Ministère de l'Enseignement Supérieur, de la Recherche Scientifique et de la Technologie
Direction Générale des Études Technologiques
☆☆☆☆☆

Institut Supérieur des Études Technologiques du Kef
Département INFORMATIQUE
☆☆☆☆☆

Matière : Multimédia et Développement Internet

Mots-clés du Support

Introduction au développement web ; les pages HTML ; le contrôle de saisi ; programmer en Java Script ; la programmation côté serveur ; la programmation côté client ; les concepts de base du HTML, du Java Script et du PHP; les pages PHP et les bases de données etc.

Support des travaux dirigés adressé aux étudiants du quatrième niveau (Réseau Informatique) ISETs

Enseignant : Mossaab BOUKHCHIM

Année Universitaire : 2006-2007

Préface

Ceci est un travail réalisé au sein du département informatique de l'ISET du Kef qui a pour objectif principal de mettre à la disposition des étudiants des ISETs orientés vers les filières « informatique de gestion » et « réseau informatique », un support d'assistance permettant aux étudiants de se familiariser pas à pas avec quelques éléments indispensables de la programmation web.

Ce document contient une suite de travaux pratiques guidés et détaillés sur le développement web et qui font partie du cours « Multimédia et développement web».

Dans ce document, nous avons opté pour des thèmes dont la répartition se présente comme suit :

- **Partie I :** HTML
  - ✓ Les concepts de base du HTML
  - ✓ Les tableaux et les frames
  - ✓ Les formulaires
  - ✓ Les fiches HTML
- **Partie II :** Java Script
  - ✓ Les concepts de base du Java Script
  - ✓ Programmer en Java Script
- **Partie III :** PHP
  - ✓ Cours PHP
  - ✓ Introduction au PHP
  - ✓ PHP et BD

Sommaire

Fiche de présentation	5
But du cours et objectifs spécifiques	6
TP 1: Les concepts de base du html	8
TP2 : Les tableaux et les frames	15
TP3 : Les formulaires	21
Note Cours 1 : Les Fiches HTML	26
Tailles de polices	27
Séparateurs	27
Pointeurs vers autres documents	28
Listes	28
Styles	29
Images	30
Tableaux <TABLE> ... </TABLE>	31
Formulaires	32
Note Cours 2 : Les Concepts de base du Java Script	35
Les bases de Java Script	35
Les opérateurs	36
Les fonctions	37
Les conditions	39
Un peu de théorie objet	41
Afficher du texte	42
Les événements	43
Les formulaires	45
L'objet String	48
L'objet Math	48
L'objet Date	48

L'objet Array	49
TP4 : TP Java Script	52
Note Cours 3 : Cours PHP	58
I- Introduction	58
1. Définition	58
2. Mise en œuvre et déploiement (EasyPHP)	58
II- Les fonctionnalités du langage	59
1. Premiers éléments du langage	59
2. Intégration de PHP dans une page HTML	59
3. Variables	60
4. Conditions et boucles	61
5. Tableaux	64
6. Passage et transmission de variables	65
7. Fonctions Prédéfinies	67
III- Utilisation d'une base de données MySql	68
1. SQL : petit récapitulatif du langage	68
2. Accéder à MYSQL via PHP	69
TP5 : Introduction au PHP	73
TP6 : PHP & BD	76

Fiche de Présentation

Multimédia et Développement Internet

Enseignant Mossaab BOUKHCHIM

Population	**:**	Étudiants du quatrième niveau (Réseau Informatique) ISETs.
Crédits	**:**	67,5 heures par semestre
Volume Horaire	**:**	4,5 heures par semaine sur 15 semaines.
Pré requis	**:**	Algorithme et structure de données, Base de données.
Langue	**:**	Français

Objectifs du cours

- Présenter les principaux concepts de base des langages Html, Java Script et PHP.
- Amener les étudiants à construire de petites site selon le principe de la programmation orienté web, c'est-à-dire en s'appuyant sur la répartition des tâches entre le serveur et le client.

