel utilisateur. S'il s'agit de votre première utilisation, alors prenez un moment pour parcourir le logiciel et découvrir ses différentes fonctionnalités. Il vous sera d'autant plus facile ensuite de comprendre la manière dont il est conçu.

Vous maîtrisez le logiciel mais vous avez envie d'apporter des modifications ? Alors ce manuel est fait pour vous. Il vous expliquera comment créer un nouveau design, comment développer un plugin ou encore comment est construite la base de données.

Rassurez-vous, il n'est pas nécessaire d'être expert pour comprendre le fonctionnement de phpMySport. Néanmoins, il est préférable d'avoir déjà quelques **notions en PHP/MySQL, HTML et JavaScript**, langages avec lesquels phpMySport a été développé. Les mots « variable » et « base de données » ne vous sont pas étrangers ? Alors lancez-vous dans la lecture de ce guide et découvrez tous les secrets de phpMySport…

Bonne lecture !

Objectif, contraintes et méthodes

Objectif

L'objectif de PhpMySport est de faciliter la création et la gestion d'un site internet pour une ligue ou un club de sport d'équipe. Cette application web se veut à la fois simple à utiliser et suffisamment riche pour gérer tout le contenu d'un site. Elle doit être assez intuitive pour qu'un néophyte puisse s'en servir mais elle doit aussi laisser suffisamment de liberté à un webmestre confirmé qui souhaiterait la personnaliser.

Contraintes

PhpMySport est destiné à des utilisateurs dont l'origine, les goûts et les besoins sont divers. Le logiciel doit donc tenir compte de ces différences d'utilisation pour répondre aux exigences du plus grand nombre. Si phpMySport se doit de satisfaire les utilisateurs, il lui faut aussi s'adapter aux environnements existants et aux bases de données couramment utilisées. Il doit pouvoir par ailleurs évoluer facilement.

Face à ces besoins, plusieurs contraintes ont été définies. Pour chacune d'entre elles, une solution a été pensée et choisie de manière à ce que le logiciel puisse s'adapter. Nous ne détaillerons pas ici le choix des solutions utilisées ; nous nous contenterons de les exposer.

Voici la liste des principales contraintes imposées et les solutions adoptées :

Contrainte	**Solution choisie**
Le logiciel doit pouvoir être édité en plusieurs langues	Utiliser des fichiers de langues (voir la section Les fichiers de langue)
Le design du logiciel doit pouvoir être facilement modifiable	Utiliser un système de template et les styles CSS (Voir la section Design et templates)
Le logiciel doit être facile à utiliser et à gérer	Intégrer l'espace d'administration (back office) à l'interface générale (front office). Utiliser des icônes explicites et un éditeur de texte de type What You See Is What You Get (WYSIWIG).
Les pages doivent être optimisées pour le référencement dans les moteurs de recherche	Utiliser du code XHTML et CSS valide. Utiliser et la technique de l'URL rewriting (Voir la section URL rewriting : des URLs simples à comprendre)
Le logiciel doit pouvoir être utilisé avec différents types de base de données	Utiliser des fonctions génériques et séparer les requêtes SQL dans des fichiers spécifiques (Voir la section Des fonctions génériques pour interagir avec la base de données)
Le logiciel doit pouvoir être déployé sur la plupart des serveurs web	Utiliser le langage PHP, dans sa version 4.0
Le logiciel doit pouvoir facilement s'installer et être mis à jour	Intégrer un assistant d'installation et de mise à jour

Méthodes de conception

Un logiciel fiable et sécurisé ne peut être développé sans une conception rigoureuse et réfléchie. PhpMySport ne déroge pas à la règle et a été créé en suivant des méthodes reconnues et fréquemment utilisées en génie logiciel : le modèle MVC pour le développement et la méthode Merise pour la modélisation de la base de données.

Le **modèle « Model View Controller »** (MVC) permet de séparer les couches de conception :

- le modèle gère la manière d'accéder aux données
- la vue se charge de présenter des données
- le contrôle est l'étape intermédiaire de sélection et de fusion des données

Cette séparation rend le code plus lisible et facilite le travail en équipe. Par exemple, les webdesigners réalisent l'interface graphique du site pendant que les développeurs s'emploient à écrire le code PHP traitant les informations à afficher.

Bien que souvent associé à la Programmation Orientée Objet (POO), le modèle MVC de phpMySport n'utilise pas de classe, ni d'objet. La POO est en effet une notion abstraite pour la plupart des internautes et ne peut être correctement appréciée que par des développeurs expérimentés. PhpMySport se veut ouvert et doit pouvoir être modifié facilement, même par une personne débutante. Un développement en POO a par conséquent été écarté afin de respecter cette contrainte.

La **méthode Merise** permet quant à elle de modéliser les données traitées par le logiciel et de définir les relations qui existent entre elles. Par exemple, on pourra dire que selon les saisons sportives, un joueur peut appartenir à un ou plusieurs clubs. Cette méthode comporte différentes étapes qui aboutissent à la création de la base de données.

Un schéma représentant la structure de la base de données de phpMySport v1.1 est disponible en annexe (remarque : faute de place, les tables du forum de discussion ne sont pas représentées).

Langages utilisés

PhpMySport n'utilise pas moins de **5 langages** qui ont tous leur particularité. Voici un rapide aperçu des langages présents dans phpMySport.

MySQL

MySQL est un système de gestion de base de données. Il permet d'organiser et de conserver des informations. Les données sont stockées dans des tables. L'insertion, la mise à jour, la suppression et la sélection des données est possible grâce à des requêtes SQL. PhpMySport utilise une base de données MySQL pour le stockage des informations du site.

PHP

PHP est un langage de programmation actuellement très utilisé pour la réalisation d'applications web comme phpMySport. Il permet d'effectuer des calculs, de récupérer les valeurs d'un formulaire ou encore d'interagir avec une base de données. PHP a également la particularité de pouvoir être associé à du code HTML. Néanmoins dans phpMySport, le code PHP et le

code HTML sont séparés. Le logiciel utile en effet un système de templates qui sera détaillé ultérieurement.

HTML

Le langage HTML ou XHTML est utilisé pour mettre en page les informations à diffuser. C'est ce langage que va comprendre votre navigateur web : il va le décoder et l'interpréter pour afficher les pages telles que vous les voyez à l'écran.

CSS

Le langage CSS est utilisé pour mettre en forme les données à afficher. Il est complémentaire à HTML. C'est également un langage très utilisé pour les applications web.

JavaScript

JavaScript est un langage qui va animer le code HTML. Il est par exemple utilisé pour vérifier les données d'un formulaire ou pour réaliser des menus déroulants interactifs. Il est par ailleurs l'un des composants de la technologie AJAX. PhpMySport l'utilise principalement pour rendre la saisie des formulaires plus agréable.

Organisation du logiciel

Architecture du logiciel

PhpMySport est construit selon une architecture qui lui est propre, mais qui s'appuie sur des méthodes largement répandues parmi les concepteurs de logiciels.

L'application est construite autour d'un noyau, ou **squelette**, sur lequel viennent se greffer différents modules, ou **organes**. Alors que les organes sont doués d'une fonctionnalité bien précise, par exemple « Afficher et gérer des actualités », le squelette n'a pas de fonctions propres mais permet aux organes de communiquer entres eux. Il rassemble par ailleurs des fonctions vitales utilisées par tous les organes. Sans organes, le squelette est inutile et inversement, les organes ne peuvent fonctionner sans squelette.

