Architecture des applications de la toile - FLIN610

Michel Meynard

19 janvier 2009

Table des matières

Chapitre 1

Introduction

1.1 Prérequis

Ce cours est destiné à des étudiants ayant les connaissances de bases du langage HTML ainsi que du protocole HTTP. Les notions de serveur HTTP, par exemple Apache, et de client léger, navigateur Web, ainsi qu'un minimum de connaissance de Javascript sont supposées acquises. L'ouvrage [1] est indiqué pour s'initier à ces techniques de base.

L'étudiant devra être familier avec la conception de sites statiques à l'aide d'outils de son choix : emacs, Dreamweaver, ... En TP, il devra être capable de rechercher les références concernant HTML, Javascript, les feuilles de style CSS.

1.2 Objectifs

Les objectifs de ce cours sont nombreux :
- apprentissage d'un langage de programmation côté serveur PHP ;
- étude de la dynamique d'une application Web : initialisation, saisie, traitement, ...
- technologies avancées : cookies, sessions, téléchargement, ...
- réalisation d'un projet de site marchand.

1.3 HTTP et architecture du Web

Quelques définitions de base sont nécessaires :

HTTP `HyperText Transfer Protocol` est un protocole de communication **client-serveur** développé pour le World Wide Web (www). HTTP a été inventé par Tim Berners-Lee avec les adresses web (URL) et le langage HTML pour créer le World Wide Web. Il utilise généralement le port 80 et est un protocole **non orienté connexion**. Schématiquement, le client envoie une requête au serveur qui lui retourne une réponse générallement sous la forme d'un fichier HTML. Le client affiche alors la réponse. Cette réponse contient généralement des **liens** hypertextes ou des formulaires, qui une fois cliqués ou soumis génèrent une requête au serveur ...

serveur HTTP par exemple, Apache, IIS, ... sont des serveurs installés sur des machines.

client HTTP appelé aussi client léger ou navigateur Web, les plus connus sont Firefox, Internet Explorer, Opera, ...

URL de l'anglais *Uniform Resource Locator* est une chaîne de caractères codée en ASCII pour adresser les ressources du World Wide Web : document HTML, image, son, forum Usenet, boîte aux lettres électroniques, etc. Elle est informellement appelée une **adresse web**. Exemples : `http://www.google.fr/`, `mailto:toto@titi.fr`, `http://www.lirmm.fr/~meynard/Ens2/rubrique.php3?id_rubrique=36&x=1`, ...

URL relative à l'intérieur d'une page HTML, des liens vers d'autres pages du même site peuvent être définis avec une écriture relative au répertoire courant de la page : par exemple, `../images/toto.jpg` est une URL relative.

Ce qu'il faut retenir absolument, c'est que HTTP n'est pas orienté connexion c'est-à-dire que le serveur ne mémorise aucune information à propos du client et qu'entre 2 requêtes du client, il peut se passer 2 secondes ou un temps infini ! Par conséquent, l'écriture des applications côté serveur doit prendre en compte cette absence de mémoire.

D'autre part, une part non négligeable du traitement dans une application peut être effectué côté client afin d'effectuer des vérifications (contrôles) qui éviteront de surcharger le serveur. Cependant, des règles de sécurité interdisent aux scripts côté client d'accéder au ressources du poste client (fichiers, imprimante, ...).

1.3.1 Côté Serveur

L'application côté serveur utilise deux technologies possibles :

cgi le programme est un script (python, bash) ou un binaire (compilé par g++) qui est chargé dans un processus externe au serveur http. La sortie standard de ce processus est redirigé dans la réponse envoyée par le serveur au client. L'inconvénient principal des cgi est la perte de temps nécessitée par la création d'un processus pour chaque nouvelle requête ;

module certains serveurs dont Apache ont intégré des modules interprétant des langages de programmation tels que Perl, Python, PHP. Ainsi, l'interprétation des scripts est beaucoup plus rapide.

1.3.2 Côté Client

L'application côté client peut utiliser plusieurs technologies possibles :

javascript langage de script interprété par le navigateur et permettant de manipuler l'arborescence du document (DOM) ;

applet Java mini application Java permettant d'utiliser toute la puissance du langage Java (API, structures de données, ...)

ActionScript langage de script compatible avec JavaScript et permettant de réaliser des animations Flash ...

Chapitre 2

Le langage HTML et XHTML

2.1 Introduction

"HyperText Markup Language", abrégé HTML, est le langage conçu pour représenter les pages web. En août 1991, Tim Berners-Lee (CERN) annonce publiquement le web sur Usenet, en donnant l'URL d'un document de suffixe .html. A l'origine, HTML est basé sur SGML jugé trop complexe. "eXtensible HyperText Markup Language", abrégé XHTML, est le langage destiné à remplacer HTML 4.0 et est une application XML. Par la suite, nous utiliserons toujours du XHTML.

A l'origine, contenu et format de document étaient mélangés dans un même fichier HTML. C'est-à-dire que l'on peut trouver dans un document HTML, des éléments de :

- structuration du contenu tels que `div`, `h1`, `h2`, ..
- mise en forme tels que `b` (bold), `i` (italic),`center`, ...

Actuellement, on utilise des feuilles de style (en cascade) "Cascading Style Sheets" pour le format. La plupart du temps, la feuille de style est externe c'est-à-dire stockée dans un fichier différent référencé (lié) par le fichier HTML. Dans certains cas, un style en-ligne (au fichier) peut être associé.

Actuellement, c'est le W3C (*World Wide Web Consortium*) qui est chargé de rédiger des recommandations (sorte de norme) concernant les technologies du web. L'adresse web de leur site est `http://www.w3.org/`. On peut notamment valider ses documents en ligne à l'adresse `http://validator.w3.org/`. Un autre site d'importance et qui n'a aucun lien avec le W3C est `http://www.w3schools.com/` qui est un site pédagogique très pratique même s'il est commercial.

2.1.1 Vocabulaire

- document : l'ensemble du texte composé du fichier principal et des fichiers inclus ou référencés (script, feuille de style, image, ...) ;
- langage à balisage : `<balise>contenu ...</balise>` ;
- élément : composé d'une balise ouvrante puis d'un contenu (possédant éventuellement d'autres éléments) puis de la balise fermante correspondante ;
- élément vide : ne possédant pas de contenu, la balise ouvrante est également fermante comme dans `<br />` ;
- attribut : information de la forme `nom='valeur'` présente uniquement dans une balise ouvrante ;
- validité : un document HTML est valide s'il correspond aux règles édicté par le "World Wide Web Consortium" (w3c) ;
- URI : *Uniform Ressource Identifier* adresse d'une ressource Internet composé d'un protocole, d'un nom de domaine complètement qualifié (FQDN), d'un chemin, et d'autres paramètres possibles ; Par exemple, `http://www.lirmm.fr/~meynard/ProjetInfoL3/index.php?rubrique=Projet&action=liste` est une URI. Autre exemple, `mailto:toto@tutu.fr` désigne une adresse email ;

2.2 Syntaxe de XHTML

2.2.1 Généralités

- les noms d'attribut sont en minuscules ;
- les valeurs d'attribut doivent être entre apostophes ou bien entre guillemets ;
- les éléments vides sont définis par une seule balise, comme dans `<br />` et dans `<img src="toto.jpg" />` ;
- la déclaration d'une DTD est obligatoire ;

Exemple 1 (Exemple de document XHTML minimum : modele.html)

```
<!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="fr" lang="fr">
<head>
<meta http-equiv="content-type" content="text/html; charset=ISO-8859-1" />

<title>mon premier document</title>
</head>
<body>
<h1>Titre de niveau 1</h1>
<p>hello World !</p>
</body>
</html>
x
```

2.2.2 Définition de la DTD

Une DTD (Document Type Definition) définit la syntaxe d'un type de document : c'est une grammaire formelle indiquant les imbrications possibles d'éléments, les attributs obligatoires, optionels, ...

La déclaration de la DTD `<!DOCTYPE ...` est obligatoire afin que le navigateur sache quel type de document gérer. Cette déclaration localise le fichier de DTD. Remarquons que plusieurs DTD pour XHTML 1 existent (strict, transitional, frameset), et que nous choisissons la transitoire qui permet certains éléments de présentation (center , applet, ...).

2.2.3 L'élément html

C'est la racine du document XML et doit par conséquent définir l'espace de nom XML grâce à l'attribut `xmlns`. Il doit contenir à sa suite 2 éléments :

head l'en-téte du document ;

body le corps du document qui sera affiché sur la page ;

2.2.4 L'en-tête du document

L'élément `head` peut et **doit** contenir d'autres éléments que le titre du document `title` :

- l'élément `meta` fournit des méta-informations sur la page en cours qui pourront être utilisées par les moteurs de recherche ou par le navigateur. L'attribut `content` est obligatoire et définit la valeur de la méta-information. L'attribut `name` est optionel et peux prendre les valeurs `author`, `description`, `keywords`. Par exemple,
  `<meta name="keywords" content="fromage, camembert, produit frais" />`
  Un autre attribut optionel très important est `http-equiv` puisqu'il permet d'envoyer des en-têtes HTTP au navigateur pour l'aider à interpréter le document. Par exemple, `<meta http-equiv="refresh" content="5" />` permet de raffraîchir la page au bout de 5 secondes. Autre exemple important qui définit le codage utilisé,
  `<meta http-equiv="content-type" content="text/html; charset=UTF-8" />`
  indique le type de document à afficher au navigateur.
- l'élément `script` permet de référencer un fichier script externe et/ou de définir des fonctions JavaScript locales. Par exemple,
  `<script type="text/javascript" charset="iso-8859-1" src="javascript.js"></script>`
  Attention à ne pas créer un élément vide `<script ... />` lorsque tout le code est dans un fichier externe.
- l'élément `link` permet de lier un autre document. Par exemple, pour définir la feuille de style associée :
  `<link rel="stylesheet" href="stylesheet.css" type="text/css" />`

2.3 Le corps du document

Le corps (*body*) du document HTML constitue ce qui va être affiché sur l'écran. Par défaut, les éléments sont affichés selon leur type :

- les éléments en ligne (*inline*) sont placés de gauche à droite jusqu'à ce qu'il n'y ait plus de place dans leur bloc et un retour à la ligne automatique est géré par le navigateur ; ainsi le texte d'un paragraphe est il affiché avec des coupures de lignes dépendant de la taille du navigateur ; de même les ancres sont des éléments en ligne ;

– les éléments de type bloc sont affichés avec une coupure de ligne avant et après eux ; les paragraphes sont de type bloc ainsi que les en-têtes.

2.3.1 Attributs de base

Un certain nombre d'attributs de base communs à quasiment tous les éléments (*core attributes*) peuvent être utilisés.

class classe CSS de l'élément. Cela permet de formatter l'élément selon un style défini dans une classe déjà définie ; par exemple, `class='monvert'` où monvert est une classe de style ;

style style en ligne ; par exemple, `style='color :green ;'`

id identifiant **unique** de l'élément permettant sa manipulation en JavaScript grâce à la méthode : `document.getElementById(id)` ;

title *tooltip* affiché lors du survol de l'élément ;

De plus, une gestion des événements souris et clavier permettent d'associer des scripts JavaScript. Cette association est réalisée grâce à des attributs tels que `onclick` ou `onkeypress`.