Évaluations

- Test N°1 d'une heure.
- Devoir surveillé d'une heure.
- Test N°2 d'une heure.
- Examen final écrit de 2 heures sur tout le programme.
- Examen TP.
- Projet.

Moyens Pédagogiques

- Support de cours papier.
- Support de cours numérique.
- Série de travaux pratiques.

But du cours et objectifs généraux

Multimédia et Développement Internet

Objectifs Généraux	Conditions de réalisation de la performance	Critères d'évaluation de la performance
Connaître l'allure d'une page HTML, sa structure générale et la démarche à suivre pour la produire.	A partir des notes de cours et des références bibliographiques, l'étudiant devrait identifier les éléments constitutifs d'une page HTML.	Aucune erreur n'est permise.
Savoir construire des tableaux et des frames, après avoir compris ces nouvelles notions.	A partir des notes de cours, des références bibliographiques et les TPs l'étudiant devrait être capable de reconnaître un tableau et une frame et de pouvoir les construire correctement.	Aucune erreur n'est permise. L'étudiant doit pouvoir construire avec succès les tableaux et les frames.
Connaître les formulaires et ses objets.	A partir des notes de cours, des références bibliographiques et les TPs l'étudiant devrait être capable de reconnaître et d'énumérer les objets d'une formulaire..	L'étudiant doit pouvoir décrire correctement un formulaire et utiliser ses objets.
Connaître et comprendre les objets, les structures de données, les événements et les opérations du Java Script.	A partir des notes de cours et les TPs l'étudiant devrait être capable de connaître les concepts de base du Java Script.	L'étudiant doit être capable d'écrire correctement une balise Java Script.
Etudier le langage PHP.	A partir des notes de cours et les TPs l'étudiant devrait être capable de connaître les différents concepts du PHP.	L'étudiant doit être capable d'écrire correctement une page web utilisant des balises PHP.
Connaître le principe de manipulation des bases de données par le php.	A partir des notes de cours et les Tps l'étudiant devrait être capable de connaître l'utilité de base de données. En outre, il devrait être capable d'utiliser le PHP pour supprimer, modifier, consulter et ajouter des données de la base.	L'étudiant doit être capable de se connecter à une base de données pour la gérer.

TP1

Concepts de base du HTML

Objectifs

- *Se familiariser avec l'environnement du développement web*
- *Connaître l'allure d'une page HTML, sa structure générale et la démarche à suivre pour la produire.*

Enoncé du TP :

TP 1

Concepts de base du HTML

Avant propos

Première chose importante à savoir sur **HTML** est la signification de ces quatre initiales : **H**yper **T**ext **M**arkUp **L**anguage.

HTML est un langage de description (et non pas un langage de programmation) qui va nous permettre de décrire l'aspect d'un document, d'y inclure des informations variées (textes, images, sons, animations etc.) et d'établir des relations cohérentes entre ces informations grâce aux liens hypertextes.

Les avantages du langage HTML sont nombreux :

- ☞peux coûteux en effet un simple éditeur de texte suffit à écrire ses premiers documents HTML
- ☞relativement facile à aborder,
- ☞il représente en outre un bon moyen de dépasser les problèmes de compatibilité entre les systèmes et les formats informatiques différents.

La description d'un document HTML passe par l'utilisation de ***BALISES*** (ou "TAGS" en anglais). Une balise est délimitée par les signes "<" et ">" entre lesquels figure le nom de la balise. Par exemple, la balise de retour à la ligne est
 La plupart du temps, on utilise une balise de début et une balise de fin, qui définissent les propriétés de l'intervalle.

Allure d'un fichier HTML ou squelette d'un fichier HTML

```
<!-- commentaire -->
<HTML>
<HEAD>
✍✍✍
</HEAD>
<BODY>
✍✍✍
</BODY>
</HTML>
```

Commentaires

Les commentaires peuvent être placés n'importe où dans un document HTML à condition de ne pas être imbriqués. Ils sont placés entre les chaînes de caractère **<!--** et **-->**.

