République Algérienne Démocratique et Populaire

Ministère de la Formation et de l'enseignement Professionnels

INSTITUT DE FORMATION

PROFESSIONNELLE De Birkhadem

Conception de pages Web Dynamiques

Avec

PHP/Mysql

Fevrier 2005

IFP de Birkhadem,

✉ : Rue des Trois Frères Djillali Birkhadem - Wilaya d'Alger

☎ : 021 54.21.44/45/46 Fax : 021 54.22.00

Site web : www.ifp-alger.edu.dz

e-mail : ifperf@wissal.dz

SOMMAIRE

I.

Introduction

Un site dynamique est une entreprise coûteuse en terme de temps. Les mises à jour régulières sont parfois difficiles à réaliser. Or un site qui n'évolue pas est parfois condamné à disparaître, ou du moins voir sa fréquentation diminuer. L'internaute aime le changement, il ne supporte pas un site statique qui n'évolue guère. Les langages dynamiques permettent de résoudre ces problèmes. Ils facilitent les opérations de mise à jour, permettent plus d'activités sur les pages. Le but de ces pages est de vous initier à l'un de ces langages **PHP**, ainsi que le **Mysql** pour la gestion de la base de données. Pour pouvoir utiliser le PHP convenablement il faut au préalable maîtriser le langage **HTML** et également le langage **javascipt**.

II. er Chapitre
Le langage HTML

II.1. Présentation du HTML

Le HTML ("HyperText Markup Language") est un système qui formalise l'écriture d'un document avec des balises de formatage indiquant la façon dont doit être présenté le document et les liens qu'il établit avec d'autres documents.
Il permet, entre autre, la lecture de documents sur Internet à partir de machines différentes grâce au protocole HTTP, permettant d'accéder via le réseau à des documents repérés par une adresse unique, appelée URL.

II.1.1 Le HTML est un langage à balises

Le HTML n'est pas un langage de programmation, c'est un simple fichier texte contenant des balises permettant de mettre en forme le texte, les images ...
Une balise est une commande (un nom) encadrée par le caractère inférieur (<) et le caractère supérieur (>).

II.1.2 Comment utiliser les balises HTML ?

Les balises HTML peuvent être uniques; la balise
 représente par exemple un retour à la ligne.
Les balises HTMl peuvent également fonctionner par paire afin d'agir sur le texte qu'ils encadrent (la balise de fin est alors précédé d'un /) :

<marqueur> Votre texte formatté </marqueur>

Ainsi les balises et permettent de mettre en gras le texte qu'elles encadrent :
 Ce texte est en gras

Les balises ne sont pas sensibles à la casse, c'est-à-dire qu'on peut les écrire indifféremment en minuscules ou en majuscules.

II.1.3 La page HTML minimum

Une page HTML est un simple fichier texte commençant par la balise <*HTML*> et finissant par la balise <*/HTML*>. Elle contient également un *en-tête* décrivant le titre de la page, puis un *corps* dans lequel se trouve le contenu de la page.
L'en-tête est délimité par les balises <*HEAD*> et <*/HEAD*>
Le corps est délimité par les balises <*BODY*> et <*/BODY*>.
Ainsi la page HTML minimum peut être représentée comme suit :

```
<HTML>

    <HEAD>
        <TITLE> Le titre </TITLE>
    </HEAD>

  <BODY>
        Contenu de la page
    </BODY>

</HTML>
```

II.2. Les balises de structure

II.2.1 Les attributs

Les attributs suivants sont placés dans les balises de structure pour permettre une disposition plus précise des éléments HTML :

Attribut	Valeur	Effet Visuel
ALIGN	LEFT RIGHT CENTER JUSTIFY	LEFT RIGHT CENTER JUSTIFY
ID		Attribue un nom au contenu de la balise (pour faire des liens)
CLASS		Assigne une classe au contenu (pour les feuilles de style)

Les attributs s'utilisent de la manière suivante:
<BALISES ATTRIBUT1=XXXXX ATTRIBUT2=XXXX> Texte </BALISE>
Par exemple:
<H1 ALIGN=LEFT> Texte aligné à gauche </H1>

II.2.2 Les Paragraphes

HTML considère les paragraphes comme des blocs de texte.
La mise en page par blocs de texte est réalisée par l'intermédiaire de la paire de balises et . Cette balise accepte n'importe lequel des attributs vus précédemment.

Balise	Attribut	Effet Visuel
 		retour à la ligne
et		paragraphe

II.2.3 Les listes

Une liste est un paragraphe structuré contenant une suite d'articles. Il en existe trois types:

- Ordonnée
- Non ordonnée
- De définition

Voici leur syntaxe:

Conteneur	Type de liste	Effet Visuel
  - article 1 - article 2 	Ordonnée	article 1 article 2 article 3
  - article 1 - article 2 	Non ordonnée	article 1 article 2 article 3
<dl> <dt>Terme</dt> <dd>Définition</dd>	De définition	article 1 définition 1 article 2

</dl>		définition 2

II.3. Utilisation de tableaux

On a souvent besoin de présenter des informations mieux structurées qu'avec des listes. Les tableaux permettent de les afficher en lignes et en colonnes.

Les tableaux sont définis comme étant des suites de lignes.

Le tableau est encadré par les balises *<TABLE>* et *</TABLE>*.
Le titre du tableau est encadrée par *<CAPTION> </CAPTION>*
Chaque ligne est encadrée par *<TR> </TR>* (*Table Row*, traduisez par *ligne du tableau*).
Les cellules d'en-tête sont encadrées par *<TH> </TH>* (pour *Table Header* : En-tête de tableau)
Les cellules de valeur sont encadrées par *<TD> </TD>* (Table Data: Donnée de tableau)

Par exemple le tableau:

```
<TABLE BORDER="1">

  <CAPTION> Voici le titre du tableau </CAPTION>

  <TR>

    <TH> Titre A1 </TH>
    <TH> Titre A2 </TH>
    <TH> Titre A3 </TH>
    <TH> Titre A4 </TH>
  </TR>
  <TR>
    <TH> Titre B1 </TH>
    <TD> Valeur B2 </TD>
    <TD> Valeur B3 </TD>
    <TD> Valeur B4 </TD>
  </TR>
</TABLE>
```

donne le résultat suivant:

Voici le titre du tableau

Titre A1	**Titre A2**	**Titre A3**	**Titre A4**
Titre B1	Valeur B2	Valeur B3	Valeur B4

II.3.1 Les attributs

	Balises auxquelles il s'applique	Valeur	Effet Visuel
ALIGN	THEAD TBODY TH TR TD	CENTER LEFT RIGHT JUSTIFY	centré gauche droite justifié
	CAPTION	TOP BOTTOM	au-dessus en-dessous
VALIGN (alignement vertical)	THEAD TBODY TH TR	TOP MIDDLE BOTTOM	en haut au milieu en bas

	TD		
BORDER=n	TABLE		taille de la bordure
CELLPADDING=n	TABLE		espacement de n pixels entre le contenu des cellules et son encadrement
CELLSPACING=n	TABLE		Epaisseur de la grille intérieure
FLOAT	TABLE	RIGHT LEFT	Spécifie la position du texte qui suivra </TABLE>
COLS=n	TABLE		Spécifie le nombre de colonnes
FRAME (contrôle les éléments individuels d'encadrement du tableau)	TABLE	NONE TOP BOTTOM TOPBOT SIDES ALL	aucun au-dessus en bas tout en haut sur les cotés tous
COLSPAN	THEAD TBODY TH TR TD		Fait déborder les cellules sur les colonnes adjacentes
ROWSPAN	THEAD TBODY TH TR TD		Fait déborder les cellules sur les lignes adjacentes

II.4. Les liens hypertextes et ancrage

II.4.1 Présentation des ancrages

Les **ancrages** (*liens hypertextes*) sont des éléments d'une page HTML (soulignés lorsqu'il s'agit de texte) qui emmènent dans un autre endroit lorsqu'on clique dessus. C'est ce qui permet de lier des pages Web entre elles.
Les liens hypertextes permettent de naviguer:

- vers un autre endroit du document
- vers un fichier HTML situé à un emplacement différent sur la machine qui héberge la page
- vers une autre machine

L'attribut principal des ancrages est *href*. Il s'écrit sous la forme:

* .. *

II.4.2 Les signets

On peut créer un signet dans une page c'est-à-dire marquer un endroit précis d'une page pour s'y rendre par hypertexte. Cela se fait grâce à l'attribut NAME ou ID (pour les browsers plus récents) Par exemple:

* ... *

On l'appellera grâce au lien suivant:

* ... *

On peut ainsi se déplacer à un endroit précis sur une autre page:

* ... *

II.5. Comment afficher des images sur une page web?

Quelques images sur votre site Web le rendront plus attractif. Cependant il ne faut pas sombrer dans l'excès car les images impliquent un temps de chargement assez long.