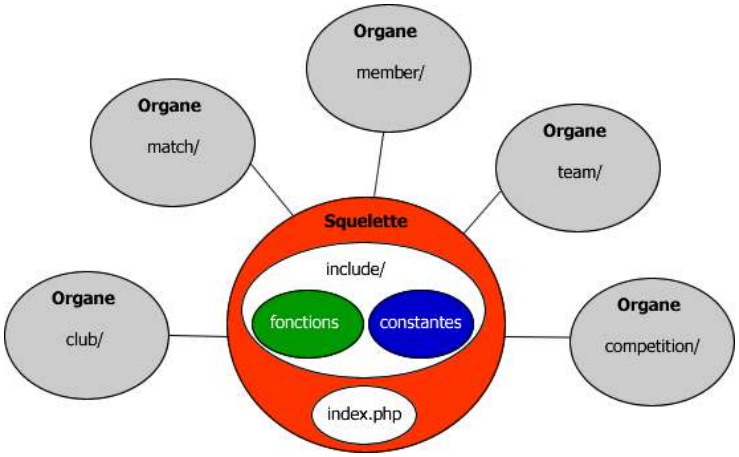
Le squelette

Le squelette est l'élément central de phpMySport. Il communique avec tous les organes et rassemblent les fonctions communes. Le squelette est formé par :

- le **fichier « index.php »** qui est à la fois le cerveau et le cœur du logiciel. C'est lui qui détecte la page demandée par l'utilisateur, qui récupère les informations et qui les affiche. Il contrôle par ailleurs l'accès aux pages réservés aux webmasters.
- le **dossier « include/ »** qui fournit les fonctions vitales du logiciel. Ce dossier contient toutes les informations communes aux organes comme les fonctions de connexion à la base de données, les fonctions de traitement des formulaires, le texte utilisé dans l'interface générale (fichier de langue) ou encore les constantes communes (fichier de configuration). La liste des éléments communs est détaillée dans la section Constantes et fonctions communes à utiliser.

Les organes

Un organe correspond à une fonctionnalité précise. Par exemple, l'organe « club » permet de gérer des clubs de sport. Pour cela, l'organe « club » utilise les fonctions du squelette pour communiquer avec la base de données et récupérer les informations sur les clubs enregistrés.



Chaque organe est constitué par plusieurs fichiers regroupés au sein d'un même dossier. La fonctionnalité d'un organe est donnée par le code contenu dans ses fichiers. L'organe « club » aura par exemple plusieurs fichiers lui permettant d'ajouter, modifier, supprimer et lister des clubs, soit tout ce qu'il faut pour assurer sa fonction.

Arborescence des fichiers

Dans sa version 1.2, phpMySport comporte 17 dossiers. 11 d'entres eux correspondent à des organes :

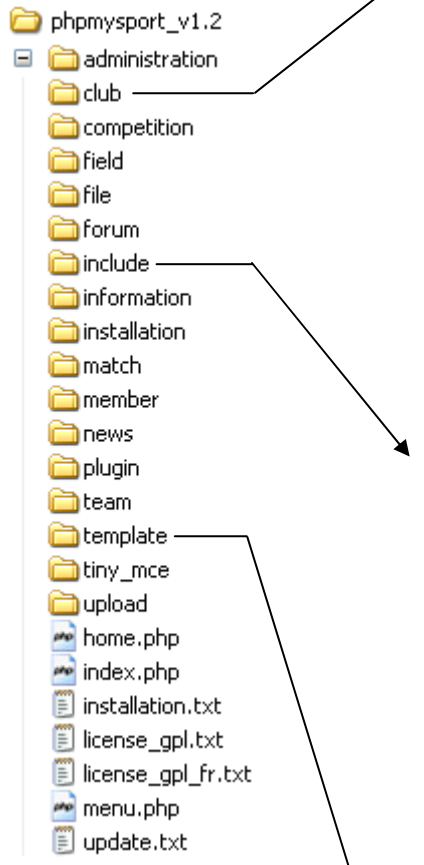
administration :	gestion et configuration du site
club :	gestion des clubs de sport
competition :	gestion des compétitions, des championnats et des saisons
field :	gestion des lieux de matchs (stades, salles, etc.)
file :	gestion de fichiers et d'images avec upload sur le serveur
forum :	gestion des forums de discussion
information :	gestion de pages web d'information
match :	gestion des matchs et des statistique de match
member :	gestion des membres, des joueurs, des dirigeants
news :	gestion d'articles d'actualités
team :	gestion des équipes et des compositions d'équipe

Les 6 dossiers restants sont :

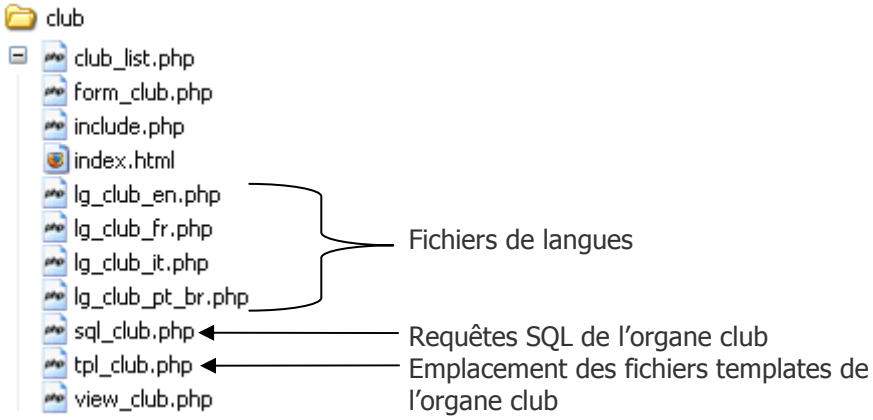
include :	contient les fichiers de configuration et de fonctions (squelette)
installation :	assistant d'installation et de mise à jour ; par mesure de sécurité, ce dossier doit être supprimé à la fin de l'installation
plugin :	dossier où doivent être copié les plugins
template :	dossier où doivent être placé les templates
tiny_mce :	éditeur de texte HTML utilisé dans les formulaires
upload :	dossier où sont uploadé les fichiers et images (via le gestionnaire de fichiers)

Voici une représentation globale de l'arborescence des fichiers avec en détails les dossiers « include », « template » et l'organe « club » :

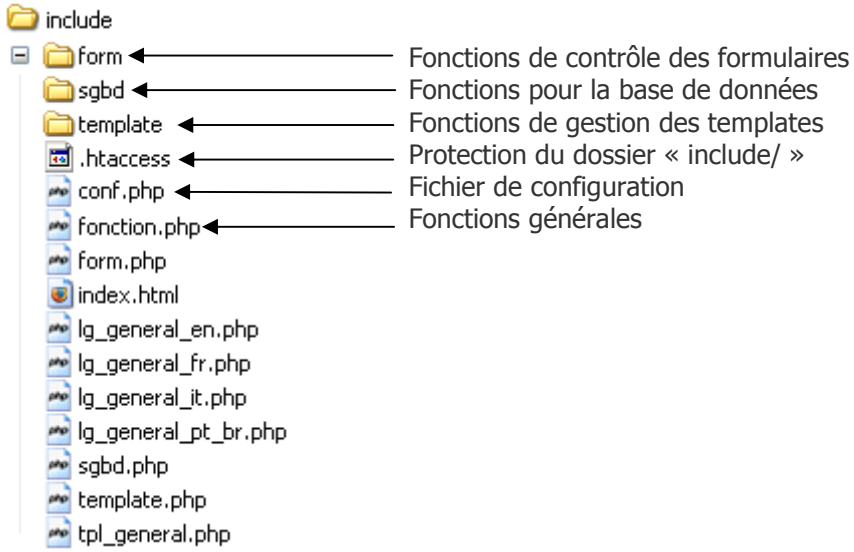
Arborescence générale



Organe « club »

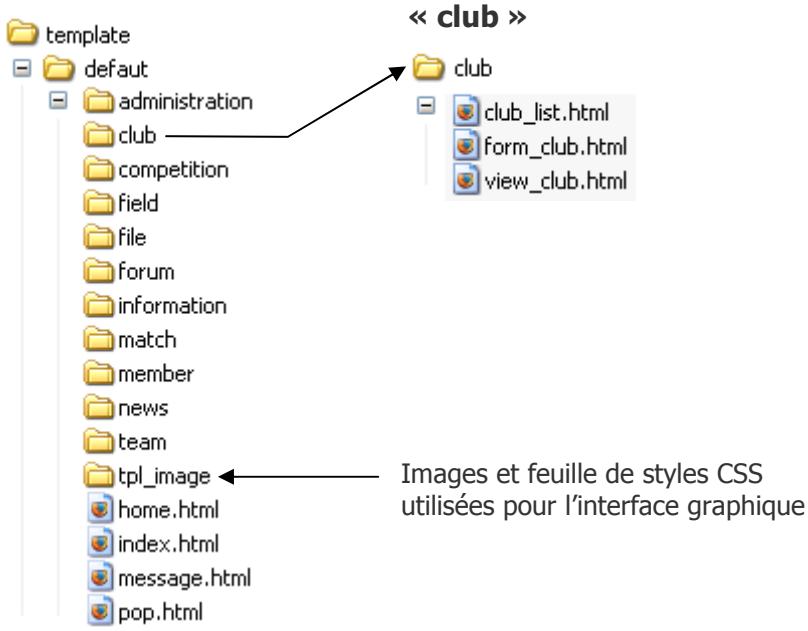


Squelette



Template

Template de l'organe « club »



Principales techniques utilisées

Comme nous l'avons vu dans la section Objectif, contraintes et méthodes, les contraintes imposées pour la conception de phpMySport ont nécessité la mise en place de solutions techniques adaptées. Voici en détails trois d'entres elles : les fichiers de langue, les fonctions génériques pour la base de données et la réécriture des URLs complexes.