HTML

C'est le premier marqueur ou tag que l'on trouve dans un document HTML. Tout le document qui sera écrit par la suite (texte ou marqueur) sera compris entre le tag <HTML> et </HTML>.

Exemple : `<HTML> Document à écrire </HTML>`

En-tête <HEAD>

L'en-tête du document est réservé aux méta-informations (les informations relatives au document) comme son titre. Ces méta-informations doivent être placées entre les tags ***<HEAD>*** de début et ***</HEAD>*** de fin.

Dans l'en-tête, on trouvera les commandes suivantes :

- ☞ **TITLE** (la seule véritablement essentielle- voir ci-dessous)
- ☞ **ISINDEX** indique au logiciel client que le document est un écran permettant une saisie pour interroger un programme externe (Programme CGI), (Servait à l'origine du HTML aux browser à envoyer de l'info.-Les formulaires ont simplifiés tout),
- ☞ **BASE** permet d'indiquer une adresse de base qui complétera tous les chemins relatifs,
- ☞ **LINK** (Indique un lien entre le présent document et un autre) ; très rarement utilisé.
- ☞ **NEXTID** (Indique le document suivant – considéré actuellement comme obsolète-)
- ☞ **META** donne des informations sur le document au serveur. Ces informations sont généralement :

  <META NAME=...> le nom,

  <META AUTHOR=....> le nom de l'auteur,

  <META CONTENT=….> ,

  <META NEXTID=….>un identifiant.

- ☞**<BASEFONT SIZE = n>** ou n prend les valeurs 1 à 7.Cette commande fixe la taille de la police de caractères (par défaut : 3) 1, petits caractères … 7, gros caractères.
- ☞ **PROMPT=texte**, permet d'afficher le texte comme message pour ***<ISINDEX>.***

Titre <TITLE>

<TITLE> Titre ***</TITLE>***

Chaque document HTML, pour être correct, doit disposer d'un titre. Le titre apparaît le plus souvent dans la barre de titre du navigateur Web. Son but est de pouvoir identifier le document dans un ensemble plus large. Le titre d'un document HTML sert aussi lorsqu'un navigateur World Wide Web dispose d'une fonction "hotlist" ou "bookmark" pour fournir un accès rapide à vos documents favoris. Le titre est limité à ***64 caractères*** y compris les espaces et ne doit normalement contenir que les 128 premiers caractères ASCII (pas de caractères accentués)

Corps du document <BODY>

Le corps du document, ce qui sera effectivement affiché par le navigateur Web, est balisé par les commandes :

<BODY> et ***</BODY>***

Exercices

Exercice 1 :

Commenter la présentation dans un navigateur Web du document suivant :

```
<HTML>
<HEAD>
<TITLE>essai n° 1</TITLE>
</HEAD>
<BODY>
Ceci est le premier essai
</BODY>
</HTML>
```

Exercice 2 (taille de la police) :

Il existe 7 tailles de polices. La commande est **<*FONT SIZE = n*> … </*FONT*>** ou n prend les valeurs de 1 à 7.

1, petits caractères

7, gros caractères.

La police par défaut possède la taille 3.

La taille de la police peut être fixée dans l'en-tête du document par la commande :

<*BASEFONT SIZE = n*> ou n prend les valeurs 1 à 7.