On utilise la balise <IMG> pour inclure une image, il ne crée pas de retour à la ligne. Ses trois principaux attributs sont:

- **SRC**: Indique l'emplacement de l'image (il est obligatoire)
- **ALIGN**: Spécifie l'alignement de l'image par rapport au texte adjacent. Il peut prendre les valeurs: TOP, MIDDLE, et BOTTOM (au-dessus, au milieu et en-dessous)
- **ALT**:Permet d'afficher un texte lorsque l'image ne s'affiche pas

Ainsi pour insérer une image, il faudra saisir une balise du type suivant :

<IMG SRC="url/image.gif\url/image.jpg" ALT="Texte remplaçant l'image">

Attribut	Valeur	Résultat
SRC		Adresse de l'image
BORDER=n		Nombre de pixels de la bordure. Sa couleur se définit dans l'attribut LINK ou TEXT de la balise <BODY>. Par défaut l'attribut Border vaut 1 ce qui crée un petit cadre autour de l'image. Pour ne pas avoir ce désagrément pensez à le définir comme étant égal à 0.
VSPACE		Nombre de pixels d'ajustement entre l'image et le texte au-dessus
WIDTH		Taille horizontale (en pixels ou en %).
HEIGHT		Taille verticale (en pixels ou en %).

II.6. Les formulaires en HTML

Les formulaires interactifs permettent aux auteurs de pages Web de dialoguer avec leurs lecteurs.

Le lecteur saisit des informations en remplissant des champs ou en cliquant sur des boutons, puis appuie sur un bouton de soumission (submit).

La balise FORM

Les formulaires sont délimités par la balise **<FORM> ... </FORM>**, une balise qui permet de regrouper plusieurs éléments de formulaire (boutons,champs de saisie,...) et qui possède les attributs obligatoires suivants:

- **METHOD** indique sous quelle forme seront envoyées les réponses "POST" est la valeur qui correspond à un envoi de données stockées dans le corps de la requête, tandis que "GET" correspond à un envoi des données codées dans l'URL, et séparées de l'adresse du script par un point d'interrogation (pour plus de renseignement sur les méthodes POST et GET, consultez l'article sur le protocole HTTP)
- **ACTION** indique l'adresse d'envoi (script CGI ou adresse email (mailto:adresse.email@machine)).

Voici quelques exemples de balises FORM:

<FORM METHOD=POST ACTION="mailtoflenr@hotmail.com">

<FORM METHOD=GET ACTION="http://yahoo.fr">

A l'intérieur de la balise FORM...

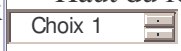
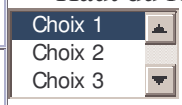
La balise *FORM* constitue en quelque sorte un conteneur permettant de regrouper des éléments qui vont permettre à l'utilisateur de choisir ou de saisir des données, ensemble de données qui seront envoyées à l'URL indiqué dans l'attribut *ACTION* de la balise *FORM* par la méthode indiquée par l'attribut *METHOD*.

Il est possible d'insérer n'importe quel élément HTML de base dans une balise *FORM* (textes,boutons,tableaux,liens,...) mais il est surtout intéressant d'insérer des éléments interactifs. Ces éléments interactifs sont:

- La balise *INPUT*: un ensemble de boutons et de champs de saisie
- La balise *TEXTAREA*: une zone de saisie
- La balise *SELECT*: une liste à choix multiples

II.6.1 Attributs de la balise FORM et types d'entrées

Balise	Attribut	Valeur	Résultat	Effet Visuel
<FORM> ... </FORM>	METHOD	POST GET		
	ACTION		envoie à l'adresse indiquée	
	ENCTYPE		spécifie le type de codage utilisé	
<INPUT>	TYPE	submit	effectue l'ACTION dans le balise <FORM>	Haut du formulaire Envoyer Bas du formulaire
		text	simple ligne de texte dont la longueur est donnée par l'attribut size	Haut du formulaire Bas du formulaire
		reset	efface le contenu du formulaire	Haut du formulaire Rétablir Bas du formulaire
		radio	bouton radio	Haut du formulaire Bas du formulaire
		checkbox	case à cocher	Haut du formulaire Bas du formulaire
	NAME		Nom	
	SIZE		Taille du texte	
<TEXTAREA> ... </TEXTAREA>	NAME		Zone de texte	Haut du formulaire Bas du formulaire
	ROWS			
	COLS			
<SELECT> <OPTION> ... </OPTION>	NAME			Haut du formulaire Choix 1 Bas du formulaire

</SELECT>	MULTIPLE		Plusieurs choix possibles	Haut du formulaire  Bas du formulaire
<OPTION> ... </OPTION>	SELECTED	Option par défaut		Haut du formulaire  Bas du formulaire
	VALUE	Valeur forcée		

III. ème Chapitre
Le Langage JAVASCRIPT

III.1.Introduction

III.1.1 De quoi s'agit-il ?

Les ordinateurs ne font qu'exécuter des suites d'instructions qui ont été écrites par des programmeurs. Il suffit de leur demander d'exécuter autant de fois que nécessaire des tâches simples pour qu'ils le fassent sans broncher. Le problème du programmeur est donc d'une part de savoir comment s'adresser poliment à son ordinateur pour être entendu, et d'autre part de traduire ses attentes en une succession de tâches simples. Ceci se fait en utilisant un langage de programmation; celui que nous allons explorer est JavaScript.
Dans ce chapitre vous trouverez les notions principales sur le **javascript**. Pour plus de détail consultez le site www.commentçamarche.com.

III.1.2 Le premier script

Apprendre à dire bonjour pour voir comment JavaScript s'insère dans un document HTML. Traditionnellement on commence à étudier un langage de programmation en lui faisant afficher le message "Bonjour !". C'est ce que nous allons faire ici, pour découvrir les liens entre les langages JavaScript et HTML.

III.1.3 JavaScript et HTML

Le langage HTML est un langage de description de pages Web. Il permet mettre des textes en forme, d'insérer des images, de créer des liens hypertextes, etc... Cependant, lorsqu'un document HTML est écrit, il affiche toujours les mêmes choses de la même façon. Le langage JavaScript est une extension du langage HTML qui permet de **calculer** ce qui doit être affiché en fonction des circonstances. Par exemple, si nous désirons créer un document HTML qui affiche la date, nous devons modifier le fichier HTML utilisé chaque jour. Grâce à JavaScript nous pourrons au contraire indiquer une seule fois comment calculer la date et donc l'afficher correctement. En ce sens, JavaScript permet de générer dynamiquement du code HTML.

III.2.Premier exemple avec JavaScript

Pour obtenir la même chose avec JavaScript, nous gardons la même structure de fichier, il nous suffit de demander à JavaScript d'afficher le code HTML de la ligne contenant "Bonjour tout le monde !". Cela se fait à l'aide de l'instruction **document.write** suivie de parenthèses contenant le code HTML entre guillemets. Cependant nous devons commencer par indiquer au navigateur que nous allons utiliser une instruction JavaScript. Cela se fait en utilisant la balise HTML **<SCRIPT LANGUAGE="JavaScript"></SCRIPT>**.

```
<HTML>
<HEAD><TITLE>Bonjour en JavaScript</TITLE></HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
document.write("<B>Bonjour tout le monde !</B>");
</SCRIPT>
</BODY>
</HTML>
```

III.2.1 Mettre le code JavaScript dans un fichier

Il est possible d'écrire le code JavaScript dans un fichier externe d'extension **.js**.
Pour l'exemple précédent, il suffit de créer un fichier nommé bonjour.js contenant la ligne :
Dans le fichier html, l'appel à JavaScript se fera simplement avec la ligne :
Cette possibilité se révèle utile lorsqu'on a des portions de code qui se répètent souvent.

III.3.Différents types d'expressions

Nous avons utilisé l'instruction document.write pour afficher une expression écrite entre guillemets. Celles-ci indiquent que ce qu'elles entourent est du code HTML que le navigateur va interpréter. Il est aussi possible d'utiliser des expressions sans guillemets. Dans ce cas JavaScript va essayer de les interpréter directement avant de les transmettre au navigateur. Les expressions mathématiques permettent de comprendre ce processus.
L'instruction **document.write("12+5*3");** provoque l'affichage du texte *12+5*3*. Par contre, l'instruction **document.write(12+5*3);** provoque l'affichage du texte *27*.
Observons le **script** suivant.

```
document.write("<BIG><B>B</B></BIG>onjour tout le monde !");
document.write("<BR>Un petit calcul : ");
document.write("12+5*3 = "); document.write(12+5*3);
document.write("<BR>Un peu de logique : ");
document.write("5*4=19 ?? "); document.write(5*4==19);
</SCRIPT>
```

On obtient l'affichage suivant :

Bonjour tout le monde !
*Un petit calcul : 12+5*3 = 27*
*Un peu de logique : 5*4=19 ?? false*

III.4.Les variables

III.4.1 Les types de variables

On peut classer les variables selon le type de leur contenu :

- chaîne de caractères
- valeur numérique
- valeur booléenne
- objet

Pour utiliser une variable, il est conseillé de la déclarer et de lui donner un contenu initial. Cela se fait en utilisant le mot **var** et le signe **=**.

Par exemple :

var v1="Flen";	La variable nommée **v1** contient la chaîne de caractères **"Flen"**.
var v2=18;	La variable nommée **v2** contient la valeur numérique **18**.
var v3=true;	La variable nommée **v3** contient la valeur booléenne **true**.

III.4.2 Opérations sur les variables

Les variables numériques peuvent être utilisées dans des expressions mathématiques faisant intervenir les quatre opérations élémentaires.

Les variables contenant des chaînes de caractères peuvent être *ajoutées*, cette opération se nomme la concaténation, elle consiste à construire une nouvelle chaîne de caractères en en assemblant deux. Ainsi, si v1="bonjour " et v2="Flen", alors v1+v2 est la chaîne de caractères contenant "bonjour Flen".

Les variables booléennes peuvent être combinées à l'aide des opérations logiques **non**, **et**, **ou** respectivement notées **!**, **&&**, ||.