Les fichiers de langue

Comment faire en sorte que phpMySport puisse être édité en plusieurs langues ? Autrement dit, comment faire pour que les textes de l'interface soit faciles à traduire ?

La solution choisie est l'utilisation de fichiers de langue. Ce sont des fichiers PHP qui contiennent l'ensemble des mots et des phrases utilisés dans l'interface générale du logiciel. Le nom de ces fichiers respecte la convention suivante :

« **lg_xxxx_yy**.php » avec :

- « **lg** » rappel qu'il s'agit d'un fichier de langue,
- « **xxxx** » est le nom du dossier auquel le fichier appartient
- « **yy** » est l'abréviation en 2 lettres de la langue de traduction

Par exemple, « lg_match_fr.php » est le fichier de langue de l'organe/dossier match traduit en français (fr).

Chaque fichier contient la variable $lang qui est en fait un tableau associatif avec pour valeurs les textes utilisés. Voici un exemple :

```
$lang['match']['match']='Match';
$lang['match']['match_list']='Listes des matchs';
$lang['match']['add_match']='Ajouter un match';
```

Ici, trois variables de langue sont définies pour l'organe « match ». Dans cet exemple, leur valeur correspond à un texte en français.

Le fichier « lg_match_en.php » contient quant à lui les même variables mais avec leurs valeurs en anglais (« en » pour english) :

```
$lang['match']['match']='Match';
$lang['match']['match_list']='Matches list';
$lang['match']['add_match']='Add a match';
```

En fonction de la langue choisie et définie lors de l'installation, les fichiers de langue correspondants seront appelés par le logiciel.

Des fonctions génériques pour interagir avec la base de données

PhpMySport doit pouvoir être installé sur différents types de plateforme et être capable de communiquer avec plusieurs types de base de données. L'application doit aussi bien s'adapter à MySQL, qu'à PostGre ou encore Oracle. Pour répondre à cette contrainte, des fonctions génériques de gestion de la base de données ont été implémentées. Elles sont utilisées dans l'ensemble du logiciel. Par exemple, la fonction sql_connect() permet de se connecter à une base de données. Peut importe qu'il s'agisse d'une base MySQL ou d'une base Oracle, selon la configuration du logiciel, cette fonction se connectera à la base de données. La liste des fonctions génériques est présentée dans la section Fonctions de base de données.

Remarque : dans la version 1.2, seules les fonctions pour MySQL sont disponibles. Les prochaines versions du logiciel devraient être compatibles avec d'avantages de base de données.

Par ailleurs, les noms des tables de la base de données sont conservés dans des constantes PHP. Si besoin, il est donc possible de les renommer. Il suffira alors de modifier le nom de ces constantes pour que toutes les requêtes utilisant ces tables soient modifiées. Les constantes sont définies dans le fichier « /include/sgbd/sgbd_tables.php ». Voici un exemple qui montre comment est définie la table des matchs :

```
define("T_match","`match`");
```

Enfin, le code SQL est séparé du code PHP. A la manière des fichiers de langue, des fichiers SQL contiennent l'ensemble des requêtes utilisées pour un organe donné. Le fichier « sql_match.php » contient par exemple l'ensemble des requêtes utilisées pour insérer, modifier, supprimer et sélectionner les matchs. Chaque fichier contient la variable $sql. Voici un exemple :

```
$sql['match']['select_match']="SELECT * FROM ".T_match." ";
```

Il s'agit ici de la requête permettant de sélectionner tous les matchs dans la table « match ». La constante « T_match » correspond au nom de la table des matchs.

URL rewriting : des URLs simples à comprendre

Comme tous les logiciels web, la navigation entre les pages de phpMySport se fait par l'intermédiaire de liens internet. Dans phpMySport, ces liens comportent des informations qui indiquent quelle page du site le logiciel doit s'afficher. Les données passées pour naviguer entre les pages sont bâties sur la convention suivante :

r	rubrique du site	*il s'agit de l'organe à appeler*
v1	première variable	*il s'agit de la page à appeler*
v2	deuxième variable	*autre variable utile à la page appelée*

Exemple :

```
index.php?r=match&v1=match_list
```

Dans cet exemple, il est demandé d'afficher la liste des matchs (v1=match_list) de l'organe match (r=match).

Ces URLs posent plusieurs problèmes : elles sont difficiles à retenir par les internautes et sont mal reconnues par les moteurs de recherche. Aussi, phpMySport a adopté la technique de réécriture des URLs (URL rewriting) qui permet de simplifier les adresses web complexes. Cette technique n'est possible que si l'application est installée sur un serveur Apache et que le mod « Rewrite » est activé. A l'installation de phpMySport, il est demandé à l'utilisateur s'il désire activer cette option. Dans la positive, le logiciel créé un fichier .htaccess qui permet au serveur Apache de comprendre ces adresses simplifiées.

Lors du développement, tous les liens doivent respecter la convention et être construits avec une fonction qui permet de les convertir si le logiciel est configuré pour la réécriture des URLs. Cette fonction est :

```
convert_url($url)
```

Voici un exemple d'utilisation :

```
convert_url("index.php?r=match&v1=match_list");
```

retournera :

```
/match/match_list.html
```

Design et templates

Qu'est-ce qu'un template ?

Dans phpMySport, le développement PHP est séparé du design HTML. Ainsi, le logiciel peut être personnalisé selon les couleurs et la mise en page souhaitées par l'utilisateur. Ceci est possible grâce un système de « template ». Il y a d'un côté les fichiers contenant le code PHP et de l'autre les fichiers « template » qui sont en réalité des pages HTML contenant un code template. Avant l'affichage d'une page web, le logiciel va lire ces fichiers HTML et remplacer le code template par la valeur des variables PHP correspondantes.

Les différents designs sont situés dans le dossier « template ». Chaque design est composé d'un dossier comportant l'ensemble des fichiers HTML, classé par organe. Un seul design est fourni avec phpMySport, il s'agit du template « defaut ».

Voyons en détails le code template.

Le code template

Trois types d'informations peuvent être affichés dans une page : des données statiques comme du texte, des données cycliques comme des listes et des données optionnelles.

Trois codes template ont donc été mis en place. Ils sont chacun associés à un type de variables PHP : du texte (string) pour les données statiques, des tableaux (array) pour les listes et un booléen pour les données optionnelles.

L'ensemble des données à afficher sont stockées dans une seule et unique variable : $page. Il s'agit d'un tableau à deux dimensions qui contient l'ensemble des textes, des tableaux et des booléens nécessaires à une page du site.

Une fonction PHP utilisée dans le fichier « index.php » va fusionner cette variable $page et le fichier template défini pour produire le code HTML final envoyé au navigateur.

Code simple

Le code simple est utilisé pour afficher une variable PHP de type « texte ».