2.3.2 Eléments indispensables

a *anchor* les ancres mettent en oeuvre l'hypertexte et peuvent être utilisées de 2 façons :

1. lien hypertexte référençant une autre page web grâce à l'attribut `href`. Par exemple, pour aller sur le site du `w3schools` :

   ```
   <a href='http://www.w3schools.com/'>CLIQUEZ ICI</a>
   ```

2. marque-page indiquant une cible potentielle d'un lien :

   ```
   <a id='toto'>Texte cible</a>
   ```

   Plus loin dans le même document, on pourra référencer cette cible par le lien suivant :

   ```
   <a href='#toto'>aller au texte cible</a>
   ```

   Bien entendu, on peut faire une référence externe sur une cible en suffixant le chemin de l'url par `#toto` ;

Enfin, la valeur d'un href peut-être une pseudo-URL contenant du code JavaScript qui sera exécuté lors de l'activation du lien comme dans l'exemple suivant :

```
<a href="javascript:alert('Bonjour le monde !')">Mon Lien</a>
```

Remarquons que le code est préfixé par `javascript :` afin d'indiquer le langage ;

script exécution de code dans le corps de page ; par exemple :

```
<script type="text/javascript">document.writeln('Bonjour le monde !');</script>
```

p paragraphe contenant une suite de phrases ;

br *break* élément vide qui permet de passer à la ligne suivante ; en HTML les espaces, tabulations et retour ligne (CR et/ou LF) multiples ne sont perçus que comme un espace séparateur lorsqu'ils sont situés entre deux mots.

h1 *header* de niveau 1 est un titre de section ; on trouve également `h2` ...

ul *unordered list* liste d'items non numérotés ;

ol *ordered list* liste d'items numérotés ;

li *list item* élément d'une liste numérotée ou non ;

div *division* est une section logique du document permettant de regrouper plusieurs blocs dans un même formatage ; on l'utilise souvent dans la mise en page du document pour scinder les différentes parties (bandeau haut, menu de gauche, page centrale, bandeau du bas) ;

table les tables HTML sont un moyen d'afficher les tableaux mais aussi de mettre en page un document (comme les div). Les tables peuvent ou doivent contenir :

caption la légende de la table ;

tr *table row* une ligne ;

td *table delimiter* une case ; `colspan` et `rowspan` sont des attributs indiquant le nombre de cases à fusionner avec la case courante vers la droite et vers le bas ;

th *table header* une case de titre de colonne ;

hr *horizontal rule* ligne de séparation horizontale ;

img *image* en format gif, png ou jpg ; les attributs indispensables :

src *source* chemin du fichier image ;

alt *alternative* texte utilisé si le navigateur ne sait pas afficher les images ;

width largeur en pixels ;

height hauteur en pixels ;

form les formulaires sont décrits dans la section suivante.

2.4 Les formulaires

Les formulaires constituent des éléments indispensables dans la saisie d'informations à destination d'une application web. Un ou plusieurs formulaires peuvent être présents dans une même page. L'attribut `name` de chaque champ de saisie correspond au nom du "paramètre" qui contiendra l'information. Plusieurs champs peuvent avoir le même nom, dans ce cas le paramètre sera de type tableau. L'attribut `tabindex` permet d'ordonner les champs de saisie quand l'utilisateur appuiera sur la touche de tabulation.

2.4.1 Eléments de formulaires

form élément racine d'un formulaire contenant les attributs suivants :

action est l'url où sera soumis le formulaire (généralement un script php où un programme cgi) ;

method soit get, soit post ; la première (get) insère les champs saisis par l'utilisateur à la fin de l'url après un point d'interrogation sous la forme `http://localhost/index.php?nom1=val1&nom2=val2` ; les noms sont les valeurs des attributs `name` des champs tandis que les valeurs sont les contenus saisis par l'utilisateur. La seconde méthode post "cache" les contenus saisis ;

target avec la valeur `_blank`, une nouvelle fenêtre sera ouverte, avec `_self` (par défaut) on recouvrira la fenêtre actuelle.

onsubmit code javaScript à exécuter avant la soumission ; Si le code JavaScript retourne faux, la soumission **est annulée** !

name ou id nom ou id du formulaire pouvant être utile pour le référencement des champs de saisie dans le code javaScript de vérification ;

input élément **vide** définissant un champ de saisie ayant :

- un attribut **name**, le nom du champ ;
- un attribut **type** permettant d'indiquer la forme du champ de saisie parmi :

text champ de type texte libre sur une ligne ; autres attributs : `size` la taille du champ de saisie à l'écran, `maxlength` le nombre maximum de caractères à saisir, `value` valeur par défaut ;

password champ de mot de passe caché ;

button bouton permettant de déclencher un script javascript ; autres attributs : `value` valeur affichée à l'écran, `onclick` code javaScript à déclencher ;

checkbox case à cocher ; autres attributs : `value` valeur affectée au nom dans le cas où cette case est cochée, `checked="checked"` si on veut que la case soit cochée par défaut ;

radio case à cocher **exclusive** ; tous les boutons radio mutuellement exclusifs doivent avoir le même attribut `name` ; autres attributs : `value` valeur affectée au nom dans le cas où cette case est cochée, `checked="checked"` si on veut que la case soit cochée par défaut ;

hidden champ caché n'apparaissant pas dans le formulaire ; historiquement, les champs cachés permettaient de faire transiter l'information passée de proche en proche lors de la navigation ; autres attributs : `value` ; Actuellement, le mécanisme de session est plus pratique à utiliser.

submit bouton de soumission permettant d'exécuter l'`action` du formulaire ; plusieurs boutons de soumission peuvent coexister dans un même formulaire ;

image bouton de soumission utilisant une image comme visuel ; autres attributs : `src`, `alt`,`width`, `height` ;

reset réinitialisation des champs du formulaire ;

file pour envoyer au serveur un fichier localisé chez le client ; le formulaire doit posséder un attribut `enctype` ayant la valeur `multipart/form-data` et sa méthode doit être `post` ; Il est également possible de limiter la taille du fichier à envoyer en ajoutant un champs caché nommé `max_file_size` et de valeur la taille maximal acceptée en octets.

textarea élément non vide contenant une zone de texte multi-ligne ; attributs : `name`, `rows` nb de lignes, `cols` nb de colonnes ;

select élément non vide contenant une liste déroulante d'options ; attributs : `name`, `size` nb de lignes visibles, `multiple` de valeur `multiple` indique que plusieurs options sont possibles ;

option élément non vide contenu dans `select` ; attributs : `value` valeur qui sera associée au `name` du `select` conteneur, `selected` de valeur `selected` indique si cette option est présélectionnée ; le contenu de l'élément option sera le texte affiché dans la liste déroulante ;

Exemple 2 (Un formulaire de login)

```
<form action="login.php" method="post" >
Nom : <input type="text" size="20" name="login"><br />
Mot de passe : <input type="password" size="20" name="passwd"><br />
<input type="submit" value="Valider">
</form>
```

2.4.2 Inclusion d'autres objets

L'élément `Object` permet d'inclure un objet typé dans une page HTML. Ses attributs sont :

type indique le type de l'objet tel que : "text/html", "application/x-shockwave-flash", "video/x-msvideo", ...

data url du fichier tel que : `../commun.html`, `ma_video.avi`, `animflash.swf`, ...

width et `height` indique la taille de l'inclusion ;

Cet élément n'est pas vide et peut ou doit contenir des sous-éléments **paramètres** utilisés par le navigateur pour un type d'objet particulier.

Exemple 3

```
<param name="autostart" value="true" />
<param name="loop" value="true" />
Texte alternatif
```

Chapitre 3

Le langage PHP

3.1 Introduction

PHP (acronyme récursif pour PHP : Hypertext Preprocessor) est un langage de script (interprété) et Open Source, spécialement conçu pour le développement d'applications web. Il doit être **intégré** au HTML. La référence du langage est sur un site Web [5] qu'il faut toujours avoir en face des yeux quand on programme !

Exemple 4

```
<html><head><title>premier prg php</title></head><body>
<?php
  echo "Bonjour le monde !";
?>
</body></html>
```

Remarques :
- différence avec les autres scripts CGI (Perl, C, ...) : page HTML avec du code inclus à l'intérieur :
  - pas besoin de générer le code HTML ;
  - prototypage du squelette du site par des spécialistes graphiques ;
- différence avec Javascript : le code est exécuté sur le serveur ;
- langage extrêmement simple et fonctionnalités avancées ;
- l'interprète PHP peut être appelé par :
  - le serveur Web comme module ou comme CGI ;
  - l'interpréteur de commandes de votre système : `php hello.php`.

3.2 Caractéristiques

Différentes versions sont apparues au cours du temps (PHP2FI, PHP3, PH4, PHP5). Nous ne traiterons ici que de **PHP5**.

L'interprète PHP fonctionne sur le fichier interprété comme un filtre laissant passer de l'entrée standard vers la sortie standard tout ce qui n'est pas entre `<?php` et `?>`. Quant au reste, il exécute les instructions, fonctions, définitions incluses. Dans le cas où plusieurs *blocs* php existent, les définitions du premier bloc sont utilisables dans tous les autres.

Exemple 5

```
<html><head><title>deuxieme prg php</title></head><body>
<?php
$i=123;
?>
bla bla bla
<?php
echo "la variable i vaut : $i";
?>
</body></html>
```

Le fichier d'extension `.php` doit être lisible par le serveur HTTP (`chmod a+r hello.php`). Le droit d'exécution n'est pas nécessaire.

3.3 Structure du langage

Au niveau syntaxique, le langage PHP ressemble fortement au langage C à ceci près qu'il n'est pas typé ! Toutes les variables doivent être **préfixées par un $** : c'est la cause principale d'erreur syntaxique. Il faut donc répéter que TOUTES LES VARIABLES DOIVENT ÊTRE PRÉFIXÉES PAR UN $!

3.3.1 Types

Les types scalaires sont :
- integer avec des littéraux tels que : 10, -454, 0777, 0xABCD ;
- float avec des littéraux tels que : 1.24, -1.2e3, 5E-6 ;
- string avec des littéraux tels que : 'hello world', `"hello $i world"` ; Entre guillemets, les variables scalaires sont substituées par leur valeur (pas entre apostrophes) ! De nombreuses fonctions strxxx existent !
- boolean : true et false (insensibles à la casse) ;
- null : la valeur spéciale NULL représente l'absence de valeur (insensible à la casse).

3.3.2 Typage dynamique

Une variable n'est pas typée statiquement mais dynamiquement au fur et à mesure de ses affectations successives. La conversion de type est automatique et dépend du contexte.

Exemple 6
```
<?php
$x = "0";  // $x est une chaîne de caractères (ASCII 48)
$x += 2;   // $x est maintenant du type entier (2)
$x = $x + 1.3;  // $x est maintenant du type double (3.3)
$x = 5 + "10 roses"; // $x est du type entier (15)
?>
```

On peut forcer le type d'une variable avec :
- la fonction settype() ;
- ou le transtypage à la C : `$x = (int) 5.6;`

On peut tester le type d'une variable avec les fonctions booléennes suivantes : `bool is_int(mixed $var)`, `bool is_string(mixed $var)`, ...

3.3.3 Existence et valeur

La fonction `isset($x)` permet de tester l'existence d'une variable ce qui s'avère primordial dans le cadre de la soumission de formulaire HTML pour savoir si un paramètre a été posté.

Pour "tuer" une variable, il faut utiliser la fonction `unset($x)`.

Attention, il faut distinguer `isnull($x)` qui teste l'égalité de `$x` avec la valeur NULL.

3.3.4 Constantes

Comme en C, les constantes sont des macros, elles sont définies par la fonction `define()` et utilisées **sans $** !

```
define("REP_TELECHARGEMENT", "Telechargement/");
define("TAILLE_MAXI_FICHIER", 1000000);
echo TAILLE_MAXI_FICHIER;
```

3.3.5 Le type tableau associatif

En PHP, le type tableau associatif est constamment utilisé, il faut donc bien comprendre son fonctionnement. Un tableau PHP est une association (Map) de couples (clé, valeur). Les clés sont uniques et sont des entiers positifs ou des chaînes. Les valeurs sont quelconques (type et valeur). Si l'on a besoin d'un tableau à la C indicé par des entiers débutant à 0, il faut considérer ces indices comme des clés.