Observons le comportement de l'opérateur + lorsqu'il est en présence d'expressions de types différents en exécutant le **script** suivant.

```
<SCRIPT LANGUAGE="JavaScript">
var v1="12";
var v2=5;
document.write("La variable v1 contient : ",v1,"<BR>");
document.write("La variable v2 contient : ",v2,"<BR>");
document.write("Avec v1+v2 on obtient :",v1+v2,"<BR>");
document.write("Avec v2+v1 on obtient : ",v2+v1,"<BR>");
</SCRIPT>
```

On obtient l'affichage suivant:

La variable v1 contient : 12 (chaîne de caractères)
La variable v2 contient : 5 (nombre)
Avec v1+v2 on obtient :125
Avec v2+v1 on obtient : 512

III.5.La fonction eval

Il arrive souvent que l'on doive interpréter une chaîne de caractères (par exemple une saisie de l'utilisateur) comme étant une valeur numérique. JavaScript permet de réaliser cette opération avec la fonction **eval**.

Observons l'effet du **script** suivant.

```
<SCRIPT LANGUAGE="JavaScript">
var v1="12";
var v2=5;
document.write("La variable v1 contient : ",v1,"<BR>");
document.write("La variable v2 contient : ",v2,"<BR>");
document.write("Avec v1+v2 on obtient :",v1+v2,"<BR>");
document.write("Avec eval(v1)+v2 on obtient : ",eval(v1)+v2,"<BR>");
</SCRIPT>
```

On obtient l'affichage suivant:

La variable v1 contient : 12
La variable v2 contient : 5
Avec v1+v2 on obtient :125
Avec eval(v1)+v2 on obtient : 17

III.6.La fonction prompt

Il arrive souvent que le contenu d'une variable doive être choisi par l'utilisateur du programme, et non par le programmeur. La fonction **prompt()** fournit une façon de réaliser cette opération avec JavaScript. Elle attend en paramètre deux chaînes de caractères: la première servira de message d'invite et la seconde de réponse proposée par défaut.

Observons le **script** suivant.

```
<SCRIPT LANGUAGE="JavaScript">
var nom;
nom=prompt("Quel est ton nom ?","");
document.write("<BIG><B>B</B></BIG>onjour "+nom+" !<BR>");
</SCRIPT>
```

Exemple: afficher le périmetre nsi qu'air d'un carré en introduisant son côté

```
var cote;
cote=prompt("Côté du carré : ","");
var perim=4*cote;
var aire=cote*cote;
document.write("<BR>Côté du carré : ",cote);
document.write("<BR>Périmètre : ",perim);
document.write("<BR>Aire : ",aire);
```

III.7.Définir de nouvelles fonctions

JavaScript permet de créer simplement de nouvelles fonctions qui permettent de répéter plusieurs fois les mêmes suites d'instructions.
Il est parfois utile de regrouper un certain nombre d'instructions dans un bloc réutilisable plusieurs fois. Cela se fait en définissant des fonctions qui pourront être utilisées comme les fonctions prédéfinies de JavaScript.

III.7.1 Structure générale

Une fonction est une suite d'instructions à laquelle on a donné un nom. Elle peut éventuellement renvoyer un résultat et utiliser des paramètres. Pour la définir, c'est à dire indiquer la suite d'instructions à exécuter lorsqu'on l'appelle, on utilise le mot-clé **function**.
On définit une fonction de la façon suivante :

- La suite d'instructions à exécuter se trouve entre deux accolades.
- La liste de paramètres est facultatives, mais on doit écrire les parenthèses même s'il n'y a pas de paramètres.
- La ligne "return résultat;" n'est utilisée que si la fonction renvoie un résultat.

Définir une fonction ne provoque pas son exécution immédiate. Cela donne simplement la possibilité de l'appeler ultérieurement comme toute fonction JavaScript déjà définie, document.write(), eval() ou prompt().
Il peut être intéressant de définir des fonctions personnelles dans un fichier externe .js qui pourra être utilisé comme une bibliothèque avec plusieurs documents HTML.

Exemple*:* Le nombre secret

```
<HTML>
<HEAD>
<TITLE>Nombre secret</TITLE>
</HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">

//nombre entre 0 et 99
var secret=Math.round(Math.random()*100);

//nombre d'essais
var nbEssais=0;

//fonction de contrôle de la réponse
function control() {
nbEssais++;
var n=document.formu.prop.value;
if (isNaN(n)) alert("Entrer un nombre !");
else {
var nb=eval(n);
var S="Essai "+nbEssais+" : ";
if (nb==secret) S+="Bonne réponse !!!";
else if (nb < secret) S+="Trop petit.";
else S+="Trop grand.";
document.formu.reponse.value=S;
}
}
</SCRIPT>
```

```
<H1>Le nombre secret</H1>
<HR>
<P>L'ordinateur a choisi un nombre entier inférieur à 100. A vous de le retrouver le plus rapidement possible en proposant des nombres et en tenant compte des réponses fournies.</P>
<FORM NAME="formu">
<DIV ALIGN="center">
<B>Nombre proposé : </B>
<INPUT NAME="prop" TYPE="text" SIZE="5">
 
<INPUT TYPE="button" VALUE="Contrôle" ONCLICK="control()">
<BR>
<INPUT NAME="reponse" TYPE="text" SIZE="30" READONLY>
</DIV>
</FORM>
<HR>
<A HREF="nbsecret.html">Autre partie</A>
</BODY>
</HTML>
```

III.8.Les formulaires élémentaires

Nous avons vu que la fonction prompt permet à l'utilisateur d'entrer des données. Une autre méthode consiste à utiliser un formulaire HTML contenant des lignes de saisie, une zone de réponse et un bouton de déclenchement des calculs. Ceci nous amènera à définir des fonctions associées à l'évènement ONCLICK du bouton.

III.8.1 Interpréter un formulaire

Le langage HTML permet de construire des formulaires; JavaScript permet d'exploiter et de modifier leur contenu. Nous allons étudier cette possibilité en construisant un document permettant de convertir des Euros en Francs.

a. Formulaire HTML

Commençons par construire le formulaire HTML de façon classique.
On obtient la **page** suivante :

```
<HEAD><TITLE>Euro-Franc</TITLE></HEAD>
<BODY>
<H1>Conversion Euro-Franc</H1><HR>
<FORM><TABLE><TR><TD>
Valeur en Euros :</TD><TD>
<INPUT TYPE="text" SIZE="10"></TD></TR><TR><TD>
Valeur en Francs :</TD><TD>
<INPUT TYPE="text" SIZE="10"></TD></TR><TR>
<TD COLSPAN="2"><DIV ALIGN="center">
<INPUT TYPE="button" VALUE="Convertir"></TD></TR>
</TABLE></FORM>
</BODY>
</HTML>
```

Le formulaire s'affiche bien, mais cliquer sur le bouton "Convertir" ne provoque aucune réaction.

b. Traitement avec JavaScript

JavaScript va nous permettre de récupérer le contenu de la zone de saisie euro, d'effectuer la conversion en francs et d'afficher le résultat dans la zone de saisie franc lors de chaque clic sur le bouton "Convertir".

c. Nommer les composants

Pour pouvoir faire référence au formulaire et à ses composants, il est possible de les nommer en utilisant l'attribut **NAME** des balises correspondantes.

Le formulaire sera nommé "formu" en écrivant :
<FORM NAME="formu">

La zone de saisie du montant en euros sera nommée "euro" en écrivant :
<INPUT NAME="euro" TYPE="text" SIZE="10">

La zone de saisie du montant en francs sera nommée "franc" en écrivant :
<INPUT NAME="franc" TYPE="text" SIZE="10">

d. Accès aux composants

L'accès au contenu des composants se fait à travers une cascade d'objets associés.
Il y a d'abord l'objet document.

1. L'objet document possède un champ-objet nommé formu qui représente le formulaire.
2. L'objet formu possède à son tour un champ-objet nommé euro qui représente la zone de saisie du montant en euros.
3. L'objet euro possède enfin un champ nommé value qui représente son contenu.

Ainsi, pour obtenir le contenu de la zone de saisie du montant en euros, on écrira :

document.formu.euro.value

De même, pour faire référence au contenu de la zone de saisie du montant en francs, on écrira :

document.formu.franc.value

e. Ecriture de la fonction de conversion

Ecrivons maintenant la fonction conversion() qui sera chargée de lire le montant en euros, de le convertir en francs et d'afficher la réponse dans la zone de saisie du montant en francs.

Elle peut prendre la forme suivante :

```
function conversion() {
 //lire la valeur euro
 var eu=eval(document.formu.euro.value);
 //conversion et arrondi
 var fr=eu*6.55957;
 fr=Math.round(fr*100)/100;
 //affichage du résultat
 document.formu.franc.value=String(fr);
}
```

Nous avons utilisé la fonction **Math.round** pour obtenir l'arrondi entier d'un nombre à virgule. La formule "fr=Math.round(fr*100)/100;" permet d'obtenir un arrondi au centième près.

f. Association clic bouton / appel fonction

Il nous reste à indiquer que chaque clic sur le bouton "Convertir" doit provoquer l'exécution de la fonction conversion().