Le nom des variables est placé entres accolades « { » et « } »

Code template (fichier HTML)	**Code PHP (fichier PHP)**	**Affichage à l'écran**
`{mon_texte}`	`$page['mon_texte'] = "Mon texte"`	Mon texte

Code des listes

Les listes sont utilisées pour afficher une variable PHP de type « array ». Le mot clé « LOOP » est utilisé pour signifier que la variable correspond à une boucle. Tout le code HTML compris entre « LOOP » et « END LOOP » sera répété autant de fois que le tableau a de lignes.

Code template	Code PHP	Affichage à l'écran
<!-- LOOP ma_liste --> {nom} {prenom} <!-- END LOOP ma_liste -->	$page['ma_liste']=array(array('nom'=>"Durant",'prenom'=>"Martin"), array('nom'=>"Dupond",'prenom'=>"Marie"));	Durant Martin Dupond Marie

Code optionnel

Le code optionnel est utilisé pour les variables de type booléen ou précisément « vide/non vide ». Si la variable est vide, alors le code HTML compris entre les deux balises « OPTION » sera invisible. Sinon il s'affichera.

Code template	Code PHP	Affichage à l'écran
<!-- OPTION var1 --> {texte1} <!-- FIN OPTION var1 --> <!-- OPTION var2 --> {texte2} <!-- END OPTION var2 -->	$page['texte1']="Texte 1"; $page['texte2']="Texte 2"; $page['var1']=""; $page['var2']="1";	Texte 2

Créer votre propre design

Pour personnaliser le design de phpMySport et le mettre aux couleurs de votre club, il vous faudra créer votre propre template. Cela est relativement rapide, mais il est préférable d'avoir déjà quelques notions en langage CSS et HTML. Voici les différentes étapes à suivre :

- Ouvrez le répertoire où sont stockés les fichiers de phpMySport (à l'aide votre client FTP si besoin).
- Ouvrez le répertoire « template » situé à la racine de votre site. Un sous-répertoire intitulé « défaut » est présent. Ce dernier contient le design par défaut de phpMySport. **Pour éviter tout risque de dysfonctionnement du script, il est vivement déconseiller d'effacer ou de modifier ce répertoire ainsi que les fichiers qu'il contient !**
- Copier/coller le dossier « defaut ». Renommer le avec un mot (texte sans espace) de votre choix (ex : « mondesign ») et placer le dans le répertoire « template ». Celui-ci contient désormais deux sous-répertoires : « defaut » et « mondesign »
- Ouvrez le répertoire que vous venez de copier/coller. Vous y trouverez plusieurs dossiers ainsi que de nombreux fichiers HTML. Rassurez-vous, il n'est pas nécessaire de tous les modifier pour changer le design de votre site ! Seuls 2 éléments sont importants :

- Le dossier « /tpl_image/ » : il contient tous les images qui composent l'interface graphique du site.
- Le fichier « /tpl_image/styles.css » : il contient toutes les informations de mise en forme et de mise en page du texte, des menus, des rubriques, des tableaux, etc

- Pour modifier le design, il vous suffit donc d'intégrer vos propres images et de modifier le contenu du fichier CSS.
- Voici un exemple si vous souhaitez insérer votre logo à la place de celui de phpMySport :
  - Placez l'image correspondante à votre logo (format .jpg, .gif, .png ou autre) dans le dossier « /tpl_image/ » du répertoire que vous avez copier/coller (« mondesign »)
  - Ouvrez le fichier « /tpl_image/styles.css »
  - Remplacer la ligne suivante :

```
div#header { height:90px; width:100%; margin:0 auto;
padding:0; }
```

  - Par :

```
div#header { height:90px; width:100%; margin:0 auto;
padding:0; background:url(mon_logo.jpg) no-repeat; }
#header img { display:none };
```

  - Le fichier « mon_logo.jpg » correspond à l'image que vous avez placé dans « /tpl_image/ ». Si nécessaire, modifier la hauteur « 90px » par celle de votre image.
- Vous pouvez bien entendu personnaliser les autres éléments de l'interface : menu, entête, pied de page, couleur de fond, taille et couleur du texte, titres, etc.
- Pour changer le design par défaut et appliquer les modifications que vous venez d'apporter :
  - Allez dans l'espace d'administration, rubrique « configuration du site »
  - Dans la liste déroulante correspondantes, choisissez le design que vous venez de créer et valider votre choix
  - Le nouveau design de votre site apparaît alors.

N'hésitez pas à faire connaître vos qualités artistiques en diffusant vos réalisations graphiques auprès des autres utilisateurs de phpMySport. Pour cela, envoyez-nous vos images et vos feuilles de styles CSS personnalisées !

Personnaliser phpMySport grâce aux plugins

Qu'est-ce qu'un plugin ?

Un plugin est un programme qui interagit avec un logiciel, en l'occurrence phpMySport, pour lui apporter de nouvelles fonctionnalités. En créant votre propre plugin ou en installant un plugin existant, il est donc possible d'augmenter considérablement les capacités de phpMySport. Par exemple, vous allez pouvoir intégrer des galeries photos, des flux RSS ou pourquoi pas reprendre la gestion des actualités inclus par défaut dans phpMySport pour y apporter des améliorations.

Vous souhaitez créer votre propre plugin ? Alors lisez attentivement les explications qui vous sont données ci-après. Après avoir étudié l'organisation des fichiers d'un plugins, nous verrons étape par étape comment réaliser un plugin, avant de finir avec un exemple concret.

Organisation des fichiers d'un plugin

Le répertoire « plugin » de phpMySport

PhpMySport possède un dossier « plugin/ » dans lequel les plugins téléchargés ou développés doivent être placés.

Ci-contre est représentée l'arborescence du dossier plugin contenant 2 plugins intitulés « link » et « my_plugin ».

Organisation d'un plugin

De son côté, un plugin comporte 3 sortes de fichiers.

Les premiers sont indispensables et constituent le cœur du plugin. Ils interviennent dans la compréhension, le fonctionnement et l'intégration du plugin dans phpMySport. Les seconds sont optionnels et dépendent uniquement des besoins du plugin. Enfin, les autres fichiers seront ceux qui apportent la nouvelle fonctionnalité proposé par le plugin.

Les fichiers indispensables

- conf.php : fichier de configuration du plugin
- include.php : fichier qui relie phpMySport et le plugin
- install.php : assistant d'installation du plugin
- lg_xxxx_yy.php : fichier de langue du plugin xxxx
- tpl_xxxx.php : fichier répertoriant les pages templates utilisées
- read_me.txt : description du plugin
- index.html : fichier vide permettant de sécuriser le plugin

Nous allons voir en détails les fichiers « conf.php », « install.php » et « include.php »

Le fichier « conf.php »

Comme son nom l'indique, le fichier « conf.php » contient des informations sur la configuration du plugin. Il permet également de faire le lien entre phpMySport et le

plugin. Les variables qu'il contient sont en effet utilisées pour le détecter et l'intégrer au logiciel.

Attention donc à **ne pas modifier le nom des variables utilisées**, sinon phpMySport ne pourra pas détecter pas votre plugin !

Voici les informations minimales que le fichier « conf.php » doit comporter :

Variable	Exemple de valeur	Description
`$plugin_name`	Mon plugin	Nom du plugin
`$plugin_idurl`	monplugin	Identifiant utilisé pour l'url. Il s'agit d'un mot sans espace et sans caractères spéciaux
`$plugin_root`	ROOT."/plugin/my_plugin"	Chemin indiquant où se situe le répertoire du plugin. Le dernier caractère ne doit pas être un slash « / »
`$plugin_install`	0 ou 1	Indique si le plugin a été installé (1) ou non (0)
`$plugin_active`	0 ou 1	Indique si le plugin est activité (1) ou non (0)
`$plugin_lang`	array("fr","en")	Tableau contenant la liste des langues dans lequel le plugin a été traduit. La langue est indiqué par une abréviation de 2 lettres.
`$plugin_page_admin`	array("install","form")	Tableau contenant la liste des pages protégées et réservées uniquement à l'administrateur du site. Il s'agit souvent du fichier d'installation et des formulaires d'ajout et de modification des données.

Le fichier « install.php »

Le fichier « install.php » est un fichier d'installation qui est nécessaire pour que les futurs utilisateurs puissent installer facilement le plugin. Il sert notamment à :

- exécuter les requêtes SQL de modification de la base de données, si besoin.
- de modifier le fichier de configuration du plugin « conf.php » pour l'activer. Lors de cette étape, les variables $plugin_install et $plugin_active sont mises à 1.