Lorsqu'on ajoute des valeurs sans clé dans le tableau, la clé est calculée comme un entier immédiatement supérieur à la plus grande clé entière du tableau ! **Attention**, l'ordre chronologique des ajouts est conservée lorsqu'on parcourt la liste.

Exemple 7

```
<html><head><title>tableaux php</title></head><body>
<?php
$tab=array("un"=>123, 2=>"deux", "abc"=>12.3);
echo "tab[2] : {$tab[2]}; tab['abc'] : {$tab['abc']}; tab[0] : {$tab[0]}<br/>";
$tab[]="toto"; echo "taille tab : ".count($tab)."; tab[3] : {$tab[3]}<br/>";
$tab[1]="un";
foreach($tab as $c=>$v){echo "$c => $v \n<br/>";}
?>
</body></html>
```

affiche :

```
tab[2] : deux; tab['abc'] : 12.3; tab[0] :
taille tab : 4; tab[3] : toto
```

Quelques fonctions utiles :

`count($tab)`	retourne le nombre de couples
`unset($tab['deux'])`	supprime la clé et la valeur
`isset($tab['deux'])`	teste si défini
`$tab[]='nouveau'`	ajoute un couple à la fin
`print_r($tab)`	affiche récursivement le tableau
`$compact=array_values($tab)`	compacte tab dans nouveau indicé de 0 à n-1
`$foreach($tab as $cle=>$valeur){...}`	parcours du tableau
`bool in_array ("toto",$tab)`	recherche de valeur
`$et=array(array(0,0),array(0,1))`	multi-dimensionnel
`bool array_xxx($tab)`	innombrables fonctions avec xxx valant sort, merge, diff, ...

TABLE 3.1 – fonctions utiles sur les tableaux

Avec une syntaxe complexe utilisant des accolades, on peut substituer des éléments de tableaux dans une chaîne : `echo "bonjour {$tab['michel'][5]}";`

3.3.6 Les expressions

Opérateurs

Quasiment utilisés que pour les **nombres** sauf la concaténation de chaîne (.) et l'affectation (chaînes et tableaux).

$$s valeur de la variable dont le nom est dans s

++, - - inc et déc : `++$x ;$y- - ;`

$i==$j ? 1 : 2 expression conditionnelle

arithmét. `+,-,*,/,%`

affectation `=,.=,+=,-=,*=,/=,%=`

binaires `&,|,~,<<,>>`

logiques `&&,||,!`

logique peu prioritaire `and,or`

comparaison `<,<=,==,!=,>=,>`

comparaison typée `===,!==`

commande syst. `` `ls` ``

concaténation `$s . "toto"`

L'affect. de chaîne longue (heredoc) est possible :

```
$s=<<<FIN
bla bla..
etc
FIN; // en début de ligne et avec un ``;''
```

Remarquons qu'il existe 2 types de comparaisons == et ===. Ce dernier n'est vrai que si les deux expressions sont du même type et sont égales. Cela peut être utile pour éviter les égalités dues à des conversions.

Exemple 8

```
<html><head><title>comparaison</title></head><body>
<?php
if (0=="" && null==false && $tab==0 && "2"=="02"){
  echo "BIZARRE";
}
?>
</body></html>
```

affiche : BIZARRE

Par exemple, la fonction `strpos($chaine, $facteur)` retourne false si le facteur est absent, la position comprise entre 0 et `strlen($chaine)-1` si le facteur est trouvé. Le test `if(strpos('abc','ab')){...` est incorrect car `false==0`! Il faut donc absolument faire le test suivant : `if (strpos('abc','ab') !== false) {`.

3.3.7 Structures de contrôle

Les coupures de lignes sont traitées comme les espaces. Les structures de contrôle classique à la C existent en PHP :

alternative `if (expr) {instons1} else {instons2}`

choix `if (e1) {ins1} elseif (e2) {ins2} else {ins3}`

choix `switch ($i) {case e0 : instons0 ; break ; case e1 : instons1 ; break ; default : instons ; }`

répétitives
- `while (e) {instons}`
- `do { instons } while (e)`
- `for (exp-init ; exp-cond ; exp-itér) { instons }`
- ruptures possibles : `break ;`, `continue ;`

parcours de tableau
- `foreach($tab as $val) { instons utilisant $val}`
- `foreach($tab as $cle => $val) { instons utilisant $cle et/ou $val }`

inclusion `include 'monfic.php';` inclusion de fichier

inclusion unique `include_once 'monfic.php';` inclusion au plus une fois !

3.3.8 Fonctions

Un nombre impressionnant de fonctions prédéfinies ! Passage des paramètres scalaires ou tableaux par valeur (par référence si précédé de & dans l'appel et la déclaration). Les variables globales ne sont pas accessibles directement. Il faut utiliser la notation : `$GLOBALS['glob']`. `$GLOBALS` est l'asso. des variables globales. On peut aussi déclarer les variables globales dans la fon par `global $i,$j ;` Les arguments peuvent avoir une valeur par défaut (optionnels) et être en nombre variable. Les arguments ayant une valeur par défaut doivent être à droite de ceux n'en ayant pas. Les valeurs de retour peuvent être scalaire ou tableaux. On peut supprimer l'affichage des erreurs produites par l'appel à une fonction f par : `@f($i)`

Fonctions sur les tableaux

int count($tab) taille du tableau

int array_push($pile,$elem) empile à la fin et ret la taille

mixed array_pop($pile) dépile le sommet

int array_unshift($fifo,$prem) ajoute au début

mixed array_shift($fifo) retire le premier

array array_values($asso) resp. `array_keys`

void shuffle($tab) mélange les valeurs

Fonctions sur les tris de tableau

void sort($tab) tri croissant (décroissant avec rsort) selon les valeurs

void ksort($asso) tri croissant selon les clés (krsort)

void asort($asso) tri croissant selon les valeurs (arsort)

void usort($tab, moncmp) tri croissant selon la fon de cmp définie par l'utilisateur

uksort, uasort tri utilisateur pour les asso

Fonctions sur les chaînes

De très nombreuse fonctions dont voici les principales. `$s` est supposée être une chaîne.

int strlen($s) taille

int strcmp($s1,$s2) comparaison -1, 0, 1

string substr($s, 2, 10) sous-chaîne à partir de 2, de taille 10

string strstr($s,$facteur) ret tout s à partir de la 1ère occurrence de facteur

array split(';',$s,10) ret un tableau des 10 (maxi) premiers champs séparés par des ;

string join(';',$tab) ret la chaîne constituée des valeurs séparées par ;

string trim($s) retire les blancs de début et de fin de chaîne. Blancs : `\n,\r,\t,\v,\0,espace`

string chop($s) supprime les blancs de fin de chaîne

string ltrim ($s) supprime les blancs de début

string strtolower($s) met en minuscules (resp. `strtoupper`)

string nl2br($s) remplace les `\n` par des `<br>`

Fonctions sur les expressions régulières

correspondance `int ereg('^[^.]*\.(.*)$',$s,$tabr)` met dans tabr la partie à droite d'un point dans s à partir de l'indice 1. Retourne TRUE si trouvé au moins une.

remplacement `string ereg_replace("([a-z]+)", "[\\1]",$s);` encadre les mots minuscules de s

Entrées/Sorties : système de fichier

De nombreuses fonctions de gestion du système de fichier à la C.

echo string, string ... ; affiche plusieurs arguments **chaînes**

print(string) ; affiche un arg. (**classé** dans les fons chaînes)

$desc=fopen("fic.txt","r+") ; ouverture en lecture et écriture. Modes (r, w (création ou raz), w+ , a (append), a+).

$d=fopen("c : \\data\\info.txt","r"); Windows

$d=fopen("ftp ://user :password@example.com/", "w") ; ouverture de session ftp

fclose($desc) fermeture

bool feof($desc) test la fin

$ligne=fgets($desc, 4096) lecture de la ligne (maxi 4096 octets)

$ligne=readline("un entier SVP") ; ne fonctionne pas sur le Web !

$car=fgetc($desc) ; lecture d'un car ;

fputs($desc,"chaine") ; écriture d'une chaîne ;

Fonctions utilisateur

La référence en avant est possible.

définition `function f($arg0, $arg1="toto"){instons ; return array(1, 4) ;}`

pointeur de fonction `$pf='f';list($i,$j)=$pf(1,2) ;`

Diverses fonctions prédéfinies

echo exp1, exp2, ... ; affiche les expressions ;

print(exp1) affiche une expression ;

print_r($var) affiche la variable scalaire ou tableau ou objet récursivement ;

void exit() termine le script

$l=system("ls") ; exécute une commande

void die ("message ") affiche le message puis termine

void eval("instons PHP") il faut échapper les caractères spéciaux : `eval('$' . f($i) . '= "toto";');` Ne pas oublier le " ;"

3.3.9 Le modèle objet en PHP5

Une classe est introduite par le mot-clé `class`. Elle contient des membres (variables) et des fonctions (méthodes). Le constructeur se nomme `__construct()` et le destructeur `__destruct()`. Par défaut, l'appel au constructeur de la classe parente n'est pas fait (`parent::__construct()`). Les méthode `__toString()` et `__clone()` sont également présentes.

```
class Point{var $x,$y;
  function _construct($px=0,$py=0){ //constr
    $this->x=$px;$this->y=$py;
  }
  function getxy(){
    return array($this->x, $this->y);
  }
}
$p1=new Point(); // (0,0)
$p2=new Point(3,4);
list($a,$b)=$p2->getxy();
```

3.3.10 Tableaux super-globaux

Ces tableaux super-globaux sont prédéfinis et permettent d'accéder à diverses informations.

$GLOBALS Contient une référence sur chaque variable qui est en fait disponible dans l'environnement d'exécution global. Les clés de ce tableau sont les noms des variables globales définies par `global $x;`

$_SERVER Les variables fournies par le serveur web, ou bien directement liées à l'environnement d'exécution du script courant, notamment `$_SERVER['PHP_SELF']` qui est le chemin du script en cours d'exécution par rapport à la racine web (sans les paramètres GET) ;

$_GET Les variables fournies au script via la chaîne de requête URL.

$_POST Les variables fournies par le protocole HTTP en méthode POST.

$_COOKIE Les variables fournies par le protocole HTTP, dans les cookies.

$_FILES Les variables fournies par le protocole HTTP, suite à un téléchargement de fichier.

$_ENV Les variables fournies par l'environnement

$_REQUEST Les variables fournies au script par n'importe quel mécanisme d'entrée et qui ne doivent recevoir qu'une confiance limitée. Note : lorsque vous exécutez un script en ligne de commande, cette variable ne va pas inclure les variables argv et argc. Elles seront présentes dans la variable $_SERVER.

3.4 Gestion de formulaire

Les champs de **formulaire HTML** sont accessibles via les tableaux superglobaux `$_GET` ou `$_POST` selon la méthode utilisée par le formulaire. Par exemple :

```
<form name='F' method='get' action='<?php echo $_SERVER['PHP_SELF']; ?>'>
<input type='text' name='nom'>
<select multiple name="biere[]">
 <option value="blonde">blonde</option>
 <option value="brune">brune</option>
</select>
<input type='submit' name='bouton' value='ok'/>
</form>
<?php print_r($_GET['biere']); ?>
```

Après saisie et validation, l'url suivante est requise :

```
essai.php?nom=toto&biere[]=blonde&biere[]=brune&bouton=ok
```

L'affichage suivant aura lieu :

```
Array ( [0] => blonde [1] => brune )
```

Exercice 1 Ecrire :
- une page HTML "Multiplication" contenant un formulaire ayant deux champs de texte x et y et un bouton `Multiplier !` ;
- un script php appelé par ce formulaire et qui affiche le résultat de la multiplication $x * y$;

Exercice 2 Réécrire le même exercice en un seul fichier !