Cela s'obtient en utilisant l'attribut ONCLICK du bouton. On écrira donc :

```
<INPUT TYPE="button" VALUE="Convertir" ONCLICK="conversion()">
```

g. Résultat final

Il nous suffit de regrouper tous les éléments précédents pour obtenir le **résultat** escompté.
Le fichier HTML a le contenu suivant :

```
<HTML>
<HEAD><TITLE>Euro-Franc</TITLE></HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
function conversion() {
 //lire la valeur euro
 var eu=eval(document.formu.euro.value);
 //conversion et arrondi
 var fr=eu*6.55957;
 fr=Math.round(fr*100)/100;
 //affichage du résultat
 document.formu.franc.value=String(fr);
}
</SCRIPT>
<H1>Conversion Euro-Franc</H1><HR>
<FORM NAME="formu"><TABLE><TR><TD>
Valeur en Euros :</TD><TD>
<INPUT NAME="euro" TYPE="text" SIZE="10"></TD></TR><TR><TD>
Valeur en Francs :</TD><TD>
<INPUT NAME="franc" TYPE="text" SIZE="10"></TD></TR><TR>
<TD COLSPAN="2"><DIV ALIGN="center">
<INPUT TYPE="button" VALUE="Convertir" ONCLICK="conversion()">
</TD></TR></TABLE></FORM>
</BODY>
</HTML>
```

III.9.Les conditions

III.9.1 Utilisation de if

La syntaxe de l'instruction **if** prend la forme suivante :

```
if (expression booléenne) {
  instruction 1;
  instruction 2;
  ...
}
else {
  instruction 1;
  instruction 2;
  ...
}
```

Lorsque l'expression booléenne prend la valeur **true**, c'est le premier groupe d'instructions qui est exécuté. Dans le cas contraire, (**false**), c'est le deuxième groupe d'instructions, situé après le mot-clé **else**, qui est exécuté.
L'expression booléenne doit toujours être écrite entre des parenthèses. Les deux groupes d'instructions sont écrits entre des accolades; celles-ci peuvent être supprimées pour les groupes ne contenant qu'une seule instruction.

Le mot-clé **else** et le groupe d'instructions associé sont facultatifs; s'ils ne sont pas utilisés, aucune instruction n'est exécutée lorsque l'expression booléenne prend la valeur **false**.

III.9.2 Utilisation de switch

L'instruction switch permet de faire exécuter certaines instructions selon la valeur prise par une variable. Elle prend la forme suivante :

```
switch (variable) {
  case val1 : instructions
  case val2 : instructions
  case val3 : instructions
  ......
  default : instructions
}
```

Dès que la variable testée prend l'une des valeurs proposées, les instructions correspondant à cette valeur et aux valeurs suivantes sont exécutées. Si aucune valeur n'est reconnue, ce sont les instructions correspondant à **default** qui sont exécutées.

La variable testée doit être écrite entre parenthèses. L'ensemble des instructions correspondant aux différentes valeurs possibles doit être écrit entre des accolades.

Pour que seules les instructions correspondant à une valeur soient exécutées, on utilise le mot-clé **BREAK** qui permet de sortir directement de la structure switch.

III.10. Les boucles

III.10.1 Boucle for

L'instruction **for** prend la forme générale suivante :

```
for (initialisation; condition; transition) {
  instruction1;
  instruction2;
  ....
 }
```

Les instructions situées entre accolades forment la boucle qui sera exécutée. Les accolades ne sont nécessaires que lorsqu'il y a plusieurs instructions.
La première partie, située entre parenthèses, contient 3 parties :

- **initialisation**: instruction exécutée avant le premier passage dans la boucle.
- **condition**: expression booléenne qui commande les passages dans la boucle qui se font tant qu'elle vaut true.
- **transition**: instruction qui est exécutée avant chaque nouveau passage dans la boucle.

On utilise en général les boucles for lorsque le nombre de passage dans la boucle est connu dès le départ. C'est un compteur (souvent une variable nommée i) qui sert à déterminer le nombre de passages. Ainsi, pour répéter n fois une suite d'instructions, on utilisera le modèle suivant :

```
for (var i=0; i<n; i++) {
  instruction1;
  instruction2;
  ....
}
```

Le compteur i part de 0. Grâce à l'instruction **i++** il est incrémenté (augmenté d'une unité) avant chaque nouveau passage. La boucle n'est plus exécutée lorsqu'il prend la valeur n. Comme on est parti de 0, le nombre de passage dans la boucle est bien égal à n.

III.10.2 Boucle while

L'instruction **while** prend la forme générale suivante :

```
while (condition) {
  instruction1;
  instruction2;
  .....
}
```

La condition, toujours écrite entre parenthèses, est une expression booléenne. **Tant que** cette expression est égale à **true** la boucle formée par les instructions écrites entre accolades est répétée.
Attention : les instructions formant la boucle doivent modifier la valeur de la condition pour que l'exécution puisse s'arrêter.
L'instruction **while** peut aussi prendre la forme :

```
do {
  instruction1;
  instruction2;
  .....
} while (condition)
```

Comme dans la forme précédente, la boucle formée par les instructions entre accolades est répétée **tant que** l'expression booléenne représentant la condition est égale à **true**.

III.11. Les tableaux

Lorsqu'un programme nécessite la création de nombreuses variables, on les regroupe dans des **tableaux**. Chaque variable correspond alors à un indice et il est facile de parcourir le tableau en utilisant une boucle. Vous pouvez créer vos propres tableaux ou utiliser ceux que JavaScript définit pour stocker certaines informations.

III.11.1 Création

Les tableaux JavaScript sont des objets de type **Array**. La création d'un tableau se fait donc en utilisant le mot-clé **new**. On peut donc créer un tableau T en écrivant :

```
var T=new Array();
```

Le tableau T sera alors vide.

Si le nombre d'éléments du tableau est connu, on écrira, par exemple :

```
var T=new Array(5);
```

Le tableau T pourra alors contenir 5 éléments.
Il est enfin possible de créer un tableau en indiquant directement son contenu. Par exemple, pour créer le tableau nommé *jours* qui contiendra les noms des jours de la semaine, on pourra écrire :

```
var jours=["dimanche","lundi","mardi","mercredi","jeudi","vendredi","samedi"];
```

III.11.2 Utilisation

Pour avoir accès, en lecture ou en écriture, à un élément d'un tableau on utilise le nom du tableau et l'indice de l'élément qui est un entier indiquant sa position. Ainsi, avec le tableau *jours* créé précédemment, on pourra écrire :
jours[i]

pour obtenir le nom du jour d'indice i. L'indice est toujours placé entre deux crochets.
Attention, les indices commencent à 0; le premier élément du tableau a donc l'indice 0, le deuxième a l'indice 1, etc...
On utilise la même notation (avec les crochets) pour lire le contenu d'un tableau et pour donner une valeur à un élément. Par exemple, pour créer un tableau de 5 nombres, on pourra écrire :

```
var T=new Array(5);
T[0]=12;
T[1]=15;
T[2]=7;
T[3]=14;
T[4]=10;
```

III.11.3 Quelques propriétés et méthodes des objets tableaux

Les tableaux étant des objets ils disposent de propriétés et de méthodes dont l'utilisation peut se révéler intéressante.

a. Propriété length

La propriété **length** d'un tableau indique le nombre d'éléments qu'il contient.
Par exemple, pour connaître le nombre d'éléments d'un tableau T, on écrira :

```
var n=T.length;
```

b. Méthode join()

La méthode **join()** permet de regrouper tous les éléments d'un tableau dans une seule chaîne de caractères. Par défaut le séparateur est une virgule, on peut en choisir un autre en l'indiquant comme paramètre.
Par exemple, si le tableau T contient les nombres 1, 3 et 5 :
l'appel de T.join() fournira la chaîne de caractères "1,3,5".
l'appel de T.join(" et ") fournira la chaîne de caractères "1 et 3 et 5".

c. Méthode split() des chaînes de caractères

Les objets chaînes de caractères possèdent une méthode **split()** qui réalise l'opération contraire de join(). Elle découpe une chaîne de caractères et renvoie le résultat dans un tableau. Le séparateur utilisé doit être fourni en argument.
Par exemple, si la chaîne de caractères S contient "1,3,5", S.split(",") renvoie un tableau de 3 éléments qui sont "1", "3" et "5".

d. Méthode reverse()

La méthode **reverse()** des tableaux fournit un nouveau tableau contenant les mêmes éléments mais dans l'ordre inverse.
Par exemple, si le tableau T contient 1, 3 et 5, le tableau T.reverse() contient 5, 3 et 1.

e. Méthode sort()

La méthode **sort()** des tableaux permet de les trier en utilisant l'ordre lexicographique (ordre du dictionnaire).
Par exemple, si le tableau T contient les chaînes de caractères "encore", "avec" et "dans", l'appel de T.sort() modifiera l'ordre des éléments en "avec", "dans" et "encore".
La méthode sort peut utiliser en argument un nom de fonction de comparaison. Celle-ci doit comparer deux variables a et b et renvoyer 1 si a est après b, -1 si a est avant b et 0 si a et b ont même rang.
Exemple d'application

L'**exemple suivant** construit un tableau à partir d'une liste de mots, compte le nombre de mots, inverse l'ordre des mots et place les mots dans l'ordre alphabétique.

```
<HTML>
<head><title>Méthodes des tableaux</title></head>
<body>
<h1>Méthodes des tableaux</h1></HR>
<SCRIPT LANGUAGE="JavaScript">
var S="poire orange abricot pomme fraise raisin";
var T=S.split(" ");
document.writeln("<P>Liste de mots : ",S,"</P>");
var n=T.length;
document.writeln("<P>Il y a ",n," mots dans la liste.</P>");
document.writeln("<P>Ordre inverse : ", T.reverse().join(" "));
T.sort();
document.writeln("<P>Ordre alphabétique : ",T.join(" "));
</SCRIPT>
</body></HTML>
```

IV. eme CHAPITRE
Le Langage PHP

IV.1. Origine de PHP

De simple idées aboutissent à de magnifiques créations. PHP est né avec le site de **RASMUS Lerdof** en 1994: une page personnelle, avec son CV qui permettait à l'origine de conserver une trace des visiteurs. A l'époque PHP supportait déjà des requêtes SQL et, comme cela arrive souvent sur le Web, des internautes ont rapidement voulue leur propre copie de programmes. Rasmus a donc décidé de mettre en ligne sa première version 1.0 de PHP (Personal Home Page).