Il est accompagné par un fichier template ce qui permet d'obtenir un assistant visuel lors de l'installation du plugin par l'utilisateur.

Remarque : les prochaines versions nécessiteront sans doute un fichier de désinstallation.

Le fichier « include.php »

Le fichier « include.php » joue deux rôles :

- il fait appel aux fichiers communs du plugin à savoir : le fichier de langue, les requêtes SQL, la liste des pages templates utilisées et éventuellement les fonctions ou classes utilisées.
- il permet d'inclure le fichier .php correspondant à la page appelée. La page appelée est connue par la variable $_GET['v1'].

Les fichiers optionnels

- sql_xxxx.php : ensemble des requêtes SQL utilisées par le plugin
- bdd.txt : requête(s) SQL nécessaire à l'installation du plugin

Les autres fichiers apportant les nouvelles fonctionnalités

Il peut y en avoir un ou plusieurs, le choix des noms reste libre mais doit refléter l'action qu'ils réalisent. Par exemple, un fichier nommé « item_list.php » correspondra à une liste d'éléments « item ».

Chacun de ces fichiers est couplé à un fichier template portant le même nom mais utilisant l'extension .html. Il est conseillé de placer les fichiers template et les images utilisées par le plugin dans un dossier séparé « tpl ». La liste des fichiers templates doit être répertoriées dans le fichiers « tpl_xxxx.php » (avec xxxx correspondant au nom du plugin).

Comment créer un nouveau plugin ?

Voici les différentes étapes à suivre pour réaliser un nouveau plugin :

Etape 1

Dans le dossier « plugin/ » de phpMySport, créez un nouveau répertoire portant le nom de votre plugin. Nous l'appellerons le dossier racine de votre plugin. Dans ce nouveau dossier, créez un autre répertoire nommé « tpl » où seront placés les pages templates de votre plugin.

Etape 2

Créez tous les fichiers indispensables au bon fonctionnement du plugin tel qu'ils ont été décrits précédemment (voir la section Organisation des fichiers). Placez-les à la racine de votre plugin.

Etape 3

Si des informations doivent êtres stockés ou conservées, vous aurez peut-être besoin d'utiliser la base de données de phpMySport et de lui apporter des modifications. Pour cela, réfléchissez à l'organisation de vos données et créer un fichier .txt contenant la/les requête(s) SQL de création et/ou de modification de tables. Ce fichier sera nécessaire pour l'installation du plugin. A noter que vous pouvez également utiliser un ou plusieurs fichiers XML à la place de la base de données.

Etape 4

Créez un a un les fichiers apportant les nouvelles fonctionnalités du plugin. Dans ces fichiers PHP, il est vivement conseillé d'utiliser les variables, constantes et fonctions de

phpMySport. Si besoin, vous pouvez également développer vos propres fonctions. Ces fichiers doivent être convenablement commentés et respecter la méthode de développement de phpMySport. Pensez à utiliser la fonction convert_url(), la variable $page et à indiquer le fichier template avec lequel ils sont reliés. N'oubliez pas de créer les fichiers templates correspondants dans le dossier « tpl/ », de les répertorier et de les déclarer dans le code de votre page.

Etape 5

Vous êtes en train de développer votre plugin ? Vous souhaitez sûrement vérifier le résultat de votre code… Pour cela, il vous suffit :

- d'exécuter votre requête SQL de modification de la base de données (si nécessaire)
- d'activer votre plugin en mettant les variables « $plugin_install » et « $plugin_active » à 1.

Ouvrez votre navigateur web et saisissez l'adresse de votre site phpMySport. Le nom de votre plugin apparaît alors dans le menu principal du logiciel.

Etape 6

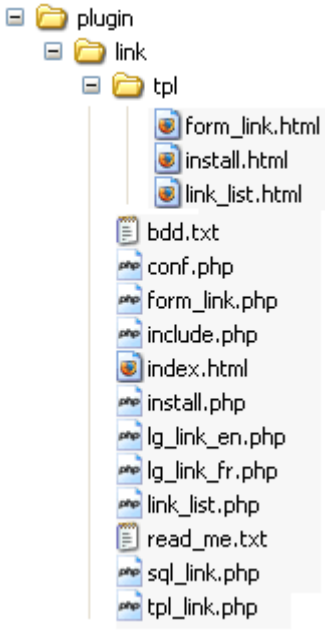
Ca y est, votre plugin est enfin fonctionnel ? Alors il ne vous reste plus qu'à le faire connaître et à le diffuser pour que d'autres utilisateurs de phpMySport puissent l'installer à leur tour. Pour cela, rendez-vous sur le site officiel de phpMySport et allez dans la rubrique « extension » où vous trouverez les informations pratiques pour proposer votre plugin.

Exemple : plugin de gestion de liens web

Vous avez compris l'organisation et les étapes de création d'un plugin ? Alors rien ne vaut un exemple qui vous permettra de mettre en pratique ce que vous avez appris. Nous allons voir comment créer une gestion simplifiée de liens web : le plugin « link ».

Organisation des fichiers

Regardons tout d'abord l'organisation des fichiers de notre plugin :



Nous avons créé et placé un dossier « link » dans le répertoire « plugin » de phpMySport. Ce dossier contient un sous-dossier « tpl » et une série de fichiers. Nous y retrouvons les fichiers indispensables, les fichiers optionnels et les fichiers qui apportent la nouvelle fonctionnalité :

- « form_link.php » : il permet d'ajouter ou de modifier un lien web. Il est couplé au fichier template « form_link.html » qui est en fait un formulaire HTML.
- « list_link.php » : il permet de récupérer la liste des liens web enregistrés. Il est couplé au fichier template « link_list.html ».

La base de données

Dans notre gestion de liens web, nous avons besoin de stocker les informations sur chaque lien web : nom, url et description. Pour cela, nous allons utiliser la base de données et créer une nouvelle table. Voici la requête SQL correspondante :

```
CREATE TABLE `link` (
  `link_id` int(11) NOT NULL auto_increment,     # Identifiant unique du lien
  `link_name` text NOT NULL,                     # Nom du lien web
  `link_url` text NOT NULL,                      # Adresse web (URL) du lien
  `link_description` text NOT NULL,              # Description du lien
  PRIMARY KEY  (`link_id`)
);
```

Cette requête est placée dans le fichier texte « bdd.txt ».

Le fichier « conf.php »

```
<?php
# plugin configuration
$plugin_name="Link";
$plugin_idurl="link";
$plugin_root=ROOT."/plugin/link";
$plugin_install="0";
$plugin_active="0";
$plugin_lang=array("fr","en");
$plugin_page_admin=array("install","form_link");
?>
```

Nous voyons ici que les variables « $plugin_install » et « $plugin_active » ont pour valeur 0. Si un utilisateur télécharge le plugin, il pourra alors l'installer et l'activer via l'espace d'administration.

Nous avons configuré le plugin pour qu'il soit disponible en français (fr) et en anglais (en). En regardant les fichiers de notre plugin, nous voyons effectivement qu'il existe deux fichiers de langues « lg_link_fr.php » et « lg_link_en.php ».

L'accès aux pages d'installation « install » et le formulaire d'ajout de lien « form_link » doit être réservé aux administrateurs. Nous le précisons dans le tableau « $plugin_page_admin » ce qui permet d'éviter l'ouverture de ces pages par un simple internaute.

Le fichier « include.php »

```
<?php
# on inclut les fichiers nécessaires au plugin
include_once(ROOT."/plugin/link/tpl_link.php");
include_once(ROOT."/plugin/link/lg_link_".LANG.".php");
include_once(ROOT."/plugin/link/sql_link.php");
include_once(ROOT."/plugin/link/conf.php");

# on vérifie que le plugin est bien installé
if($plugin_install!=1)  {
 include(ROOT."/plugin/link/install.php");
}
else {
 # par défaut on affiche la liste des liens
 if(!isset($_GET['v1'])) {
  include(ROOT."/plugin/link/link_list.php");
 }
 else {
  # selon la page demandée, on fait aux différents fichier PHP
  switch($_GET['v1']) {
   case "form_link" :  include(ROOT."/plugin/link/form_link.php");
break;
   case "link_list" :  include(ROOT."/plugin/link/link_list.php");
break;
   default : include(ROOT."/plugin/link/link_list.php");
  }
 }
}
?>
```

Ici, nous incluons les fichiers de templates, de langue, de requêtes SQL et de configuration du plugin. Nous utilisons pour cela les constantes ROOT et LANG de phpMySport.