3.4.1 Caractères spéciaux

Sources de séances de débogages longues et pénibles, il est important de comprendre les transformations des caractères spéciaux.
- Certains caractères (anti-slash, guillemet, NULL et apostrophe) saisis dans des champs de formulaires : (`<input ... name="champ">`) sont automatiquement **échappés** par un anti-slash par PHP (magic_quotes). La variable PHP correspondante `$_POST['champ']` (ou GET) doit donc être traitée par : `stripslashes($_POST['champ'])` pour supprimer **tous** les anti-slash générés.
- **Attention**, ceci est valable pour touts les tableaux super-globaux : GET, POST, COOKIE. En particulier, un objet sérializé dans un cookie aura été échappé si `get_magic_quotes_gpc()` est vrai ! Il faut donc enlever les slashes avant la désérialisation !
- Lors de l'affichage d'une chaîne contenant des caractères html spéciaux (`&"<'>`), il faut les transformer en entité HTML (`&quot;`). Pour afficher proprement une chaîne contenant ces caractères, il faut au préalable la traiter avec `htmlspecialchars($champ,ENT_QUOTES))`.
- **Attention**, si le champ posté doit être remis en tant que "value" dans un `<input type='text'` de formulaire, il faut donc enlever les anti-slash **puis** transformer ces caractères en entités HTML
- Par conséquent, pour un champ interactif, on écrira : `echo '<input name="champ" value="'.htmlspecialchars(stripslashes($_POST['champ'],ENT_QUOTES)).'">';`
- La suppression des slashes sur un tableau, par exemple de checkbox, pose problème ! Il faut les supprimer sur chaque élément du tableau !
- Lors de l'envoi d'une requête à un SGBD, il faut parfois échapper les caractères spéciaux tels que : `',\,",NULL`. La fonction `addslashes($chaine)` effectue ce travail.
- De même en JavaScript, la fonction `addslashes()` permetra d'échapper du code.
- Remarquons que dans une session, les caractères spéciaux ne sont pas échappés.

3.5 Administration PHP

Pour visualiser la configuration PHP, il suffit d'appeler la fonction `phpinfo()` pour voir :
- la version de php ;
- la localisation du fichier de configuration de php : `php.ini` ;
- les librairies incluses et leur configuration (mysql, gd, ...) ;
- de nombreuses autres choses importantes.

La configuration de l'interprète PHP est réalisé dans le fichier `php.ini`. Ce fichier `php.ini` contient des directives (sous forme d'affectation de variables) fondamentales :

register_globals=On si **on** alors les variables d'Environnement, Get, Post, Cookie, Serveur sont globales ; si **off** alors les variables sont accessibles via `$_POST[], ...` (Sécurité +) ;

variables_order="EGPCS" ordre de résolution des conflits de noms de variables Env, GET, POST, COOKIE, Serveur ;

magic_quotes_gpc=On permet d'anti-slasher les caractères anti-slash, guillemet et apostrophe dans les variables GPC.

3.6 Session

HTTP n'étant pas un protocole orienté connexion, chaque nouvelle requête semble être la première. Depuis PHP4, Les **sessions** PHP permettent de conserver de l'information **du côté serveur** sur la suite de requêtes émises par le même client (navigateur). Les variables enregistrées dans le tableau `$_SESSION` doivent être de type `string`. Pour les variables de type tableau, elles doivent être sérialisées. La communication des identifiants de sessions est réalisée :
- soit par cookie ce qui est le plus simple ;
- soit par URL de manière quasi-transparente pour le programmeur.

Par défaut, la durée de vie du cookie de session (0) est égale à la durée de vie du navigateur.

bool session_start() démarre une session ; doit être réalisé en tout **début de script** avant tout en-tête !

string session_id() retourne l'identifiant de session qui est une suite de chiffres hexadécimaux.

$_SESSION['z']="contenu" ; ajoute une variable de session ;

echo $_SESSION['z'] ; affiche le contenu d'une variable de session ;

bool session_is_registered("mavar") teste si $mavar a été sauvé dans la session ;

bool session_unregister("x") supprime la variable de session x ;

$_SESSION=array() ; réinitialise la session !

string serialize(mixed) retourne une chaîne composée des différents champs de l'objet ou du tableau ;

mixed unserialize(string) retourne la valeur de la variable d'origine ;

bool session_destroy() supprime toutes les données de la session ; le cookie de session sera supprimé dès la prochaine page.

bool session_register("x", "y", ...) démarre une session si pas encore fait et y enregistre les variables $x et $y ;

Afin de gérer les sessions de manière transparente, le script PHP doit tenir compte du fait que le navigateur client peut accepter ou refuser les cookies. Pour ce faire, il suffit d'indiquer dans chaque référence interne du site (href d'ancre, action de formulaire, ...), la constante `SID` comme paramètre de l'URL afin de transmettre l'identifiant de session. En effet, la constante `SID` a comme valeur :

- " chaîne vide lorsque le navigateur accepte les cookies ;
- 'PHPSESSID=0c92dbd...51ff2' lorsque le navigateur ne les accepte pas ;

Ainsi dans un formulaire, l'attribut action devra avoir la forme suivante :
`action="<?php echo "{$_SERVER['PHP_SELF']}".(strlen(SID)?'?'.SID:''); ?>"` L'exemple suivant illustre les sessions avec un historique de multiplications.

Exemple 9 (Multiplication avec mémorisation des résultats dans une session)

```
<?php
session_start();
?>
<html><head><title>Multiplication et session</title></head><body>
<h1>Multiplication et session</h1>
<?php
if(isset($_SESSION['historique'])){ //
  $historique=unserialize($_SESSION['historique']); // tableau
  //print_r($historique);
  foreach($historique as $tab){ // tableau 'x'=>2, 'y'=>3, 'r'=>6
    echo "{$tab['x']} * {$tab['y']} = {$tab['r']}<br/>\n";
  }
  echo "<hr/>\n"; // pour délimiter l'historique
}else { // début de session
  $historique=array(); // init tableau de tableau
  $_SESSION['historique']=serialize($historique);
}

if ($_POST['mult']){ // nouvelle mult
  echo "Mult. courante : {$_POST['x']} * {$_POST['y']} = ".$_POST['x']*$_POST['y'].
    " !<br/>";
  $tab=array('x'=>$_POST['x'], 'y'=>$_POST['y'], 'r'=>$_POST['x']*$_POST['y']);
  $historique[]=$tab; // ajout dans l'historique
  $_SESSION['historique']=serialize($historique);
  echo "<hr/> Nouvelle Multiplication :<br/>";
}
?>
<form action="<?php echo "{$_SERVER['PHP_SELF']}".(strlen(SID)?'?'.SID:''); ?>"
      method="post">
X<input type="text" name="x" size="10"><br>
Y<input type="text" name="y" size="10"><br>
<input type="submit" value="Multiplier !" name="mult">
</form>
```

```
</body></html>
```

La page affichée ressemblera à :

```
Multiplication et session
2 * 3 = 6
4 * 5 = 20
-----------------------------
Mult. courante : 6 * 7 = 42 !
-----------------------------
Nouvelle Multiplication :
X ____
Y ____
Multiplier!
```

Avec cookies non acceptés, voici le contenu de la barre d'adresse :
`.../multsession.php?PHPSESSID=1d5192e149789a9a52b10f948bbf6843`

3.7 Bases de données

Les fonctions PHP d'accès aux bd sont nombreuses. Il existe des fonctions spécialisées natives pour des sgbds tels que MySQL ou Oracle ou PostgreSQL, mais aussi des fonctions ODBC qui sont plus indépendantes du sgbd cible.

Enfin, il existe des librairies permettant d'abstraire le SGBD cible (Creole, PDO, ...). Cette interface d'abstraction à l'accès de données signifie que vous utilisez les mêmes fonctions pour exécuter des requêtes ou récupérer les données quelque soit la base de données utilisée.

3.7.1 MySQL

Fonctions spécifiques à MySQL :

int mysql_connect('hostname','username','password') `or die('Connexion au serveur impossible !')` ; ret un descripteur de connexion `$co`. La connexion se termine en fin de script.

int mysql_pconnect('hostname','username','password') connexion **persistante** : si une connexion persist. existe déjà avec le même nom et passwd, la réutilise ; ne ferme pas la connexion en fin de script. `$co`

int mysql_select_db('mabase',$co) `or die('Base de données inaccessible !')` ; ret vrai si sélectionné (`use`)

int mysql_query('select|insert| ...',$co) `or die('Requête impossible')` ; retourne faux si impossible ; retourne un entier résultat (`$res`) si requête select ; retour inutile si requête action

int mysql_num_fields($res) ret nb de colonnes du résultat d'un select

string mysql_field_name($res,$i) ret le nom de la colonne d'indice i

string mysql_field_type($res,$i) ret le type. (`mysql_field_len`, `flags` existent également)

int mysql_num_rows($res) ret nb de lignes

array mysql_fetch_assoc($res) ret une ligne sous forme d'asso. indicé par les noms de colonnes

array mysql_fetch_row($res) ret une ligne sous forme de tableau indicé de 0 à mysql_num_fields - 1

int mysql_affected_rows($co) ret le nombre de lignes affectées par la dernière requête `DELETE, INSERT, REPLACE, UPDATE`

int mysql_db_query ('base2','select ...',$co) sélectionne une bd et pose une requête

string mysql_error($co) ret la chaîne indiquant l'erreur MySQL

int mysql_close($co) ferme la connexion (**INDISPENSABLE**)

Un exemple avec MySQL

```
$co=mysql_connect('sql.free.fr','monnom',
  'monpasswd') or die('Connexion au serveur
  impossible !');
mysql_select_db('mabase',$co) or die("Base de
  données inaccessible !");
$r=mysql_query('select nom, prenom from matable',
```

```
  $co) or die('Requête impossible');
$nc=mysql_num_fields($r); $nl=mysql_num_rows($r);
while ($ligne=mysql_fetch_assoc($r)){
  echo "NOM : ",$ligne['nom'],"; PRENOM : ",
  $ligne['prenom'],"<BR>\n";
  }
mysql_close($co);
```

phpMyAdmin

phpMyAdmin est un outil d'administration d'une BD MySQL écrit en PHP. Cet outil est accessible via le web et permet donc d'administrer à distance une BD. On peut : créer, supprimer, modifier (alter) des tables, interroger, ajouter, supprimer, modifier des lignes, importer des fichiers textes contenant les données ... Le site de référence où l'on peut tester en direct est : `http://www.phpmyadmin.net/home_page/index.php`.