La version 2.0 permet au développeur d'intégrer des instructions de programmation puissantes directement dans du code HTML. En janvier 1999, plus de 100.000 sites ont adopté le langage PHP.

Une communauté de développeurs s'est crée autour du langage. De nouvelles fonctions ont ainsi pu voir le jour et de nombreux bugs ont été rapidement résolus. La version 3.0 sortit le 6 juin 1998.

La version PHP4 est sorti en Avril 2000 quant à la version PHP5 vit le jour en 2003.

IV.2. Description

La souplesse et la performance de PHP n'est plus à prouver. De nombreux sites dans le monde, et parmi eux de très nombreuses applications professionnelles ont choisi PHP.
Lorsqu'on réalise de sites Internet dynamiques, on est vite amené à utiliser une base de données pour stocker des informations. Il existe de multiple de bases de données utilisables avec PHP, Mais le MYSQL est très largement diffusé sur Internet.

MYSQL est un système de gestion de base de données (SGBD). Cela signifie que c'est un ensemble d'applications permettant de manipuler les données (ajout, suppression, modification, et lecture), mais aussi d'en contrôler l'accès. Il est donc tout naturel que les sites interactifs fassent appel au serveur de base de données gratuit MySQL.

IV.3. Installation

Pour pouvoir utiliser PHP, il y a tout un tas de fichiers à installer... serveur Apache, PHP, MySQL, PhpMyAdmin... heureusement, quelqu'un a pensé à simplifier ça !

Installer EasyPhP

1. Allez sur le site www.easyphp.org, téléchargez le logiciel gratuit **EasyPHP**... Exécutez-le après avoir éventuellement désactivé le PWS , une nouvelle icône apparaît à côté de l'heure en bas de l'écran indiquant que **Easyphp** est actif (par défaut Apache, PHP et MySQL aussi);

2. Créez sur votre PC un dossier **essais** dans le dossier racine **...\EasyPHP\www** ; C'est dans ce dossier que vous mettrez les fichier *.PHP.

3. Recopiez dans un éditeur la source suivante puis enregistrez sous le nom **essai_1.php** attention à l'extension ! dans le dossier **essais** ;

```
<HTML><HEAD><TITLE>Essai 1</TITLE>
</HEAD><BODY>
  Nous sommes le <? echo date("d/m/Y"); ?>
</BODY></HTML>
```

4. Exécutez en tapant l'URL **http://localhost/essais/essai_1.php** dans le navigateur... demandez la connexion si nécessaire (elle sera interne donc gratuite) et la date courante doit apparaître ;
   Autre possibilité : click droit sur l'icône, puis Web local qui liste vos dossiers du www...

Remarques

1. Pour pouvoir produire un fichier php il faut avoir paramétré Windows afin qu'il affiche toutes les extensions sinon un .txt risque d'être ajouté ! (Menu du dossier : Outils > Options des dossiers... > Affichage)

2. "L'exemple n'affiche pas la date !" : Jusque là, vous cliquiez sur l'icône de votre page html pour lancer le navigateur et l'afficher... Ce n'est plus le cas avec vos pages asp ou php, il faut se connecter au serveur interne en tapant dans la barre d'adresses de votre navigateur une URL complète: **http://localhost/essais/essai_1.php**.

3. Le navigateur peut parfois afficher le dialogue de connexion, comme pour aller sur le Web, et il faut alors choisir "**Se connecter**" mais ce sera une connexion interne, donc gratuite ! Si la date ne s'affiche pas, c'est que vous n'êtes pas connecté au serveur interne, inutile donc d'aller plus loin, relisez cette fiche et REESSAYEZ!!!
   Astuce : Internet Explorer affiche **Intranet local** en bas de fenêtre, si vous êtes correctement connecté !

4. Les deux serveurs PWS (ou IIS) et Apache ne fonctionnent pas ensemble à moins qu'ils ne soient sur des ports différents... donc:

   - soit vous pensez à désactiver l'un avant d'activer l'autre (click droit sur leurs icônes respectives) ;
   - soit vous mettez Apache sur le port 81 en modifiant le fichier **...\EasyPHP\apache\conf\httpd.conf** et vous utilisez alors **http://localhost:81/** au début de vos URL locales pour le PHP.

5. Pour faire démarrer PHPMY Admin: dans IE taper http://127.0.0.1/mysql/

IV.4. Mise en pratique de PHP

La notation en PHP n'est pas très difficile. Le code PHP est simplement inséré au sein d'un document HTML tout à fait classique. L'important est que vous signaliez qu'il s'agit de code Php avec des balises comme ci-dessous:

<?php
…Instructions PHP
?>

ou

<?php
…Instructions PHP
?>

Premiers scripts : Afficher du texte:
A partir du moment où vous placez du code PHP dans un fichier *.htm ou *.html, vous devrez renommer ce fichier en *.php. Pour que le serveur analyse et exécute **les requête PHP.** Si vous ne faites pas cette

manipulation, le code risque d'apparaître en toute lettre dans le navigateur sans être exécuter par le serveur.

```
<html>
<head>
</head>
<body>
<p>
<?php
echo "Bienvenue à l'IFP""";
<?
</p>
</body>
</html>
```

Essayez*:*

```
<?php echo "hello word ! <br> bonjour monde !" ?>
```

C'est la fonction **echo** que nous utiliserons pour afficher du texte à l'écran. Ici, on voit bien que la phrase n'est pas du tout formatée, voici donc comment l'on peut inclure les balises **HTML** dans le PHP (ci-dessous).
Notez qu'une instruction PHP se termine par un point virgule**(;)**.

```
<?php
echo"<font face=\"arial\" size="2\" color=\"#003300\"> Bienvenue à l'IFP!</font>";
<?
```

Nous avons ajouté la balise **Font** comme en HTML pour formater le texte, la grosse différence, étant l'ajout des antislash avant chaque caractère spécial.
Exemple: " devient \", il s'agit tout simplement de dire à PHP qu'il ne s'agit pas d'un élément de la syntaxe du script mais bien d'une apostrophe ou guillemet qu'il faut pas afficher.
<**br**> est une balise HTML qui permet le saut de ligne.

```
<html>
<head>
</head>
<body>
<p>
<?php /* commentaire de
2 lignes */
echo "hello word !", "<br>" ;
echo "bonjour" ; // commentaire de fin de ligne
echo " monde !" # commentaire de fin de ligne
?>
</p>
</body>
</html>
```

IV.5. La notion de variable

Une variable permet de définir une valeur qui peut être modifiée tout au long de l'exécution du script, alors qu'une constante possède une valeur fixe qui n'est définie qu'une seule fois. Les variables peuvent contenir plusieurs types de données : entier, double et chaîne.

```
$variable = 5; // entier
$variable = 5.0 // double
$variable = "5"; // chaîne
```

IV.5.1 Définir des variables

Pour définir des variables, il suffit de lui assigner une valeur. Pour cela, il faut utiliser la syntaxe suivante :

$nom_de_la_variable = "valeur"; // chaîne valeur
$nom_de_la_variable = 10; // entier valeur

Attention toutefois, $Variable ne correspond pas a $variable, ce sont deux variables différentes.

Pour récupérer une variable, on l'appelle tout simplement par son nom :

echo $variable;
echo "Texte avant $variable Texte après";
echo "Texte avant " . $variable . " Texte après";

On peut faire des opérations sur les variables .

$a = 10;
$b = 15;
$c = $a + $b;
echo $c;
ou encore:
$a=20;
$b=$a/2;
echo $b;

IV.5.2 Afficher la date du jour et l'heure en cours

La fonction date() permet d'obtenir l'heure local du serveur mais attention l'heure local est fonction de la situation géographique du serveur n lui-même. En effet, un serveur situé au Canada vous donnera l'heure du Canada.

Nous utilisons pour cela la fonction date():
<?
$date = date("d-m-y");
$heure = date("H:i");
echo "Nous sommes le $date() et il est $heure";
?>

Dans le code ci-dessus nous générons la variable $date en lui donnant la valeur de ce que retourne la fonction date("d-m-y"): le jour, le mois et l'année. Les paramètres contenues entre les parenthèses d-m-y peuvent être placés dans l'ordre que vous désirez.

Astuce pratique: Vous voudrez peut être remplacer le"**:**" par un "**h**" pour que le format Français soit vraiment respecté (ex: 13h15), dans ce cas il suffit de procéder ainsi : date("H\hi"), remarquez l'antislash devant le deuxième "h", en effet, il faut spécifier à PHP que cette lettre est un caractère à afficher et non à nouveau l'heure en cours, on y place un backslash comme devant tous les caractères spéciaux qui ne doivent pas être interprétés comme du code.