Nous regardons ensuite quel fichier du plugin doit être appelé. Il y a plusieurs cas possible :

- on souhaite installer le plugin, on fait alors appel au fichier « install.php »
- on souhaite ajouter/modifier un plugin, on appelle alors le fichier « form_link.php »
- on souhaite afficher la liste des plugin, on inclut alors le fichier « link_list.php »

Le fichier « install.php »

Le code du fichier d'installation étant assez long, nous allons seulement décrire son fonctionnement. Vous pouvez néanmoins visualiser le code puisque ce fichier est fourni par défaut avec phpMySport.

L'installation du plugin se passe en 2 temps :

- tout d'abord une page présente les fonctionnalités du plugin à l'utilisateur et un formulaire lui de demande de confirmer l'installation.
- Si l'utilisateur valide le formulaire, le processus d'installation est alors lancé : la nouvelle table « link » est créée dans la base de données et le fichier « conf.php » est mis à jour pour indiquer que le plugin a été installé et est actif.

Les fichiers « link_list.php » et « link_list.html »

Le code du fichier « link_list.php » permet de récupérer la liste des liens contenus dans la table « link » de la base de données. C'est également à travers lui qu'il va être possible de supprimer un lien.

Voici une partie du code correspondant à la récupération de la liste des liens :

```
# requête SQL de sélection de tous les liens
$sql_link=sql_replace($sql['link']['select_link_condition'],$var);

$sgbd = sql_connect(); # connexion à la base de données
$res_link = sql_query($sql_link); # on exécute la requête
$nb_ligne=sql_num_rows($res_link); # on compte le nombre de résultat
if($nb_ligne=="0")
{
 # s'il n'y a pas de résultat, on affichera un message d'erreur
 $page['L_message']=$lang['link']['E_link_not_found'];
}
else
{
 # on récupère les résultats un par un dans la variable $page['link']
 $i="0";
 while($ligne = sql_fetch_array($res_link))
 {
  $page['link'][$i]['id']=$ligne['link_id'];
  $page['link'][$i]['name']=$ligne['link_name'];
  $page['link'][$i]['url']=$ligne['link_url'];
  $page['link'][$i]['description']=$ligne['link_description'];
  $page['link'][$i]['L_show_view']=$lang['link']['show_view'];
  $i++;
 }
}
sql_free_result($res_link);
sql_close($sgbd); # on ferme la connexion à la base de données

# titre de la page
$page['L_title']=$lang['link']['link_list'];

# fichier template correspondant
$page['template']=$tpl['link']['link_list'];
```

La liste des liens est ici récupérée au sein de la variable $page['link']. Cette variable est en fait un tableau qui regroupe toutes les informations sur les liens et qui va pouvoir être exploité par le fichier « link_list.html ».

Voici le code HTML de ce fichier et les aspects visuels correspondants :

Code HTML du template « link_list.html »	Affichage dans un navigateur web	Exemple avec des liens web
`<h1>{L_title}</h1>` `<!-- LOOP link -->` `<div>` `<h2>{name}</h2>` `<p>{description}` ` ` `<a href="{url}">{url}</a>` `</p>` `</div>` `<!-- END LOOP link -->`	**{L_title}** **{name}** {description} {url}	WEBSITES LIST My link Desccription of the link http://www.mylink.com PhpMySport PhpMysport official website http://phpmysport.sourceforge.net

Constantes et fonctions communes à utiliser

Constantes

Voici la liste des constantes utilisées dans le logiciel. Elles sont définies dans le fichier de configuration « include/conf.php » et dans le fichier « index.php ».

Nom de la constante	Type	Exemple	Description
ROOT	text	/var/www/site	Répertoire racine du site. A utiliser pour l'inclusion de fichier ou de classes PHP. Le dernier caractère ne doit pas être « / »
ROOT_URL	text	http://www.monsite.com	Adresse web (URL) du site. Le dernier caractère ne doit pas être « / »
SITE_TITLE	text		Titre du site
CLUB	entier	0	Identifiant du club par défaut (égal à 0 si phpMySport est configuré en mode comité)
SGBD_HOST	text	localhost	Adresse du serveur Apache
SGBD_USER	text	root	Nom d'utilisateur du serveur
SGBD_PWD	text	password	Mot de passé du serveur
SGBD_NAME	text	mabase	Nom de la base de données
VERSION	text	1.1	Version de phpMySport utilisée
MAX_FILE_SIZE	entier	2000000	Taille maximum des fichiers à uploader (en octet)
URL_REWRITE	booléen	0	Active (1) ou non (0) la réécriture des url complexes
SENDER_EMAIL	text	contact@monsite.com	Adresse mail pour l'envoi du courrier
SENDER_NAME	text	Martin	Nom de l'expéditeur
NB_MAX_PLAYER	entier	11	Nombre maximum de joueurs par équipe
SITE_OPEN	text	1	Définie si le site est ouvert (1) ou en construction (0)
LANG	text	fr	Définie la langue du site
MEMBER	booléen	1	Définie si l'utilisateur est un membre connecté (1) ou non (0)
MEMBER_ID	entier	12	Identifiant du membre qui est connecté. S'il s'agit d'un simple

			visiteur, alors il vaut 0.
ADMIN	booléen	1	Définie si l'utilisateur a le statut d'administrateur (1) ou non (0)

Fonctions

Voici la liste des fonctions et procédures définies dans phpMySport.

Fonctions générales

Les fonctions suivantes sont définies dans le fichier « include/fonction.php »

convert_url($s[,$c])
Description : convertit une url php en url html
Entrée : $s est l'url à convertir, $c est un booléen qui mis à 0 empêche la réécriture
Sortie : url convertie
Exemple :

```
convert_url("index.php?&r=member&v1=sign_up");
// retourne : /member/sign_up.html
```

convert_date_sql($date)
Description : convertit une date au format SQL aaaa-mm-jj
Entrée : $date : date au format
Sortie : date au format aaaa-mm-jj
Exemple :

```
convert_date_sql("08-07-2007");
// retourne : 2007-07-08
```

date_day($format)
Description : retourne la date du jour selon le format indiqué
Entrée : $format est un format de date (voir la fonction date() de PHP)
Sortie : date du jour formatée

generate_pagination($url, $nb_page,$page_num,[$end_url])
Description : génère une liste de page
Entrée : $url : url de la page php
$nb_page : nombre de page au total
$nb_max : nombre d'élément par page
$page_num : numéro de la page en cours
$end_url : éventuelles variables de fin d'url
Sortie : tableau contenant la liste des pages associées à leur url

html2txt($text)
Description : converti une chaîne de caractère en text brut sans balise HTML, ni ponctuation. Les espaces sont conservés.
Entrée : $text est la chaîne de caractère à convertir en text brut
Sortie : texte brut sans code HTML ni ponctuation
Exemple :

```
html2txt("<img src=image.jpg> Voici une image !");
// retourne : voici une image
```

unhtmlentities($chaineHtml)
Description : remplace les caractères html par leur équivalent texte

text_replace($text,$var)
Description : remplace dans $text l'élément {xxx} par la valeur de la variable $var['x'] correspondante
Entrée : $text est la chaîne de caractère contenant les éléments à remplacer
$var est un tableau contenant les valeurs de remplacement
Sortie : texte avec les éléments remplacés
Exemple :

```
$var["exemple"]="Exemple de texte";
text_replace("Voici le code à remplacer : {exemple}",$var);
// retourne : Voici le code à remplacer : Exemple de texte
```

text_tronquer($text,$nb[,$b])
Description : tronque un texte après un nombre de caractères souhaité
Entrée : $text est le texte à tronquer
$nb est le numéro de caractère où il faut tronquer
Si $b est à 1, on force la coupe du mot. Par défaut $b est à 0
Sortie : texte tronqué
Exemple :

```
$text="Exemple de texte trop long à tronquer";
text_tronquer($text,20);
// retourne : Exemple de texte trop...
```

return_bytes($val)
Description : converti la taille d'un fichier en octet
Entrée : nombre en go, ko ou ko
Sortie : nombre en octet
Exemple :

```
return_bytes("2 mo");
// retourne : 2048
```

filesize_format ($bytes, [$decimal, $spacer, $lowercase])
Description : modifie le format d'affichage de la taille d'un fichier

ecartDate($date1,$date2)
Description : retroune l'écart en secondes entre deux dates au format aaaa-mm-jj hh:mm:ss