Ecrire le document ayant la présentation suivante :

Exercice 3 (Séparateurs)

Ecrire puis interpréter le document suivant :

```
<HTML>
<HEAD>
<TITLE>essai n° 3</TITLE>
</HEAD>
<BODY>
<H4>
HR<HR>
HR WIDTH="75%"
<HR WIDTH="75%">
HR WIDTH="25%"
<HR WIDTH="25%">
HR WIDTH="50%" & SIZE=5
<HR WIDTH="50%" SIZE=5>
HR WIDTH="50%" & SIZE=20
<HR WIDTH="50%" SIZE=20>
HR WIDTH="50%" & SIZE=5 ALIGN=LEFT
HR WIDTH="50%" & SIZE=5 <HR WIDTH="50%" SIZE=5
ALIGN=LEFT>
HR WIDTH="50%" & SIZE=5 ALIGN=LEFT NOSHADE
HR WIDTH="50%" & SIZE=5 <HR WIDTH="50%" SIZE=5
ALIGN=LEFT NOSHADE>
</H4>
</BODY>
</HTML>
```

Remarque :

***
*** Cette commande impose la coupure d'une ligne de texte en rejetant ce qui suit à la ligne suivante.

<HR> pour une ligne horizontale de séparation :

WIDTH fait varier la largeur de la ligne en % soit en pixel (valeur par défaut 100%)

SIZE fait varier l'épaisseur de la ligne en pixel (valeur par défaut 1)

ALIGN fait un alignement de la ligne suivant les 3 possibilités :

CENTER par rapport au centre de la fenêtre,

LEFT par rapport à la gauche de la fenêtre,

RIGHT par rapport à la droite de la fenêtre.

NOSHADE Lorsqu'il est présent dans le marqueur <HR> l'effet est une ligne pleine sans ombrage.

Exercice 4 (Pointeur vers autres documents) :

Ecrire deux pages html. La première présente notre établissement. La seconde contient une description de notre département.

Remarque :

Grâce au langage HTML, vous pouvez créer un hyperlien vers un document local, un document distant...

L'hyperlien est créé par l'utilisation du tag ***<A>***.

La syntaxe complète est ***<A HREF="document">texte</A>***

Où **HREF** est un raccourci pour "Hypertexte référence", ***document*** désigne le document vers lequel on pointe (il existe deux chemin d'accès relative et absolue) et ***texte*** représente le texte qui sera affiché pour représenter le lien hypertexte.

Exercice 5 (Listes) :

Commentez les documents suivants :

```
<HTML>
<HEAD>
<TITLE>Liste non ordonnée</TITLE>
</HEAD>
<BODY>
<H1>
<ol>
<lh>Voitures allemandes
<li>BMW
<li>MERCEDES
<li>PORSCHE
</ol>
</H1>
<BODY>
<HTML>
```

```
<HTML>
<HEAD>
<TITLE>Liste non ordonnée</TITLE>
</HEAD>
<BODY>
<H1>
<ul>
<lh>Voiture allemandes
<li>BMW
<li>MERCEDES
<li>PORSCHE
</ul>
</H1>
<BODY>
<HTML>
```

```
<HTML>
<HEAD>
<TITLE>Liste descriptive</TITLE>
</HEAD>
<BODY>
<H2>
<DL>
<lh>Voitures allemandes
<DT>BMW
<DD>série 3 5 7 8
<DT>MERCEDES
<DD> classe C E
<DT>PORSCHE
<DD> 956 et 911
</DL>
</H2>
<BODY>
<HTML>
```

```
<HTML>
<HEAD>
<TITLE>Liste menu</TITLE>
</HEAD>
<BODY>
<H2>
<MENU>
<lh>Voitures allemandes
<LI>BMW
<LI>MERCEDES
<LI>PORSCHE
</MENU>
</H2>
<BODY>
<HTML>
```

```
<HTML>
<HEAD>
<TITLE>html\nv_html</TITLE>
</HEAD>
<OL>
<LI> BMW
<UL>
<LI> série 3
<LI> série 5
</UL>
<LI> MERCEDES
<DL>
<DT>classe C
<DD>180, 200, 220.
<DT> classe E
<DD>250, 300
</DL>
<LI> FIN
</OL>
</BODY>
</HTML>
```

Exercice 6 (Styles) :