3.8 Cookies

Les cookies sont stockées du côté client par le navigateur et contiennent des variables. Quand celui-ci envoie une requête à une URI d'un domaine et d'un chemin dont il possède un cookie, toutes les variables du cookie sont envoyées au serveur. Un cookie a une durée de vie (expire) par défaut égale à la durée de vie de la session. Pour accepter/afficher les cookies sur `Firefox`, utiliser le menu Outlis, Option, Vie Privée, Accepter/Afficher les cookies.

int setcookie('x','Hello !',int expire, string path, `string domain, int secure`**)** doit être exécutée avant toute chose et permet de positionner la variable $x avec la valeur 'Hello !' ; attention le cookie ne sera renvoyé au serveur qu'à la prochaine requête ;

$_COOKIE['x'] accès à une variable de cookie ;

setcookie('x','',time()-1000) efface la variable x du cookie ;

setcookie('y',$value,time()+3600) ; expire dans une heure ;

setcookie('x','345') ; expire en fin de session c'est-à-dire lorsque le navigateur sera fermé (expire=0) ;

3.9 Cookies et session

Les variables de session sont conservées côté serveur. Du côté client, si les cookies sont acceptées, une variable de cookie, généralement nommée `PHPSESSID` est présente. Elle contient un identifiant qui est envoyé au serveur à chaque requête ce qui permet de l'associer à la session correspondante. La variable de configuration `session.cookie_lifetime` spécifie la durée de vie du cookie en secondes. Par défaut, la valeur de 0 signifie : "Jusqu'à ce que le navigateur soit éteint". Cependant, une autre variable de configuration `session.gc_maxlifetime` qui vaut généralement 1440 soit 24 minutes, indique le temps maximum dont jouissent les données de session sur le serveur. Par conséquent, au bout de 24 minutes d'INACTION, les données de session seront supprimées même si le cookie de session perdure côté navigateur.

3.10 Authentification

L'authentification selon le mode courant (login, password) peut être réalisé de bien des façons différentes sur le web. Dans la plupart des cas, on peut réaliser l'authentification par un formulaire HTML classique puis interrogation d'une base de données des utilisateurs contenant les couples (login, password **crypté**). Une fois authentifié, il suffira de conserver dans le tableau de session les informations concernant l'utilisateur loggé (id, nom, type, ...) : `$_SESSION['uid']`, ...

Cependant, il faut également connaître les autres types d'authentification !

3.10.1 Authentification HTTP

La procédure d'authentification HTTP est associée à un **nom de domaine** (realm ou AuthName) et à un répertoire. Elle peut être déclenchée :

- soit par Apache, indépendamment de PHP ;
- soit en utilisant PHP.

Authentification HTTP via Apache

L'authentification est valable pour toute une **arborescence**. Un fichier `.htaccess` spécifie les règles d'authentification valables pour ce répertoire et tous ses **descendants** :

```
AuthType Basic
AuthUserFile "/auto/.../AuthApache/.htpasswd"
AuthName "Le Domaine Privé"
<Limit GET POST>
 require valid-user
</Limit>
```

Le fichier `.htpasswd` contient la liste des utilisateurs et leurs mots de passe cryptés. Il est obtenu grâce à l'utilitaire `htpasswd` fourni par Apache. Par exemple : `htpasswd -c .htpasswd un` ; crée un fichier avec l'utilisateur `un`.

Lors de tout accès à un fichier descendant de `AuthApache`, Apache va envoyer un en-tête au navigateur client qui va afficher une fenêtre popup d'authentification. Par la suite, l'utilisateur reste authentifié pour "Le Domaine Privé" jusqu'à la fin du navigateur ou si une nouvelle authentification PHP est lancée.

Authentification HTTP via PHP

PHP doit être un module d'Apache. On utilise alors la fonction `header` pour demander une authentification ("WWW-authenticate") au client, générant ainsi l'apparition d'une fenêtre de demande d'utilisateur et de mot de passe. Une fois que les champs ont été remplis, l'URL sera de nouveau appelée, avec les variables `$_SERVER['PHP_AUTH_USER']`, `$_SERVER['PHP_AUTH_PW']` et `$_SERVER['PHP_AUTH_TYPE']` contenant respectivement le nom d'utilisateur, le mot de passe et le type d'authentification. Actuellement, seule l'authentification de type "Basic" est supportée. Si l'authentification est réalisée via Apache (`.htaccess`), la variable `$_SERVER['REMOTE_USER']` est égale à `$_SERVER['PHP_AUTH_USER']`.

Exemple 10 (Exemple d'authentification HTTP par PHP)

```
<?php
if (!isset($_SERVER['PHP_AUTH_USER']) || !verifierAuth())
  header("WWW-Authenticate: Basic realm=\"Le Domaine Privé\"");
  header("HTTP/1.0 401 Unauthorized");
  echo "Texte à envoyer si le client annule\n";
  exit;
  } else {
  echo "Bonjour ",$_SERVER['PHP_AUTH_USER'];
  // suite de la page privée
  }
?>
```

La fonction booléenne `verifierAuth()` utilisera tout moyen nécessaire à la vérification du nom et du mot de passe (Base de données, ...). Remarquons que les variables `$_SERVER['PHP_AUTH_...']` sont utilisables même si l'authentification n'a pas été effectuée par le module PHP.

Les variables d'authentification PHP sont valables pour tous les fichiers descendants du répertoire. Par contre, tout fichier html ou php ne testant pas l'authentification est accessible, contrairement à l'authentification par .htaccess (Apache) qui sécurise tout le répertoire.

Désauthentification PHP

Le navigateur écrase le cache d'authentification client d'un domaine quand il reçoit une nouvelle demande d'authentification. Cela permet de déconnecter un utilisateur, pour le forcer à entrer un nouveau nom et son mot de passe. Si l'utilisateur annule l'authentification, il est alors désauthentifié ! Penser à recharger la page. Certains programmeurs changent dynamiquement le nom de domaine pour donner un délai d'expiration, ou alors, fournissent un bouton de réauthentification.

3.11 Téléchargement

- du site web vers le client : download ou déchargement ;
- du client vers le site web : upload ou chargement ;

3.11.1 Déchargement

Réaliser un lien référençant le fichier à décharger, par exemple : Cliquer ici. Si le type du fichier est affichable par le navigateur, il sera affiché (donc déchargé), sinon il sera proposé à l'utilisateur soit de sauver le fichier, soit de lancer l'application associée. Remarquons que tout lien peut être enregistré en utilisant le bouton droit de la souris sur le lien.

3.11.2 Chargement

Réaliser un formulaire de chargement envoyant une requête POST. L'action effectuée (script) après soumission doit vérifier que le fichier chargé est du bon type et de la bonne taille puis l'afficher depuis la zone temporaire (tmp/) ou il est chargé. Le fichier temporaire sera **automatiquement effacé** de la zone à la fin du script, s'il n'a pas été déplacé ou renommé. Par exemple, le formulaire sera :

```
<FORM ENCTYPE="multipart/form-data" ACTION="traite
ment.php" METHOD="POST">
  <INPUT TYPE="hidden" NAME="MAX_FILE_SIZE" value=
"1000">
  Envoyez ce fichier : <INPUT NAME="fichier" TYPE=
"file" SIZE=15>
  <INPUT TYPE="submit" VALUE="Envoyer le fichier">
</FORM>
```

Dans le script `traitement.php` on a accès à différentes variables (différent selon les versions de PHP) :
- `$_FILES['fichier']['name']` le nom du fichier original chez le client ;
- `$_FILES['fichier']['type']` le type mime du fichier "image/gif, text/plain, ..." ;
- `$_FILES['fichier']['size']` la taile du fichier chargé ;
- `$_FILES['fichier']['tmp_name']` le nom du fichier temporaire sur le serveur ;
- `$_FILES['fichier']['error']` le code d'erreur (PHP 4.2.0).

Remarquons que tous les navigateurs ne vérifient pas le `MAX_FILE_SIZE`.

3.12 Développement et débogage

Certaines constantes peuvent être utiles à des fins de débogage :

__FILE__ nom du fichier exécuté ;

__LINE__ numéro de ligne courante ;

De plus, on peut fixer un niveau de rapport exigeant à l'interprète afin de vérifier tous les avertissements et erreurs possibles : `error_reporting(E_ALL);`

Visualiser le source HTML sur le navigateur est très souvent utile notament pour la gestion des caractères spéciaux.

La console Javascript ou console d'erreurs permet également de visualiser les problèmes locaux aux navigateur : javascript et feuille de style.

Chapitre 4

Architecture de site

4.1 Introduction

Ce chapitre est destiné à indiquer la ligne de conduite à tenir lors de la création d'un site Web.

4.2 Analyse des besoins

Lors de la création d'un nouveau site Web ou de le mise à jour d'un site Internet existant, il faut commencer par analyser les besoins :

- Quelle est la cible du site ?
- Choix du vocabulaire exact de la cible, sans ambiguités ni imprécisions.
- Choix des fonctionnalités à intégrer au site.
- Choix des contenus : Quels sont ceux qui seront les plus opportuns ?
- Choix d'une communication (image de marque, charte graphique, argumentaire, ...).
- Structuration du contenu et de la navigation. Ergonomie du site web.
- Règles de rédaction du contenu ;
- Comment améliorer le référencement.
- Rédaction du cahier des charges.

4.3 Cahier des charges

4.3.1 Les exigences

Contexte décrire les caractéristiques fondamentales de votre groupe ou organisme (histoire, buts, principes fondamentaux, fonctionnement, budget, etc) ;

Type du site définir ici ce que vous attendez de votre site. S'agira t'il d'un site vitrine, d'un site outil ? vous placez vous dans une logique de communication (site-vitrine) ou d'information (site-service) ? souhaitez vous un site de vente ou un site d'informations ?

Rubriques il s'agit de l'architecture de votre information, des grandes rubriques et sous rubriques qui structureront le contenu du site.

Description des fonctionnalités Voulez vous un forum, un agenda, des listes de discussion, des base de données consultables en ligne ? tous ces exemples sont autant de fonctionnalités types d'un site Internet, que vous aurez à définir au préalable.

La cible préciser ici le public que vous souhaitez atteindre. L'aspect général du site, son graphisme et ses animations en dépendront.

L'image quelle image souhaitez vous donner de votre groupe, association ou organisme ? Les objectifs du site : il peut s'agir de la constitution d'une base de donnée, de la diffusion d'une information particulière ou encore de la mise en réseau de compétences propres à votre groupe.

Périodicité et types de mises à jour définir ici la façon dont les contenus futurs devront être mis en ligne : Souhaitez vous gérer en interne ou en externe les mises en ligne d'informations, de photos ou des autres contenus ?

Bénéfice attendu décrire en quelques mots le bénéfice que vous attendez de ce site. Il sera aussi important de définir les personnes les plus concernées par ce bénéfice attendu.

4.3.2 Le calendrier

La dead line la description du projet pourra fixer une date clé pour la mise en ligne et la remise des sources. Prévoir les modalités des renvois d'informations et la description des navettes et des validations.

4.3.3 Les ressources

Ressources humaines en externe, désigner une personne chargée de la coordination pour la réalisation et le suivi de votre site. Ce correspondant aura autorité pour prendre ou relayer les décisions vers le prestataire, et saura assurer les échanges mutuels. En interne, désigner un chef de projet et une équipe ayant les compétences dans les différents domaines (BD, prog, graphisme, ...).

Budget Plusieurs éléments sont à prévoir : conception, réalisation, suivi ; formation, embauche ; achat de matériel, de logiciel ou de livres ; coûts de connexion et d'hébergement...

Contenus vous créez un site pour y apporter du contenu. Vous pouvez dès à présent penser aux différentes photos, textes ou modules que vous placerez.

Impératifs techniques Matériel informatique disponible en interne ; Types de connexions et pratique de l'Internet. Matériel prévu spécialement pour le site. Hébergement interne ou externe ?

4.3.4 Maintenance

Un site Internet engendre des coûts annuels, liés essentiellement à sa connexion au réseau, à l'hébergement des pages chez le fournisseur d'accès et au maintien du nom de domaine. A ces frais fixes s'ajoutent ceux concernant la mise à jour régulière des données, les réponses rapides aux demandes qui vous parviennent et les actions de promotion de votre site.