Fonctions de base de données

Les fonctions suivantes sont définies dans le fichier
« include/sgbd/mysql/mysql_fonctions.php »

sql_replace($sql,$var)
Description : identique à la fonction text_replace()

sql_connect()
Description : se connecte au serveur et sélectionne la base de données
Entrée : rien
Sortie : identifiant de connexion

sql_close($id_connection_bd, $resultat = false)
Description : ferme la connexion à la base de données et au serveur SQL
Entrée : $id_connexion est l'identifiant de la connexion à fermer
Sortie : rien

sql_query($sql, $resultat = false)
Description : exécute une requête SQL
Entrée : $sql est la requête SQL à exécuter
Sortie :

sql_num_rows($result)
Description : compte le nombre de lignes de résultats d'une requête
Entrée : $result est le résultat provenant de sql_query()
Sortie : nombre de résultats

sql_result($resultat)
Description : retourne le résultat d'une requête

sql_fetch_array($resultat)
Description : retourne un tableau contenant les valeurs d'une ligne

sql_insert_id($sgbd)
Description : retourne l'identifiant de la dernière ligne insérée

sql_free_result($resultat)
Description : libère les résultats d'une requête

function sql_error()
Description : retourne un message d'erreur en cas de problème avec une requête SQL

Fonctions de traitement des données issues d'un formulaire

Les fonctions suivantes sont définies dans le fichier « include/form/fonctions.php »

format_txt($txt)
Description : formate un texte en remplaçant les caractères spéciaux par leur équivalent HTML

check_text($text)
Description : vérifie qu'un texte ne contient aucun chiffre

check_integer($nb)
Description : vérifie qu'un nombre ne contient aucun caractère de texte

check_email($email)
Description : vérifie que la syntaxe d'un email est correcte

check_date($date)
Description : vérifie qu'une date soit bien dans l'un des formats jj-mm-aaaa, jj/mm/aaaa, jj.mm.aaaa ou aaaa/mm/jj

check_hour($hour)
Description : vérifie qu'une heure soit bien au format hh:mm:ss

check_login($text)
Description : vérifie le format d'un login (pas d'espace, pas d'accent, entre 4 et 10 caractères)

check_file_name($text)
Description : vérifie le format du nom d'un fichier (pas d'espace, pas d'accent, entre 1 et 100 caractères)

check_url($url)
Description : vérifie que la syntaxe d'une URL est correcte

check_formula($formula)
Description : vérifie que la syntaxe d'une formule (statistique) est correcte

Fonctions pour les templates

Les fonctions suivantes sont définies dans le fichier « include/template/fonctions.php »

parse_template($tpl,$var)
Description : récupère le contenu d'un fichier template et remplace les éléments de code template par leurs valeurs respectives

parse_html($code,$var)
Description : remplace les éléments d'un code template par leurs valeurs respectives

Conventions et règles de programmation

Ces règles sont pour la plupart issues du « phpBB Coding Standard Guidelines » proposé par le groupe phpbb, concepteur du célèbre forum.

Conventions de nom

L'anglais est la langue utilisée pour l'ensemble du logiciel. Il est recommandé de l'utiliser pour le nom des fichiers, des variables, des constantes et des fonctions.

Variables

Minuscule, mots séparés par un tiret bas « _ », caractères alphanumérique (sans accents). Les nombres sont autorisés mais déconseillés. Les noms courts sont préférés mais ils doivent rester suffisamment compréhensibles.

Exemple incorrect : `$nommembre, $NomMembre, $nom_du_membre`

Exemple correct : `$nom_membre`

Constantes

Même convention que pour les variables mais en majuscule.

Exemple incorrect : `define("Site_url");`

Exemple correct : `define("SITE_URL");`

Indices de boucle

Seul cas où les variables auront un caractère : `$i`, `$j` ou `$k`

Fonctions : nom et paramètres

Même convention que pour les variables.

Structure du code

Indentation et espace

L'indentation du code se fera par des espaces et non des tabulations. Ne pas mettre d'espace inutiles : utiliser des espaces entre les différentes entités, mais pas entre les structures communes :

Exemple incorrect : `if ( $tmp == 1 && $k != true )`

Exemple correct : `if($tmp==1 && $k!=true)`

Les accolades

Toujours utiliser les accolades, même si elles ne sont pas obligatoires pour le fonctionnement du script. Leur position doit se faire comme suit :

Exemple incorrect

```
if (condition)    action();
if (condition)
 action();
while (condition)
```

Exemple correct

```
if (condition)
{
 action();
}
```

```
 action();
for ($i = 0; $i < size; $i++)
 action($i);
```

```
while (condition)
{
 action();
}
for ($i = 0; $i < size; $i++)
{
 action();
}
```

Autres règles

Utiliser des guillemets simples

Exemple incorrect :

```
$member_name="Nom du membre";
action("$member_name");
$_POST[member_name];
```

Exemple correct :

```
$member_name='Nom du membre';
action($member_name);
$_POST['member_name'];
```

Concaténer les textes et les variables

Exemple incorrect : `$text="Début $variable et fin du texte";`

Exemple correct : `$text="Début ".$variable." et fin du texte";`

Ne pas utiliser des variables non initialisées

Exemple incorrect : `if($nom_membre)`

Exemple correct : `if(isset($nom_membre))`

Commenter votre code

Un code est d'autant plus facile à comprendre qu'il est correctement commenté. Les utilisateurs qui souhaiteront analyser le code de phpMySport auront sûrement déjà un minimum de connaissances en PHP ou au moins en informatique. Il n'est donc pas utilise de sur-commenter le code en expliquant tout point par point. En revanche, le fonctionnement général du programme ainsi que la signification des nouvelles variables doivent être explicités correctement. Les commentaires doivent être placés entre « /* » et « */ » ou après le caractère « # ».

Crédits

Auteurs

Le logiciel phpMySport a été conçu par Jérôme PLACE, bio-informaticien de formation et amateur de sport. Il est également le webmestre du site Cité Sport (www.citesport.com) dont l'objectif est de promouvoir les clubs de sports amateurs et le sport dans son ensemble. En réalisant ce logiciel, il a souhaité mettre à profit son expérience en conception de site web pour toutes les associations sportives et les amoureux de sport.