Commentez les documents suivants :

```
<HTML>
<HEAD>
<TITLE>Les commandes de styles physiques</TITLE>
</HEAD>
<BODY>
<I> met le texte en italique.</I><BR>
<B> met le texte en gras.</B><BR>
<TT> pour l'utilisation d'une police
à chasse fixe (encombrement des caractères constant).</TT><BR>
<U> souligne le texte.</U><BR>
<S> barre le texte.</S><BR>
normal<SUB> indice. </SUB><BR>
normal<SUP> exposant. </SUP><BR>
<BODY>
<HTML>
```

```
<HTML>
<HEAD>
<TITLE>Centrage de texte</TITLE>
</HEAD>
<BODY>
<CENTER>
<H1>I.U.T. d'Amiens</H1><BR>
<H3>Promotion 1998/1999</H3><BR>
Cours HTML
</CENTER>
<BODY>
<HTML>
```

```
<HTML>
<HEAD>
<TITLE>texte préformaté</TITLE>
</HEAD>
<BODY>
enseignements :<BR><BR>
<PRE>
MATIERES PROFESSEUR NB d'heures
-------------------------------------
Visual Basic B. Mahric 40
Réseaux E. Brassart 32
C++ L.Delahoche 32
</PRE>
<BODY>
<HTML>
```

Remarques :

L'utilisation de différents styles de polices de caractères permet de varier la présentation d'un texte. En HTML, on peut utiliser des styles logiques où le nom du style indique la nature du fragment de texte à écrire dans ce style ou des styles physiques où le nom du style indique explicitement le style de police que l'on souhaite voir utiliser.

Styles de caractères :

Les commandes de styles sont les suivantes :

<*I*> texte **</*I*>** met le texte en italique.

<*B*> texte **</B>** met le texte en gras.

<*TT*> texte **</*TT*>** pour l'utilisation d'une police à chasse fixe (encombrement des caractères constant).

<*U*> texte **</*U*>** souligne le texte.

<*S*> texte **</*S*>** barre le texte.

<*SUB*> texte **</*SUB*>** indice.

<*SUP*> texte **</*SUP*>** exposant.

CENTRAGE D'UN TEXTE <CENTER>…</CENTER>

La commande **<*CENTER*>** ... **</*CENTER*>** permet de centrer toutes les lignes d'un texte.

TEXTE PRÉ-FORMATÉ <PRE> ... </PRE>

Les espaces (plusieurs à la suite), tabulations, retour chariot n'ont pas de valeur en HTML.

La commande **<*PRE*> ... </*PRE*>** est utilisée pour inclure un texte pré-formaté dans un document HTML.

Les espacements, tabulations et retour chariot gardent alors leur sens initial.

Attention : les commandes HTML existant dans le texte pré-formaté seront interprétées.

L'option WIDTH=n peut-être utilisée pour limiter la longueur de la ligne.

TP 2

Les Tableaux et les Frames

Objectifs

- *Se familiariser avec l'environnement du développement web*
- *Savoir construire des tableaux et des frames, après avoir compris ces nouvelles notions.*

Enoncé du TP :

TP 2

Les Tableaux et les Frames

Avant propos

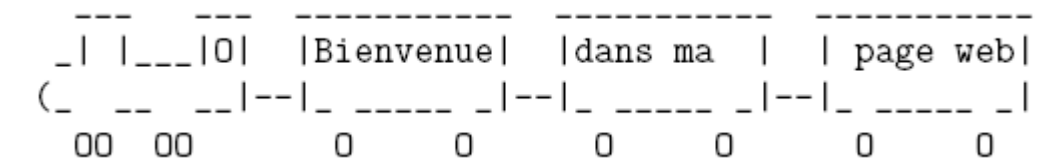
Comme les listes, les tableaux utilisent plusieurs balises imbriquées :

- Un tableau commence toujours par une balise <TABLE> et se termine par une balise </TABLE>.
- Les balises <TR> et </TR> signalent le début et la fin d'une ligne (Cette balise crée une ligne dans un tableau). ***<tr> = Table Row***
- Les balises <TD> et </TD> indiquent le début et la fin d'une cellule (Cette balise crée des cellules individuelles dans les lignes du tableau). **<td> = Table Data**
- Les balises <TD> sont toujours contenues dans les balises <TR>, elles-mêmes à l'intérieur des balises <TABLE>.