4.4 Architecture

4.4.1 Arborescence du site

Définir clairement l'arborescence du site. Choisissez de concevoir plusieurs pages, avec de nombreuses sous thématiques, plutôt d'une page à contenu multiple trop longue et difficile à consulter. Faire un grand nombre de pages n'est pas un vice rédhibitoire, bien au contraire. Essayez de diviser votre site en thématique et sous thématique pour gagner en lisibilité et en clarté.

L'arborescence de votre site Web doit ressembler à un arbre généalogique avec la page d'accueil en haut et les différentes rubriques qui se répartissent en dessous en suivant une logique de navigation. Pour le moment, il est simplement question d'un plan du site succinct pour se donner une base de départ cohérente. Un bon début est un site de 10-15 pages clair et bien structuré. De toutes façons, votre site se doit d'être vivant et si vous vous impliquez correctement dans le projet, le nombre de pages ira vite en croissant. Ne soyez pas trop ambitieux au départ pour privilégier de solides fondations à votre site plutôt qu'un site mis en ligne à moitié achevé.

Attention quand même à ne pas tomber dans l'effet inverse qui est de créer trop de pages en diluant l'information sans raison. Une règle de base à adopter est celle des **3 clics** : il faut que l'internaute puisse accéder à l'information qu'il cherche en 3 clics de souris.

4.4.2 Soignez l'accueil

Votre page d'accueil est le carrefour de vos visites. Il faut la soigner pour faire une page attractive et structurée. Identifiez posément le thème de votre site et définissez clairement vos différentes rubriques. **Evitez** la page d'accueil constituée d'une animation flamboyante qui fait **patienter** l'internaute et engendre un clic de plus pour rien. Il vaut mieux aller directement aux faits en donnant un accès direct aux rubriques et sous rubriques du site dans une page plaisante à visualiser et bien structurée. De plus, évitez aussi les longs discours que vous garderez pour vos sous rubriques.

L'internaute est toujours pressé, alors ne risquez pas qu'il fuit votre site en alourdissant votre page d'accueil. Pour un chargement décent de votre page d'accueil en connexion bas débit, il faut qu'elle tombe en dessous de 40-50Ko, images comprises.

4.4.3 Menu clair et accessible

Votre menu va évoluer dans le temps, mais il faut le concevoir avec soin dès le départ. Définissez des effets de couleurs ou de soulignage lorsque l'internaute passe avec sa souris sur les liens. De plus, privilégiez les liens texte aux liens image ou zone réactive pour optimiser le référencement par les moteurs de recherche.

L'idéal serait d'avoir les différentes rubriques et sous rubriques accessibles depuis toutes les pages du site. L'internaute et le moteur de recherche apprécieront l'attention.

4.4.4 Plan du site

Il est intéressant à plusieurs titres d'intégrer un plan de votre site accessible depuis toutes les pages du site. D'une part, un internaute perdu pourra facilement retrouver son chemin et d'autre part le robot de crawl des moteurs de recherche aura plus de facilité à trouver les pages enterrées sous plusieurs niveaux de sous rubriques.

4.4.5 Intégrer un moteur de recherche

Si on garde à l'esprit que le but d'un site Web est de procurer à l'internaute l'information qu'il recherche dans les meilleures conditions et le plus rapidement possible, il est intéressant d'adjoindre un moteur de recherche à votre site. Plusieurs solutions s'offrent à vous, mais elles requièrent des connaissances en langage de programmation comme le PHP ou l'ASP. Si votre site est correctement référencé sur Google, c'est à dire que toutes les pages du site sont présentes dans son index, vous pouvez utiliser le moteur de recherche Google adaptable à votre site (pour savoir si toutes les pages de votre site Internet sont présentes dans Google, tapez la commande site :www.monsite.com dans Google).

4.4.6 Communiquez avec vos visiteurs

Toujours dans l'optique de fidéliser les internautes qui visitent votre site, il est important d'inclure un moyen de vous contacter. Au minimum, il faut que l'internaute puisse vous envoyer un eMail. Pour cela, faites un lien vers votre eMail, accessible depuis toutes les pages du site. Il s'agit d'un hyperlien de type mailto, que tous les éditeurs HTML sauront faire pour vous.

Pour aller plus loin, vous pouvez créer un formulaire de contact qui semble être un support plus convivial.

Parmi les autres moyens qui existent pour fidéliser les internautes, il faut citer la newsletter, les forums de discussion, les livres d'or, les Weblogs et les Chats.

4.4.7 Mentions légales

Désormais, la Commission Informatique et Libertés (CNIL) dicte à tous les sites français de s'identifier clairement. Il faut donc indiquer, sur la page d'accueil ou sur une page spéciale accessible depuis toutes les pages du site, certaines mentions obligatoires comme le propriétaire du site, la personne à contacter en cas de besoin de rectification sur les données personnelles et aussi l'hébergeur du site.

De plus, il faut déclarer son site Internet à la CNIL qui se chargera de vous attribuer un numéro.

En tout cas, il est utile de présenter une certaine transparence par rapport à votre site, augmentant ainsi le capital confiance que l'internaute attribuera à votre site Web.

4.4.8 Faites des liens

Un des composants essentiels à la réussite d'un site sur le Web est son référencement. Google et d'autres moteurs de recherche attribuent une attention toute particulière aux liens hypertexte qui renvoient d'un site à un autre. Les sites qui reçoivent beaucoup de liens entrants sont jugés comme populaires par les moteurs de recherche et ils sont ainsi favorisés dans les résultats de recherche.

Il est ainsi intéressant de demander des échanges de liens avec d'autres sites Web, de préférence à thèmes similaires ou complémentaires. Prévoyez donc une page d'échanges de liens pour placer les URLs des sites qui sont d'accord pour parler de vous, à condition que vous parliez d'eux.

4.5 La charte graphique

L'aspect visuel du site devra être homogène. Pour cela, on rédige une charte graphique, c'est à dire qu'on définit une mise en page à respecter sur l'ensemble du site. La cohérence graphique doit être respectée quels que soient les différents intervenants de la production (graphiste, directeur artistique, programmeur, ...).

Pour l'aspect graphique, on peut s'inspirer d'autres sites ou des modèles fournis par les logiciels de création de sites web, ainsi que de la mise en page de certains magasines ou d'affiches publicitaires.

Quelques règles de mise en page sont à respecter :

- L'oeil parcourt les pages en diagonal, du coin supérieur gauche au coin inférieur droit. Les éléments importants de la page sont à placer sur cet axe.
- On évitera que l'internaute ait à utiliser sa barre de défilement horizontale. L'affichage le plus répandu étant le 800x600 et 1024x768.
- En ce qui concerne la longueur de la page, il est recommandé de ne pas dépasser 3 fois la hauteur d'écran. Les informations importantes sont à placer en haut des pages : certains internautes n'utilisent pas le défilement vertical !
- Pour faciliter la lecture, les lignes ne doivent pas contenir plus d'une dizaine de mots, sinon l'oeil perd le fil de la lecture. Cela conduit souvent à utiliser des colonnes pour les textes.
- Pour des raisons esthétiques, se limiter à 3 polices. Préférer les polices sans empattement qui sont plus lisibles sur un écran. Arial et Verdana sont des bons choix. N'utiliser que des polices courantes qui seront disponibles chez l'internaute. Si une police exotique doit être employée (pour un titre par exemple) le mieux est d'utiliser un logiciel de retouche d'images pour transformer le texte en une image que l'on insérera dans la page.
- Le choix des couleurs : il existe des règles universelles régissant l'harmonie des couleurs, dûes à des propriétés physiques de l'oeil. En effet, lorsqu'il contemple une couleur, l'oeil crée automatiquement, sur son contour, un filtre de la couleur complémentaire. On parle de "contraste simultané". Avec ce mécanisme, la perception des couleurs dépend des couleurs avoisinantes. Ainsi, le jaune paraîtra plus orangé lorsqu'il est associé à du bleu et le bleu paraîtra plus violet. Un bleu à côté d'un rouge apparaîtra vert, etc.

  Par ailleurs, des couleurs voisines sur le diagramme chromatique créent une sensation d'équilibre pour l'oeil, en vertu de l'absence de contraste, on parle ainsi "d'harmonie d'analogie". Il existe donc globalement deux façons principales de choisir des couleurs harmonieuses :
  - en choisissant des nuances d'une même couleur, soit des couleurs de même teinte dont les tons sont proches ;
  - en mêlant des couleurs complémentaires (chaudes et froides), c'est-à-dire des couleurs éloignées sur le diagramme chromatique. Pour deux couleurs, il suffit de choisir des couleurs complémentaires, diamétralement opposées ; pour trois couleurs, les couleurs choisies doivent former un triangle équilatéral, etc.

Chapitre 5

Les paquets PEAR

5.1 Introduction

Pear (PHP Extension and Application Repository) est une bibliothèque contenant de nombreux paquets PHP déjà développés. Son adresse web est : `http://pear.php.net/`, voir [6]. C'est un projet communautaire destiné à fournir :

- une librairie structurée de code "open-source" en PHP ;
- un système de distribution par paquets et une maintenance ;
- un style standard de programmation ;
- PHP Extension Community Library (PECL), des extensions écrites en C et compilées ;
- un site Web, une liste de diffusion, ...

5.2 Installation

Pear nécessite que php fonctionne ! Or php peut fonctionner en ligne de commande sur une machine que vous administrez ou pas, mais également en hébergement mutualisé en tant que module Apache. Le gestionnaire de paquets fonctionnera donc :

- soit en ligne de commande (Command Line Interpreter CLI) ;
- soit via une intervace Web ;

5.2.1 Installation en ligne de commande sous Windows

Il faut utiliser l'installateur PEAR minimum appelé go-pear. Par exemple, avec wamp (Apache, MySQL, PHP pour Windows) (voir [7]) installé en local, le répertoire `C:\wamp\php` contient l'interprète php (CLI) mais également un fichier `go-pear.bat` :

- `cd C:\wamp\php;`
- `go-pear.bat`
- répondre aux questions posées en installant PEAR par exemple dans le répertoire courant `C:\wamp\php`
- pour installer un paquetage : `pear.bat install Log`

Remarques :

- le gestionnaire de paquet est donc `pear.bat`
- `go-pear.bat` modifie le fichier `php.ini` du CLI de `C:\wamp\php` mais il faut également modifier le fichier `php.ini` du module php d'Apache `C:\wamp\Apache2\bin\php.ini` en modifiant le chemin d'inclusion : `include_path=".;c:\wamp\php\pear"`
- si l'installation de certains gros paquetages génère une erreur : `Fatal error : Allowed memory size of 8388608 bytes exhausted`, il faut modifier `C:\wamp\php\php.ini` en modifiant la taille maxi d'un pus php : `memory_limit = 12M`.