Version

Version du logiciel : 1.2

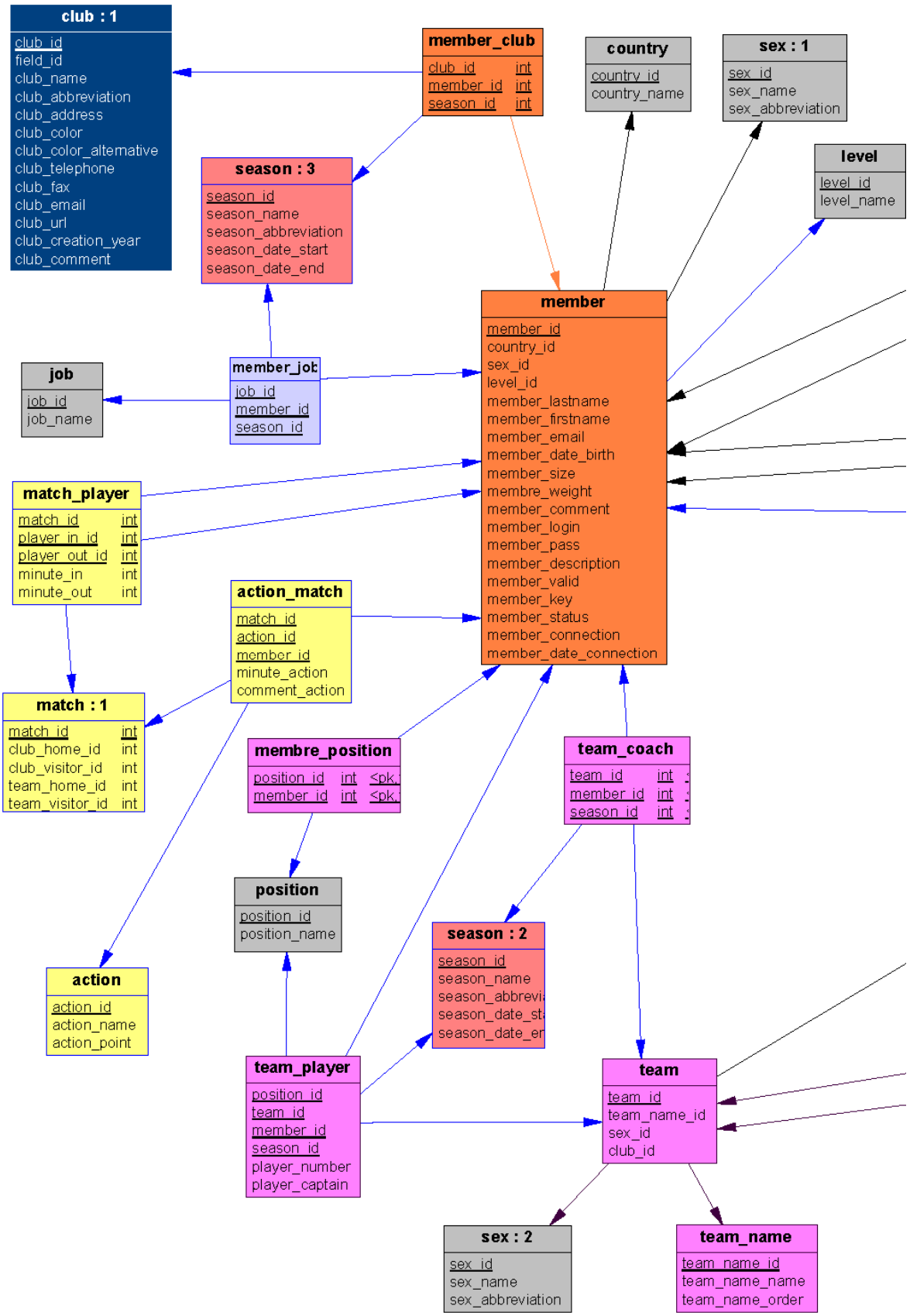
Logiciel sorti le : 26/11/2006

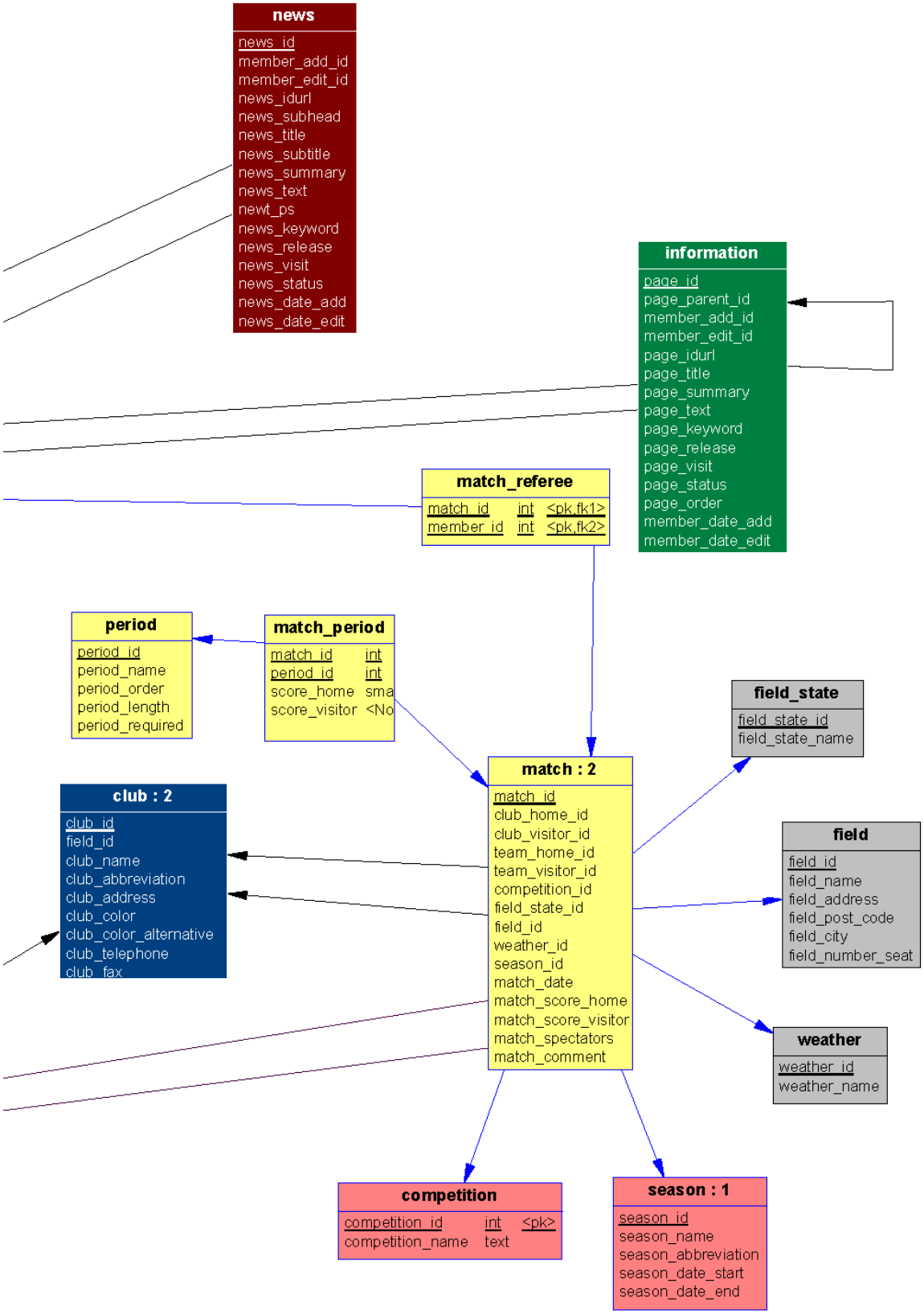
Manuel publié le : 30/08/2007

Rédaction : Jérôme PLACE

Annexes

Modèle physique de la base de données (phpMySport v1.1)




news
news_id
member_add_id
member_edit_id
news_idurl
news_subhead
news_title
news_subtitle
news_summary
news_text
newt_ps
news_keyword
news_release
news_visit
news_status
news_date_add
news_date_edit
information
page_id
page_parent_id
member_add_id
member_edit_id
page_idurl
page_title
page_summary
page_text
page_keyword
page_release
page_visit
page_status
page_order
member_date_add
member_date_edit
match_referee
match_id int <pk,fk1>
member_id int <pk,fk2>
period
period_id
period_name
period_order
period_length
period_required
match_period
match_id int
period_id int
score_home sma
score_visitor <No
field_state
field_state_id
field_state_name
match : 2
match_id
club_home_id
club_visitor_id
team_home_id
team_visitor_id
competition_id
field_state_id
field_id
weather_id
season_id
match_date
match_score_home
match_score_visitor
match_spectators
match_comment
club : 2
club_id
field_id
club_name
club_abbreviation
club_address
club_color
club_color_alternative
club_telephone
club_fax
field
field_id
field_name
field_address
field_post_code
field_city
field_number_seat
weather
weather_id
weather_name
competition
competition_id int <pk>
competition_name text
season : 1
season_id
season_name
season_abbreviation
season_date_start
season_date_end