Pas de déclaration de variable:
<html>
<head>

```
</head>
<body>
<p>
<?php
$a = 3; echo "\$a = $a<br>" ;
$a = 3.3; echo "\$a = $a<br>" ;
$a = "3a"; echo "\$a = $a<br>" ;
echo "\$b = $b<br>" ;
?>
</p>
</body>
</html>
```

• EXECUTION
$a = 3
$a = 3.3
$a = 3a

les variables en PHP n'ont **pas besoin d'être déclarées**,

- on peut commencer à les utiliser sans en avoir averti l'interpréteur au préalable !
- si la variable existait précédemment, son contenu est utilisé,
- sinon l'interpréteur lui affectera la valeur en lui assignant NULL par défaut
- Il n'est pas nécessaire en PHP de typer les variables, c'est-à-dire de définir leur type, il suffit de leur assigner une valeur pour en définir le type.

IV.5.3 variable affectée ou non

```
<?php
echo "variable non declaree isset : " ; echo isset($b); echo "<br>" ;
$b = NULL;
echo "variable NULL isset : " ; echo isset($b); echo "<br>" ;
$b = 13;
echo "variable a valeur $b isset : " ; echo isset($b); echo "<br>" ;
unset($b);
echo "apres unset(\$b) isset : " ; echo isset($b); echo "<br>" ;
?>
```

• EXECUTION

variable non déclarée isset :
variable NULL isset :
variable a valeur 13 isset : 1
après unset($b) isset :

- **isset**(variable): détermine si une variable est affectée/définie, renvoie TRUE , sinon FALSE
- **unset**(variable) : détruit une variable

unset() à l'intérieur d'une fonction n'a pas la même portée de destruction suivant le type de variable

IV.5.4 Définir des constantes

Pour définir des constantes, on utilise la fonction define(). La plupart du temps, le nom d'une constante est en majuscule. Par la suite, si vous souhaitez récupérer la valeur d'une constante, il suffit de l'appeler par

son nom :

```
<?php
define("PI", 3.1416);
define("BR", "<br>");
echo PI; echo BR;
$a = 2 * PI ;
echo $a; echo BR;
?>
```

• ***EXECUTION***

3.1416
6.2832

Une constante est une valeur fixe nommée :

- l'identificateur suit la syntaxe PHP des identificateurs sans le $!!
- la valeur est integer, boolean, flot ou string
- sa portée est globale

IV.5.5 Une variable Existe ou pas

Pour vérifier si une variable existe, on utilise la fonction **empty**() qui renvoie true si elle existe et false sinon.

```
$variable = "a";
if(empty($variable))
echo "existe";
else echo "n'existe pas";
```

Pour vérifier si une constante existe, il faut utiliser dans ce cas la fonction **defined**() qui renvoie true en cas de réussite et false sinon.

```
define("CONSTANTE","a");
if(defined("CONSTANTE"))
echo "Existe";
else echo "N'existe pas";
```

IV.5.6 Type de variable

La fonction **gettype**() permet de changer le type de données d'une variable. On peut assigner les types suivants:

- Class
- integer
- object
- string
- array
- double
- unknown type

if(gettype($variable) == "integer")

La fonction settype() permet de définir explicitement le type d'une variable.

```
$variable = 2.5;
settype($variable, "integer");
```

La variable $variable renverra maintenant 2 et non 2.5.

IV.6. Variable variable

Une variable variable permet de prendre la valeur d'une variable 1 pour donner le nom a la variable 2.

```
$variable1 = "nom_de_la_variable2";
$$variable1 = "valeur_variable2";
echo $$variable1;
```

IV.7. Les formulaires

```
<html>
<head>
<title>formulaire à soumettre</title>
</head>
<body>
<h2 align=center> formulaire à soumettre</h2>
<p>

Renseigner le chien :<br>

<form method="post" action="submit1.php">
        Nom : <input type="text" name="nom"><br>
        Maitre : <input type="text" name="maitre"><br>
        Aboiement : <input type="text" name="aboiement"><br>
        Nombre de puces : <input type="text" name="nombrePuces"><br>
                <input type="submit" name="submit" value="Soumettre">
</form>

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

- Les **formulaires** constituent un moyen de récupérer de l'information provenant de l'utilisateur

la balise form délimite/définit le formulaire

- l'attribut method permet de choisir la méthode d'envoi HTTP entre POST et, à déconseiller, GET (envoi de paramètres visibles)
- l'attribut action précise l'URL demandée par la requête HTTP

un formulaire se compose d'éléments :

- la balise **input** définit des champs d'information simples elle possède un attribut name pour désigner l'élément et éventuellement un attribut value de valeur initialeelle se sub-divise selon son attribut type
- **text** un petit champ pour que l'utilisateur saisisse un nombre ou une phrase
- **button** pour cliquer dessus
- **checkbox** des cases à cocher (information on/off)
- **radio** des "radio-boutons" (information on/off mais un seul peut être coché à la fois !)
- **hidden** est un champ caché pour transmettre des informations vers le serveur, sans que l'utilisateur ne le sache : c'est très utile pour gérer un caddie ...
- password un champ texte mais masquant le texte tapé
- submit est l'action indispensable lorsque le formulaire est utilisé pour envoyer de l'information

depuis l'utilisateur vers le serveur : il "envoie" les données récupérées de l'utilisateur dans le formulaire vers le serveur

- reset pour remettre le formulaire dans son état initial
- select pour les listes
- textarea pour un texte multi-lignes.

* Voici quelques exemples d'objets de formulaire :

Nom	Resultat	Code
Zone de texte	valeur initiale	<input type='text' name='nom' size='20' value='valeur initiale'>
Zone de texte déroulante	Valeur initiale	<textarea rows='1' name='nom' cols='16'>Valeur initiale</textarea>
Case a cocher	☑	<input type='checkbox' name='nom' value='ON' checked>
Menu déroulant	valeur1	<select size='1' name='nom'> <option value='valeur1'>valeur1</option> <option value='valeur2'>valeur2</option> <option value='valeur3'>valeur3</option> </select>
Case d'option	◉	<input type='radio' value='nom' checked name='nom_groupe'>
Bouton	titre	<input type='submit' value='titre' name='nom'>

IV.8. Les expressions:

Quelques opérateurs peu ou mal connus

```
<?php
$i = 2;
$j = 6;
echo "int i = $i, j = $j<br>" ;
echo 'expression $i++*$j+1 : ' ; echo $i++*$j+1; echo "<br>" ;
echo "i = $i, j = $j<br>" ;
$j += $i ; echo 'affectation $j += $i : <br>' ;
echo "j = $j<br>" ;
echo 'expression ($j > $i ? $j - $i : $i - $j) ' ;
echo $j > $i ? $j - $i : $i - $j; echo "<br>" ;
?>
```

• **EXECUTION**

```
int i = 2, j = 6
expression $i++*$j+1 : 13
i = 3, j = 6
affectation $j += $i :
j = 9
expression ($j > $i ? $j - $i : $i - $j) 6
```

• ++ et -- opérateurs de pré ou post incrémentation et décrémentation :
o dans l'expression ci-dessus, $i++ la valeur 2 de $i est utilisée dans l'expression avant de l'incrémenter de 1, c'est une post-incrémentation
• $var op= expression
est équivalente à $var = $var op expression ,où op est +, -, /, *, !, ., &, ...
• exp_bool ? expression1 : expression1
utilise l'opérateur conditionnel ternaire

signifie si exp_bool vaut true, alors expression1 sinon expression2

IV.9. Création de tableaux

Un tableau est un conteneur renfermant plusieurs valeurs. Chaque élément d'un tableau possède une valeur propre ainsi qu'une clé qui permet de faire référence à cet élément.

Variable	$provider[0]	$provider[1]	$provider[2]
Valeur	wanadoo	club-internet	aol

Il existe plusieurs manières pour créer des tableaux :

a. Avec des indices entiers

1. **Affecter des valeurs aux variables :**
   $provider[] = "wanadoo";
   $provider[] = "club-internet";
   $provider[] = "aol";
   En ne mettant rien entre crochet, l'élément est tout simplement ajouté a la suite.

2. **Affecter des valeurs et des indices explicites :**
   $provider[0] = "wanadoo";
   $provider[1] = "club-internet";
   $provider[2] = "aol";
   L'indice peut être n'importe quel entier, ainsi, cela aurait pu être 20, 80 et 90.

3. **Affecter des valeurs avec array() :**
   $provider = array("wanadoo", "club-internet", "aol");
   Ainsi **$provider[3]** va renvoyer **aol**.

4. **Affecter des valeurs et des indices avec array() :**
   $provider = array(1=> "wanadoo", "club-internet", "aol");
   Ainsi **$provider[3]** va renvoyer **aol.** Tout simplement, en placant l'opérateur **=>**, vous changez l'indice, c'est a dire que les autres éléments vont prendre les valeurs suivantes, ici ce sera **1, 2 et 3** au lieu de **0, 1 et 2**. Il est possible de placer l'opérateur devant n'importe quelle valeur du tableau.

b. Avec des indices chaines

Comme nous avons déja pu le constater ci-dessus, il est possible d'utiliser des entiers comme indices. Toutefois, il est aussi possible d'utiliser des chaînes de caractères, ce qui peut être plus clair :

$provider["rapide"] = "wanadoo";
$provider["moyen"] = "club-internet";
$provider["lent"] = "aol";
*La variable **$provider[moyen]** va renvoyer **club-internet**.*

Il est aussi possible d'utiliser la fonction array() pour créer des tableaux utilisant des indices chaine :

$provider = array("rapide" => "wanadoo", "moyen" => "club-internet", "lent" => "aol");
*La variable **$provider[moyen]** va aussi renvoyer **club-internet**.*

IV.9.1 Parcourir un tableau

a. Parcourir un tableau utilisant des entiers comme indices :

La manière la plus rapide pour extraire l'ensemble des valeurs d'un tableau, quel que soit le nombre de valeurs, est d'utiliser une boucle for() :

```
$provider = array("wanadoo", "club-internet", "aol");
for($nb =0; $provider[$nb] != false; $nb++)
{
echo $provider[$nb]."<br>";
}
```

*Bien évidemment, si le tableau **ne commence pas par l'indice 0**, il faut seulement initialiser la variable **$nb** avec l'indice de départ du tableau afin que l'ensemble des valeurs soit affichées. (ce qui donnera : for($nb = 10; $provider[$nb] != false; $nb++) // pour l'indice 10)*

Cependant, il existe une autre façon pour extraire les valeurs d'un tableau, il faut utiliser la boucle for() mais aussi la fonction count().