5.2.2 Installation en hébergement mutualisé

Sans ligne de commande ou sans droits suffisants pour modifier la configuration du serveur et du module PHP, on utilise un gestionnaire de paquets écrit en php qui est exécuté par le module php du serveur HTTP de la façon suivante :

- récupérer la page située à l'adresse `http://pear.php.net/go-pear` dans le fichier local `go-pear.php` ;
- par ftp, sauver ce fichier dans un répertoire sous votre racine Web, par exemple `~/public_html/Pear`.
- accéder à l'adresse Web `http://monserveur/~jdupont/Pear/go-pear.php`

- Un formulaire apparaît où l'on peut noter que le `$php_dir` sera `~/public_html/Pear/PEAR`
- Ensuite l'adresse `http://monserveur/~jdupont/Pear/index.php` est l'adresse qui devra être utilisée pour gérer (installer, désinstaller, ...) des paquetages, ...
- Pour installer le paquet Log, il suffit de saisir "Log" dans la zone de texte `Quick-install a package` de l'onglet `Package Management`.
- Il ne faudra pas oublier de rajouter `/auto/jdupont/public_html/Pear/PEAR` dans la configuration php des inclusions des scripts utilisant des paquets PEAR. Comme on n'a pas accès au fichier php.ini, ceci peut se réaliser par l'appel suivant en début de chaque script :
  `set_include_path("/auto/jdupont/public_html/Pear/PEAR".PATH_SEPARATOR.get_include_path());`

5.3 Canaux PEAR

Les canaux PEAR depuis la version 1.4 permettent d'installer des paquets depuis différentes sources (pear.php.net, mais également d'autres). Pour installer un paquet :`./pear.bat install channel/Packagename`. Mais avant, il faut indiquer l'adresse de la source de paquet :

- `pear channel-discover pear.phpdb.org`
- puis `pear config-set preferred_state beta`
- puis `pear install phpdb/propel_generator`

La seconde ligne indique la version la plus récente du paquet.
En hébergement mutualisé, il faut :

- aller dans l'onglet "Channel Management" et saisir l'adresse `pear.phpdb.org` dans la zone de texte "Add a new channel server :"
- aller dans l'onglet "Configuration" et saisir la valeur `beta` dans la zone de texte "Preferred Package State :"
- saisir `phpdb/propel_generator` dans la zone de texte `Quick-install a package` de l'onglet `Package Management`
- si l'installation nécessite d'autres paquets alors il faut "découvrir" le canal correspondant "pear.phing.info" puis installer `phing/phing`

5.4 Sécurisation en hébergement mutualisé

Le gestionnaire de paquets Web doit être protégé des accès extérieurs par exemple avec un fichier `.htaccess` sous Apache :

- `htpasswd -c passwd admin` crée un fichier nommé passwd contenant l'utilisateur admin et son mot de passe crypté ; ce dernier est demandé interactivement. `htpasswd` est une commande fournie par Apache dans : `C:\wamp\Apache2\bin` ;
- créer un fichier `.htaccess` contenant :

```
AuthType Basic
AuthUserFile "/auto/jdupont/public_html/Pear/passwd"
AuthName "Domaine Pear"
<Limit GET POST>
 require valid-user
</Limit>
```

- télécharger ces 2 fichiers dans le répertoire `~/public_html/Pear`

5.5 Utilisation de paquet PEAR

Pour utiliser un paquet PEAR, Il suffit :

- d'ajouter le chemin dans la configuration d'inclusion ; (pas nécessaire si déjà fait dans `php.ini`)
- d'inclure le ou les fichiers du paquet PEAR ;
- de les utiliser selon la documentation en ligne ;

5.5.1 Formulaire HTML

Il existe de nombreux paquets PEAR permettant de générer du HTML. `HTML_Form` est très simple d'utilisation. Tout d'abord, l'installation :

```
pear.bat install HTML_Form
```

Puis on écrit un script utilisant `HTML_Form` :

Exemple 11 (exempleHTMLForm.php)
```
<?php
require_once "HTML/Form.php";

$form = new HTML_Form($_SERVER['PHP_SELF']);

$form->addText("nom", "Votre Nom :");
$form->addText("email", "Votre adresse email :");
$form->addPasswordOne("password", "Votre mot de passe :");
$form->addPlaintext("Attention", "Vérifiez vos champs avant de valider !");
$form->addSubmit("valider", "Valider");

$form->display();
?>
```

L'affichage sera réalisé dans un tableau à deux colonnes dont celle de gauche sera alignée à droite et celle de droite alignée à gauche.

5.5.2 Ajax et HTML

On va montrer l'utilisation du paquet `HTML_Ajax` qui permet d'utiliser Ajax dans des pages HTML.
Tout d'abord, l'installation :

```
pear.bat install HTML_Ajax
```

Puis on écrit des script utilisant `HTML_Ajax`. Tout d'abord pour gérer les requêtes Ajax lancées depuis du javascript, il faut définir le script `server.php` :

Exemple 12 (server.php)
```
<?php
session_start();
//une session est nécessaire

include 'HTML/AJAX/Server.php';

$server = new HTML_AJAX_Server();
$server->handleRequest();
?>
```

Ensuite, il faut écrire la page HTML qui contient le script Javascript de raffraîchissement :

Exemple 13 (exempleAjax.html)
```
<html>
<script type="text/javascript" src="server.php?client=all"></script>

<div id="cible">Element remplacé par Ajax</div>
<script type="text/javascript">
  HTML_AJAX.replace('cible','output.php');
</script> <!-- Pour la première fois -->
<form> <!-- pour les raffraîchissements par bouton -->
  <input type="button" onclick="HTML_AJAX.replace('cible','output.php');"
  value="Mettre la cible à jour !">
</form>
</html>
```

Cette page récupère, par `server.php`, une bibliothèque Javascript qui gère `XMLHttpRequest`. Puis elle remplace la cible par le contenu généré par `output.php`. Enfin, elle affiche un bouton permettant d'actionner la mise à jour de la cible et SEULEMENT de la cible : la page n'est pas rechargée !

Exemple 14 (output.php)

```
<?php
print("<h1>". strftime("%H:%M:%S") ."</h1>");
?>
```

Chapitre 6

Creole et Propel

6.1 Introduction

Ces paquets Pear permettent d'abstraire la couche données d'une application en permettant la manipulation des données sous forme d'objets PHP. Leurs adresses web sont :

- `http://creole.phpdb.org/trac/`, voir [8]
- `http://propel.phpdb.org/trac/`, voir [9].

6.2 Creole

Le projet Creole est un projet communautaire destiné à fournir une couche d'abstraction de différents SGBD (Oracle, PostgreSQL, MySQL, ...) en une interface orientée objet homogène. L'objectif principal de cette couche est de permettre la portabilité du code indépendamment de l'implémentation des données. L'intérêt de cette portabilité réside dans la migration possible d'un SGBD à un autre sans une réécriture complète de l'application. L'exemple de JDBC (abstracion de SGBD pour Java) a inspiré le développemment de Creole.

Il existe de nombreuses autres couche d'abstraction de SGBD (PEAR : :DB, PEAR : :MDB, ADOdb, PDO). Creole est un sous-projet de Propel.

Ses fonctionnalités sont les suivantes :

- PHP5 : nouveau modèle objet, exceptions, ...
- permet la création de pilote spécifique à une utilisation de SGBD bien que Creole propose des pilotes pour la plupart des SGBD ;
- API orientée objet ;
- gestion itérative des résutats de requête avec déplacement relatif ou absolu ;
- métadonnées (structure de la BD) accessible par API orientée objet ;
- système de types de données unifié (basé sur JDBC) ;
- gestion des BLOB et des CLOB ;
- méthodes spécifiques aux types de données afin d'assurer la conversion ;
- gestion de l'auto-incrémentation et des séquences ;

6.2.1 Installation

C'est un paquet Pear appartenant au canal phpdb. Son installation en ligne de commande se résume donc à : `pear.bat install phpdb/creole`

En hébergement mutualisé, il suffira d'utiliser le gestionnaire de paquets web.

6.2.2 Utilisation de Creole

Connection et consultation

Un DSN (Data Source Name) est une référence à une source de données (la plupart du temps, une BD) qui peut être formatté en tableau ou en chaîne PHP. Cette référence doit contenir :

- le type du SGBD (mysql) ;
- l'hôte (localhost ou mysql.free.fr ou ...) ;
- le nom d'utilisateur ;
- son mot de passe (optionel) ;

– le nom de la BD ;

Une fois défini le DSN, il suffit de se connecter à la base et d'y exécuter une requête de consultation qui retournera une ressource contenant l'ensemble des résultats (ResultSet). Il suffira ensuite d'itérer sur ce résultat afin de d'opérer un calcul ou un affichage comme dans l'exemple suivant :

Exemple 15 (creole1.php)

```
0   <html><head><title>essai Creole 1</title></head><body>
1   <h1>essai Creole 1</h1>
    <?php
3   set_include_path("C:\wamp\php\PEAR" . PATH_SEPARATOR . get_include_path());
    include_once "creole/Creole.php";
    $dsn = "mysql://meynard:toto@localhost/meynardprojetinfol3"; // string ou array
6   // $dsn = array('phptype' => 'mysql', 'hostspec' => 'localhost',
    // 'username' => 'guest', 'password' => null, 'database' => 'meynardprojetinfol3');
    $co = Creole::getConnection($dsn);
9   $rs = $co->executeQuery("SELECT id, titre  FROM projet");
    while($rs->next()) {
      echo $rs->getInt("id") . " : " .$rs->getString("titre") . "<br/>\n";
12  }
    ?>
    </body></html>
```

Ce script effectuera l'affichage suivant :

```
essai Creole 1
2 : Dessin vectoriel
4 : projet suivant
```

L'intérêt de récupérer les valeurs par des get typés (getString, ou getInt, ...) consiste dans la conversion de la valeur NULL depuis la BD vers la valeur null PHP. Si on utilise un get non typé, on récupère un chaîne de caractères.

Déplacement et itération sur un ResultSet (curseur)

On peut se déplacer dans les lignes retournées :

```
$rs->first();     //va en 1ere ligne
$rs->relative(3); // avance de 3 lignes
$rs->previous();  // précédent
$rs->last();      // dernière ligne
```

On peut également parcourir ligne par ligne à l'aide d'un tableau associatif :

```
foreach($rs as $ligne) {
  foreach($ligne as $cle => $val){
    echo "$cle = $val,";
  }
  echo "<br/>";
}
```

Ce qui donnera le résultat suivant :

```
essai Creole 1
id = 2,titre = Dessin vectoriel,
id = 4,titre = projet suivant,
```

Mise à jour

Pour réaliser une mise à jour (INSERT, UPDATE, DELETE), il suffit d'utiliser la méthode `executeUpdate()` qui retourne le nombre de lignes affectées.

```
$nb = $co->executeUpdate("DELETE FROM projet WHERE id='4'");
```

Requête préparée

Les requêtes préparées (prepared queries ou prepared statement) sont des requêtes incomplètes soumises au SGBD qui prépare un plan d"exécution et que l'on peut complète au moment de la soumission. La performance est améliorée lorsqu'on utilise plusieurs fois la même requête préparée.

Avec Creole, même si le SGBD ne supporte pas celles-ci, Creole les émulera.

```
$stmt = $co->prepareStatement("INSERT INTO users (id, name, created) VALUES (?,?,?)");
$stmt->setInt(1, $id);
$stmt->setString(2, $name);
$stmt->setTimestamp(3, time());
$stmt->executeUpdate();
```

On peut également utiliser un tableau de paramètres :

```
include_once 'Date.php'; // using PEAR Date
$stmt->executeUpdate(array(2, "Myname", new Date(time())));
```

Transactions

Par défaut, le mode transactionnele est en auto-commit, c'est-à-dire que chaque requête constitue une transaction. Pour débuter une transaction il suffit de désactiver le mode auto-commit. Pour valider (commit) ou annuler (rollback) une transaction, il faut utiliser les exceptions Creole comme dans l'exemple suivant :

```
try {
  $conn->setAutoCommit(false); // start a transaction
  $conn->executeUpdate("DELETE FROM mytable WHERE id = 3");
  $conn->executeUpdate("DELETE FROM myothertable WHERE id = 4");
  $conn->executeUpdate("UDPATE stillanothertable SET col = 1 WHERE id = 5");
  $conn->commit(); // commit the transaction
} catch (Exception $e) {
  $conn->rollback(); // abort all delete/update queries in the transaction
  print "Aborted the transaction because: " . $e->getMessage();
}
```

6.3 Propel

Propel est une couche "Object Relational Mapping" (ORM) écrite en PHP permettant de manipuler une BD relationnelle par l'intermédiaire d'objets PHP5. L'adresse web est : `http://pear.php.net/`.