```
$provider = array("wanadoo", "club-internet", "aol");
$elements_table = count($provider);
for($nb =0; $nb <= $elements_table; $nb++)
{
echo $provider[$nb]."<br>";
}
```

Lorsque vous avez plusieurs indices qui ne se suivent pas, vous ne pouvez pas utiliser de boucle for(), il faut donc utiliser une boucle while() :

```
$provider = array(60 => "wanadoo", 90=> "club-internet", "aol");
$key = key($provider);
$val = current($provider);
reset($provider);
while(list ($key, $val) = each ($provider))
{
echo "L'indice $key correspond a $val<br>";
}
```

b. Parcourir un tableau utilisant des chaînes comme indices :

Lorsque l'indice est une chaine, il est nécessaire d'utiliser une boucle while pour afficher l'ensemble des valeurs et des indices du tableau :

```
$provider = array("rapide" => "wanadoo", "moyen" => "club-internet", "lent" => "aol");
$key = key($provider);
$val = current($provider);
reset($provider);
while(list ($key, $val) = each ($provider))
{
echo "L'indice chaine <b>$key</b> correspond a <b>$val</b><br>";
}
```

IV.9.2 Fonctions de tri

Il existe plusieurs fonctions permettant de trier les valeurs d'un tableau.

```
$lettres = array("b","c","a","d");
sort($lettres);
$key = key($lettres);
$val = current($lettres);
while(list ($key, $val) = each($lettres)) {
echo "L'indice $key correspond à $val<br>\n";
}
```

Ceci va ranger les valeurs du tableau dans l'ordre alphabétique en conservant les indices.

```
$lettres = array("a" =>"b", "b" => "c","c" => "a","d" => "d");
asort($lettres);
$key = key($lettres);
$val = current($lettres);
while(list ($key, $val) = each($lettres)) {
echo "L'indice $key correspond à $val<br>\n";
}
```

Ceci va ranger les valeurs, en conservant les indices.

IV.10.Les fonctions Mathématiques

- **Plusieurs nombres à ajouter :**

```
$valeur1 = "10";
$valeur2 = "11";
$valeur3 = "12";

$result = $valeur1 + $valeur2 + $valeur3;
print($result);
```

- **Multiplier :**

```
$valeur1 = "10";
$valeur2 = "11";
$valeur3 = "12";
$result = $valeur1 * $valeur2 * $valeur3;
print($result);
```

- **Diviser :**

```
$valeur1 = "10";
$valeur2 = "11";
$valeur3 = "12";
$result = $valeur1 / $valeur2 / $valeur3;
print($result);
```

- **Soustraire :**

```
$valeur1 = "10";
$valeur2 = "11";
$valeur3 = "12";
$result = $valeur1 - $valeur2 - $valeur3;
print($result);
```

IV.11.Les fonctions

IV.11.1 Définir des fonctions :

- *Fonctions simples :*

```
function bonjour()
{
echo "Bonjour !";
}

bonjour();
```

- *Fonctions complexes :*

```
function type($police, $taille)
{
echo "<font face='$police' size='$taille'>Texte Mis en forme</font>";
}

type("Arial", "2");
```

IV.12.Les instructions

IV.12.1 If, elseif et else

- **if, elseif et else sont utilisés pour vérifier quelque chose (si...autrement...sinon).**

```
$arg1 = "Flen";
$arg2 = "Flen";
$arg3 = "Flen";

if($arg1 == $arg2)    // Pas de ; a la fin
{
echo "Valeur de arg1 égale a arg2";
}
elseif($arg1 == $arg3)
{
echo "Valeur de arg1 égale a arg3";
}
else
{
echo "Arg1 n'est ni égal a arg2, ni a arg3";
}
```

- **Opérateurs de comparaison :**

< (est inférieur a)
> (supérieur a)
<= (est inférieur ou égal)
>= (est supérieur ou égal)
== (est egal a)
!= (n'est pas égal a)
AND && (et)
OR || (ou)
! (sauf)

IV.12.2 switch : le "choix parmi"

Exemple:

```
<?php
import_request_variables("P","recu_");
define("premier",3);
switch (($recu_niemeJour + premier -1)% 7)
{
case 0 :
$res = "lundi";
break;
case 1 :
$res = "mardi";
break;
case 2 :
$res = "mercredi";
break;
.....
case 6 :
$res = "dimanche";
break;
default :
$res = "erreur";
break;
}
print("le $recu_niemeJour-i&egrave;me de l'ann&eacute;e 2004 est un $res");
?>
}
```

EXECUTION pour 23 :
le 23-ième de l'année 2004 est un vendredi

• **switch** :

```
o switch (expression) {
case valeur1 :
  instructions
break;
case valeur2 :
  instructions
break;
...
case valeur7 :
case valeur8 :
  instructions
break;
default :
  instructions
break;
}
```

o l'expression est de type int, float, string
o Elle est évaluée puis le programme exécute alors les instructions à partir de la valeur correspondante jusqu'à rencontrer le mot break
o si aucune valeur correspondante est proposée, alors le programme commence en default.
o remarquons que plusieurs valeurs peuvent être regroupées comme ci-dessus.

• **break**

permet de sortir d'une instruction switch, while, for, foreach

IV.12.3 boucles while et do-while

```
<?php
import_request_variables("P","recu_");
$i = 0;
do {
print("\$i = $i<br>");
$i += 1;
while ($i < $recu_nombre);
$i = 0;
print("puis<br>");
while ($i < $recu_nombre)
{
print("\$i = $i<br>");
$i++;
}
?>
```

• EXECUTION pour 4 :
$i = 0
$i = 1
$i = 2
$i = 3
puis
$i = 0
$i = 1
$i = 2
$i = 3

do
instructionCorpsDeBoucle
while (test) ;

avec test = expr. booléenne
  o exécution de l'instruction,
  o évaluation du test,
si sa valeur est true,
alors exécution de l'instruction,
  o évaluation du test,
si sa valeur est true,
alors exécution de l'instruction,
o ....
o arrêt de la boucle quand la valeur du test devient false.

while (test)
instructionCorpsDeBoucle

o évaluation du test,
si sa valeur est true,
alors exécution de l'instruction,
o évaluation du test,
si sa valeur est true,
alors exécution de l'instruction,
o ....

o arrêt de la boucle quand la valeur du test devient false.
• Dans les 2 formes, l'instruction dite corps de la boucle est une instruction simple, un bloc ou une instruction alternative ou une boucle, ...

IV.12.4 boucle for

```
<?php
import_request_variables("P","recu_");
for ($nombre=1; $nombre<=$recu_entierMax; $nombre++)
{
$divisible = FALSE;
for ($diviseur=floor($nombre/2);
!$divisible && ($diviseur>1);
$diviseur--)
$divisible = ($nombre % $diviseur == 0);
if (! $divisible)
print("$nombre est premier<br>");
}
?>
```

?>

• EXECUTION pour 14 :

```
1 est
premier
2 est
premier
3 est
premier
5 est
premier
7 est
premier
11 est
premier
13 est
premier
```

Syntaxe

for (initialisation ; test ; nouvelle_itération)
instruction_corps_de_boucle

o l'instruction d'initialisation peut être vide
o test = expr. bouléenne
o nouvelle_itération = instruction qui prépare l'itération suivante (peut être vide)
o la boucle for fonctionne comme

```
initialisation;
while (test) {
instruction_corps_de_boucle
nouvelle_itération
}
```

• **continue**
passe directement à l'itération suivante de la boucle

IV.12.5 array "associatif"

```
<?php
$jourSemaine = array("lundi" => "travail",
"mardi" => "travail", "mercredi" => "travail",
"jeudi" => "travail", "vendredi" => "travail",
"samedi" => "week-end", "dimanche" => "week-end");
printf("le samedi est un jour de "
.$jourSemaine["samedi"]."<br>");
foreach ($jourSemaine as $jour => $sorte)
print("$jour : $sorte <br>");
$compte = 0;
foreach ($jourSemaine as $sorte)
if ($sorte == "travail")
$compte++;
printf(" il y a $compte jours de travail sur ".count($jourSemaine));
?>
```

• EXECUTION

```
le samedi est un jour de week-end
lundi : travail
mardi : travail
mercredi : travail
jeudi : travail
vendredi : travail
samedi
```

Un array "PHP" est un ensemble ordonné d'éléments accessible par un index (ordered map) :

- o c'est une association de paire (clé, valeur) : la clé permet d'accéder à l'élément
- o la clé est soit un entier soit une chaîne
- o la valeur peut être de n'importe quel type
- o en PHP4, l'array peut être utilisé comme :
  - tableau classique,
  - liste,
  - pile, file, ensemble,
  - table accessible par clé, dictionnaire.

• array(cle0 => valeur0, cle1 => valeur1, ...)
crée un tableau avec les éléments indiquées dans l'ordre
les éléments du tableau sont respectivement les paires (clé, valeur) spécifiées
le premier élément a la clé cle0 et la valeur0
le second élément a la clé cle1 et la valeur1
.....
• les éléments se manipulent ainsi : $tab[clé] où clé est un entier ou un string
• **foreach**
o PHP4 !
o foreach(array_expression as $key => $value)
réalise une boucle où les variables $key et $value prennent successivement les clés et valeurs des éléments du tableau.

o foreach(array_expression as $value)
réalise une boucle où la variable $value prend successivement les valeurs des éléments du tableau

V. eme Chapitre
MYSQL

V.1. Création de la base de données

Pour exploiter une base de données, la première chose à faire est évidemment de créer cette base. Il existe pour cela un outil très pratique: ***PHPMYADMIN.***

Dans une base de données Mysql les informations sont stockées dans des tables.
Créons une table avec trois champs: Nom, prenom et email.
Ils sont tous de type **Varchar** de longueur 50.

V.2. Connexion à Mysql

Pour se connecter à Mysql par l'intermédiaire de PHP, il faut utiliser la fonction **mysql_connect()**.

Syntaxe de la fonction :
int mysql_connect(string [hostname [:port]], string [username] , string [password]);
Exemple :
$host = "127.0.0.1";
$user = "root";
$bdd = "essai";
mysql_connect($host, $user,$password) or die("erreur de connexion au serveur");
mysql_select_db($bdd) or die("erreur de connexion a la base de donnees");

V.3. Exécution d'une requête SQL

Avant tout, il faut se connecter**.** Pour exécuter une requête SQL, il faut utiliser la fonction **mysql_db_query()**.

Syntaxe de la fonction :
int mysql_db_query(string database, string query, int [link_identifier]);

Exemple : Insertion de données dans la table list.
I) *$mysql_connexion = mysql_connect("localhost", "root", "motdepasse");*
$requete = "INSERT INTO list (nom, prenom, email) VALUES ('Benahmed', 'farid', 'smoumou@hotmail.com)";
$requete2 = mysql_db_query("bdd", $requete);

V.4. Enregistrement dans une table Mysql

Considérons une table valeurs qui se trouve dans une base essai.
- Structure de cette table *valeurs* :

CREATE TABLE valeurs (
***num** int(11) DEFAULT '0' NOT NULL auto_increment,*
***nom** blob NOT NULL,*

```
prenom blob NOT NULL,
PRIMARY KEY (num));
```

Le script PHP :

```
<?php
// Vérification des champs nom et prenom (si il ne sont pas vides ?, ces champs ayant été saisis dans un formulaire)
if($nom != "" && $prenom != "")
{
// Connexion a Mysql (changer l'host, le login et le mot de passe SVP)
$mysql_link = mysql_connect("localhost","root","");

// Vérification de la validité de la connexion MYSQL
if($mysql_link)
{
// Requete d'insertion MYSQL
$requete = "INSERT INTO valeurs (nom,prenom) VALUES ('flen','flenou')";

// Execution de cette requete dans la base essai
$execution = mysql_db_query("essai", $requete);
echo "<font face='Verdana' size='2'>Les valeurs ont bien été enregistrées dans la table
<b>valeurs</b></font>";
}
// La connexion Mysql est indisponible
else echo "<HTML><HEAD><TITLE>Erreurs</TITLE></HEAD><BODY><font face='Verdana'
size='2'>Vous avez du faire une erreur : Ce problème se pose soit : <br>- Parce que vous n'avez pas
créer la base, ni la table MYSQL;<br>- Parce que vous n'avez pas changé le Mot de passe d'accès a
MYSQL dans ce programme (Par défaut, c'est Host : 'localhost', login : 'root', MDP : '[vide]';<br>- Soit
vous n'avez pas lancé MYSQL.</font></body></html>";
// Les champs ne sont pas tous remplis
}
else echo "<HTML><HEAD><TITLE>Erreurs</TITLE></HEAD><BODY><font face='Verdana'
size='2'>ATTENTION : Le champs nom ou prénom n'a pas été remplis correctement, veuillez
vérifier</font></body></html>";
?>
```

La table Mysql contiendra ensuite :

num	nom	prenom
1	le_nom	le_prenom

v.5. Extraction de données

Comme toujours, il faut d'abord se connecter à mysql . Pour extraire des données, nous allons utiliser la fonction **mysql_fetch_array()** ainsi qu'une simple boucle **while{}**.

Pour extraire la totalité d'une table dans un tableau :

```
$mysql_link = mysql_connect("localhost", "root", "");
$requete = "SELECT * FROM table_article";
$result = mysql_db_query("base", $requete);

echo "<table>";
while ($voir = mysql_fetch_array($result))
```

```
{
echo "<tr>
<td><font face='Verdana' size='2'>Nom : ".$voir[nom_colone]."</font></td>
<td><font face='Verdana' size='2'>Prenom : ".$voir[nom_colone2]."</font></td>
</tr>";
echo "</table>";
```

Pour extraire une partie d'une table dans un tableau : (ici, la ligne ou la colone nom = Mohamed)

```
$mysql_link = mysql_connect("localhost", "root", "");
$requete = "SELECT * FROM table_article WHERE nom='Mohamed'";
$result = mysql_db_query("bdd", $requete);

echo "<table>";
while ($voir = mysql_fetch_array($result))
{
echo "<tr>
<td><font face='Verdana' size='2'>Nom : ".$voir[nom_colone]."</font></td>
<td><font face='Verdana' size='2'>Prenom : ".$voir[nom_colone2]."</font></td>
</tr>";
}
echo "</table>";
```

Suppression de données dans la table essai:
On peut effacer des données dans une table grâce à la fonction **delete from**

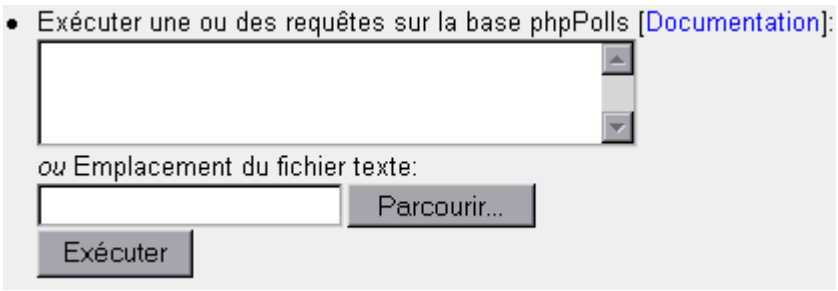
```
$mysql_link = mysql_connect("localhost", "root", "");
mysql_db_query("bdd", "delete from essai Where nom='flen'");
```

V.6. Syntaxe Mysql

V.6.1 Exécution de requêtes

Pour créer une table mysql, vous pouvez utiliser par exemple PhpMyAdmin (Le plus connu) ou bien directement exécuter la requete avec mysql_db_query() de php.

Avec PhpMyadmin, c'est très simple :



A gauche, cliquez sur le lien intitulé par le nom de votre base de données, sur free c'est votre login, puis dans le formulaire (Voir ci-contre), faites un copier coller de votre fichier contenant le code SQL ou bien tapez directement la requête que vous souhaitez exécuter. Cliquez tout simplement sur le Bouton Exécuter pour terminer.

V.6.2 Création de table

Afin de créer une table Mysql, vous devez utiliser la fonction CREATE TABLE.
Voici la syntaxe simple de l'utilisation de cette fonction. La table ci-dessous contiendra toutes les colonnes en bleu.

CREATE TABLE essai (num INT not null AUTO_INCREMENT, nom VARCHAR (100) not null , prenom VARCHAR (100) not null , adresse VARCHAR (255) not null , codepostal INT (5) not null , ville VARCHAR (100) not null , pays VARCHAR (100) not null , ***PRIMARY KEY*** (num));

Sachez que vous pouvez aussi utiliser PhpMyAdmin pour créer vos tables, ce qui vous évitera de taper la requête. (Voir ci-contre)

V.6.3 Modification d'une table

Vous évitez de créer un nouvelle fois la table que vous souhaitez modifier, il esiste la fonction ALTER TABLE qui permet d'ajouter ou de supprimer des champs d'une table.

a. Ajouter une colonne :
ALTER TABLE essai ADD email VARCHAR(100) **not null**

b. Supprimer une colonne
ALTER TABLE essai DROP email

c. Suppression de données

Pour supprimer des lignes contenues dans une table, il faut utiliser la fonction DELETE FROM.

Syntaxe : DELETE FROM nom_table WHERE champs = 'valeur'

En fait, le WHERE permet de choisir quelle ligne va être supprimée. Par exemple si le champs 'num' est AUTO_INCREMENT**, vous pourrez l'utiliser pour vos requêtes.**

Exemple pour notre table essai :
DELETE FROM essai WHERE num = '2'