6.3.1 Installation de propel

Propel est constitué de 2 paquets PEAR du canal phpdb :
- `propel_generator` : permettant de générer le projet ;
- `propel_runtime` : nécessaire à l'exécution du projet ;

L'installation en ligne de commande sous windows/cygwin se réalise donc :

```
cd /c/wamp/php/
pear.bat config-set preferred_state beta
pear.bat install phpdb/propel_generator
pear.bat install phpdb/propel_runtime
```

Il est nécessaire d'avoir installé auparavant `phing/phing` et `phpdb/creole`. La commande principale `propel-gen.bat` est située dans le répertoire `/c/wamp/php` qui contient également `php.exe` et `pear.bat`. Par conséquent, la variable `PATH` doit contenir ce répertoire. La mise à jour est réalisée par `pear.bat upgrade ...`

6.3.2 Utilisation du générateur propel

Le générateur propel utilise un **fichier central** `schema.xml` décrivant le schéma de la BD relationnelle : database, table, column, foreign-key, unique, index, ... Ce schéma peut :
- soit être écrit à la main ;
- soit être fabriqué à partir d'une BD existante ;
- soit être initialisé à partir d'une BD existante puis modifié à la main !

Exemple 16 (fichier schema.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<database name="projetinfol3">
  <table name="etudiant">
```

```
    <column name="id" type="INTEGER" required="true" primaryKey="true"
 default="" />
    <column name="ue_id" type="VARCHAR" size="5" required="true" default="" />
    <column name="projet_id" type="INTEGER" />
    <column name="numetu" type="INTEGER" required="true" default="" />
    <column name="dateModifAffect" type="TIMESTAMP" />
    <foreign-key foreignTable="utilisateur" onDelete="cascade"
 onUpdate="cascade" />
      <reference local="id" foreign="id"/>
    </foreign-key>
    <foreign-key foreignTable="ue" onDelete="restrict" onUpdate="cascade">
      <reference local="ue_id" foreign="id"/>
    </foreign-key>
    <foreign-key foreignTable="projet" onDelete="restrict" onUpdate="cascade">
      <reference local="projet_id" foreign="id"/>
    </foreign-key>
    <unique name="numetu">
      <unique-column name="numetu"/>
    </unique>
    <index name="ue_id">
      <index-column name="ue_id"/>
    </index>
    <index name="projet_id">
      <index-column name="projet_id"/>
    </index>
  </table>
...
```

schema.xml partir d'une BD existante

- créer et aller dans le répertoire du projet "repprojet" ;
- créer le fichier `build.properties` contenant :
  `propel.project = ProjetInfo`
  `propel.database = mysql`
  `propel.database.url = mysql://guest@localhost/mabase`
- générer le fichier `schema.xml` par : `propel-gen.bat . creole`

Le schéma est maintenant constitué et décrit la BD fournie dans la propriété propel.database.url.

Configuration d'exécution (runtime)

Avant de générer à partir du schéma, il faut définir la configuration d'exécution dans `runtime-conf.xml` :

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<config>
 <log>
  <ident>ProjetInfo</ident>
  <level>7</level>
 </log>
 <propel>
  <datasources default="mabase">
   <datasource id="mabase">
    <adapter>mysql</adapter>
    <connection>
     <phptype>mysql</phptype>
     <database>mabase</database>
     <hostspec>localhost</hostspec>
     <username>guest</username>
     <!--      <password></password> -->
    </connection>
   </datasource>
```

```
  </datasources>
 </propel>
</config>
```

A noter que la configuration d'exécution peut varier de celle de construction (BD cible, username, ...).

Construire le code SQL et les classes PHP

La commande `propel-gen.bat .` permet de construire :
- le code SQL de génération de la BD qui sera situé dans : `repprojet/build/sql/schema.sql` ;
- le modèle objet du projet situé dans : `repprojet/build/classes/ProjetInfo`
- la configuration d'exécution `build\conf\ProjetInfo`conf.php-

Les classes sont situées dans le répertoire : `/c/wamp/www/repprojet/build/classes/ProjetInfo`

Créer la BD d'exécution

La commande `propel-gen.bat . insert_sql` permet d'exécuter les `CREATE TABLE ...` dans la BD vide qui doit avoir été créée auparavant.

6.3.3 Utilisation à l'exécution de Propel

Include path

Tout d'abord, le module php d'Apache doit connaître l'emplacement des paquets PEAR ainsi que des classes PHP générées. Ceci peut être réalisé de 2 façons :
- soit par modification du fichier `C:\wamp\Apache2\bin\php.ini` avec une ligne `include_path=".;c:\wamp\php\pear"`
- soit par la fonction PHP `set_include_path("C:...\build\classes" . PATH_SEPARATOR . get_include_path());`

On utilisera l'une et/ou l'autre des méthodes selon qu'on a accès ou non à `php.ini` et que PEAR est toujours utilisé !

Initialisation de Propel

```
include_once 'propel/Propel.php';
Propel::init("c:\wamp\www\ProjetInfoL3\build\conf\ProjetInfo-conf.php");
```

Le fichier d'initialisation a été généré par `propel-gen` à partir de la configuration d'exécution au format XML.

Utilisation du modèle objet

Supposons qu'une table utilisateur existe dans la BD. Pour gérer un utilisateur, il suffit :
- d'inclure sa classe par `include_once 'ProjetInfo/Utilisateur.php';`
- La création d'un nouvel utilisateur sera réalisée par `$u=new Utilisateur();`
- On accède à ses propriétés par get/set : `$u->setNom('Dupont');`
- On insère dans la BD par : `$u->save();`
- On recherche dans la BD par sa clé primaire avec : `$u=UtilisateurPeer::retrieveByPK(123);`
- On supprime dans la BD en supprimant l'objet : `$u->delete();`

Bien d'autres possibilités existent notamment pour réaliser des requêtes SQL à partir de critères. Il faut lire la documentation ! Quelques exemples suivent :
- clause ORDER BY : `$c=new Criteria();$c->addAscendingOrderByColumn(ProjetPeer::TUTEUR_ID);`
- récupération d'un tableau d'objets Etudiant à parit du projet où ils sont inscrits (navigation au long d'une contrainte d'intégrité référentielle) : `$pjt->getEtudiants();`
- Transaction Propel :

```
$con = Propel::getConnection(UtilisateurPeer::DATABASE_NAME);
try {
  $con->begin();        // débute transaction
  $u->save($con);       // sauve l'utilisateur
  if ($insc->getType()=='e'){ // si etudiant
      $e=new Etudiant();
      $e->setId($u->getId());
      $e->setNumetu($insc->getNumetu());
      $e->setUeId($insc->getUe());
      $e->save($con);  // sauve etudiant
```

```
    }
    $insc->delete();     // supprime l'inscription
    $con->commit();
  }
  catch (Exception $e) {
    $con->rollback();
  }
```

– requête Creole à travers Propel :

```
$con = Propel::getConnection(EtudiantPeer::DATABASE_NAME);
$sql = "SELECT u.id,u.login,u.nom,u.prenom,u.email,e.numetu,e.projet_id
FROM utilisateur u, etudiant e
WHERE u.id=e.id AND e.ue_id='". $_SESSION['ue'] ."' ORDER BY u.nom, u.prenom";
$stmt = $con->createStatement();
$rs = $stmt->executeQuery($sql);
while($rs->next()) {
  ...
}
```

6.3.4 Erreurs fréquentes

Nom du projet, de la bd, du DSN

En cas de changement de BD et/ou de SGBD et/ou de projet, il faut bien comprendre quelles modifications effectuer. Les noms suivants doivent être **absolument identiques** :

– dans le fichier `schema.xml`, la racine `<database name="toto">` ;
– dans `build.properties`, `propel.project = toto` ;
– dans `runtime-conf.xml`, `<datasources default="toto">` et `<datasource id="meynardprojetinfol3">`

Partage de projet via SVN

En cas de partage du code PHP, on peut très bien attaquer 2 bd différentes à condition :

– qu'elles aient le même nom (toto) et le même type ;
– de changer directement dans `.../build/conf/toto-conf.php` les paramètres d'accès à la bd : hostspec, username, password ;
– ignorer ce fichier ainsi que `runtime-conf.xml` dans les commit SVN ;
– à chaque `propel-gen`, il faudra rétablir le fichier local (on l'aura sauvé préalablement)

makefile

A cause d'une erreur de la version 1.2, si on importe le fichier xml depuis une BD MySQL, il faut convertir certains mots-clés en minuscule !

```
PROPEL-GEN=propel-gen.bat

schema:
    #genere le schema.xml à partir de la bd définie dans build.properties
    $(PROPEL-GEN) . creole
    #convertit majuscules en minuscules
    sed 's/"RESTRICT"/"restrict"/g' schema.xml >tempo
    sed 's/"CASCADE"/"cascade"/g' tempo > tempo2
    sed 's/"SET NULL"/"set null"/g' tempo2 >schema.xml
    rm tempo tempo2
```

Bibliographie

[1] *Programmation HTML et Javascript*, de Chaléat, Charnay, Eyrolles (2000), "simple et rapide pour les bases HTML, CSS, JavaScript, 450 pages"

[2] *Webmaster in a nutshell*, de S. Spainhour, R. Eckstein, O'Reilly (2003), "Manuel de référence rapide pour HTML, CSS, JavaScript, PHP, XML, 550 pages, en anglais"

[3] *Spécification HTML*, `http://www.w3.org/TR/REC-html40/`, W3C (1999), "La spécification complète en anglais"

[4] *Un site de documentation sur les technologies du web*, `http://www.w3schools.com/`, "Très complet et en anglais (XHTML, CSS, JavaScript) "

[5] *Le manuel PHP en français*, `http://fr3.php.net/manual/fr/`, "La référence "

[6] *Site PEAR*, `http://pear.php.net/`, "Le site Web de PEAR"

[7] *Site WAMP*, `http://www.wampserver.com/`, "Le site Web de WAMP (Apache, MySQL, PHP pour Windows)"

[8] *Site Creole*, `http://creole.phpdb.org/trac/`, "Le site Web de Creole, une couche d'abstraction de différentes bases de données relationnelles"

[9] *Site Propel*, `http://propel.phpdb.org/trac/`, "Le site Web de Propel, une couche d'abstraction du relationnel par l'objet"

Annexe A

Solutions des exercices

Solution 1 – multiplication.html

```
<html><head><title>Multiplication</title></head><body>
<h1>Multiplication</h1>
<form action="multiplication.php" method="get">
X<input type="text" name="x" size="10"><br>
Y<input type="text" name="y" size="10"><br>
<input type="submit" value="Multiplier !" name="mult">
</form>
</body></html>
```

– multiplication.php

```
<html><head><title>Multiplication</title></head><body>
<h1>Multiplication</h1>
<?php
  echo "Résultat {$_GET['x']} * {$_GET['y']} = ".$_GET['x']*$_GET['y']." !";
?>
</body></html>
```

Solution 2 mult.php

```
<html><head><title>Multiplication</title></head><body>
<h1>Multiplication</h1>
<?php
if ($_GET['mult']){
  echo "Résultat {$_GET['x']} * {$_GET['y']} = ".$_GET['x']*$_GET['y']." !<br/>";
  echo "<hr/> Nouvelle Multiplication :<br/>";
}
?>
<form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="get">
X<input type="text" name="x" size="10"><br>
Y<input type="text" name="y" size="10"><br>
<input type="submit" value="Multiplier !" name="mult">
</form>
</body></html>
```