Exercices

Exercice 1 :

Utiliser la balise <PRE> pour faire afficher le logo suivant :



Exercice 2

1. Tapez le texte suivant :

```
<html> <head> <title> table 1 </title></head>
<body>
<table>
<tr> <td>Nom</td><td>Prenom</td><td>Groupe</td></tr>
<tr> <td>Nom 1</td><td>Prenom 1</td><td>Groupe 1</td> </tr>
<tr><td>Nom 2</td> <td>Prenom 2</td><td>Groupe 2</td></tr>
<tr> <td>Nom 3</td> <td>Prenom 3</td> <td>Groupe 3</td></tr>
<tr><td>Nom 4</td><td>Prenom 4</td> <td>Groupe 4</td></tr>
<tr><td>Nom 5</td><td>Prenom 5</td> <td>Groupe 5</td></tr>
</table>
</body>
</html>
```

2. Remplacez <td>Nom</td><td>Prenom</td><td>Groupe</td> par <th>Nom</th> <th>Prenom</th> <th>Groupe</th> et visualisez le résultat.
3. Remplacez <table> par <table border=1> et visualisez le résultat.
4. Remplacez <table> par <table border=20 cellspacing=10 bgcolor="white"> et visualisez le résultat

5. Remplacez <table> par <table border=20 cellspacing=10 bgcolor="white" cellpadding=10> et visualisez le résultat
6. Modifiez le texte entre <body> et </body> comme suit et visualisez le résultat.

```
<table border=1>
<tr><th>Nom</th><th>Prenom</th><th>Groupe</th></tr>
<tr><td>Nom 1</td><td>Prenom 1</td><td rowspan=2>Groupe 1</td></tr>
<tr><td>Nom 2</td><td>Prenom 2</td></tr>
<tr><td>Nom 3</td><td>Prenom 3</td><td rowspan=3>Groupe 2</td></tr>
<tr><td>Nom 4</td><td>Prenom 4</td></tr>
<tr><td>Nom 5</td><td>Prenom 5</td></tr>
</table>
```

7. Remplacez <td rowspan=2> par <td rowspan=2 valign="top"> et <td rowspan=3> par <td rowspan=3 valign="bottom"> et visualisez le resultat.
8. Remplacez <tr><th>Nom</th><th>Prenom</th><th>Groupe</th></tr> par <tr bgcolor="yellow"><th bgcolor= "red">Nom</th><th>Prenom</th> <th>Groupe</th> </tr> et visualisez le résultat.
9. Remplacez <td rowspan=2 valign="top"> par <td rowspan=2 bgcolor="blue"> et <td rowspan=3 valign="bottom"> par <td rowspan=3 bgcolor="green"> et visualisez le résultat.
10. Modifiez le texte entre <body> et </body> comme suit et visualisez le résultat. <table border=1><tr><th>Nom</th><td>Nom1</td><td>Nom2</td><td>Nom3</td><td>Nom 4</td><td>Nom5</td></tr><tr><th>Prenom</th><td>Prenom 1</td><td>Prenom 2</td><td>Prenom3</td><td>Prenom4</td><td>Prenom5</td></tr><tr><th>Groupe</th><td colspan=2>Groupe 1</td><td colspan=3>Groupe2</td></tr></table>
11. Remplacez <td colspan=2> par <td colspan=2 align="right"> et <td colspan=3> par <td colspan=3 align="center"> et visualisez le résultat.
12. Ajoutez après <table border=1> ceci : <caption> Liste des groupes</caption> et visualisez le résultat.
13. Remplacez <table border=1> par <table border=1 width="50%"> et visualisez le résultat.
14. Remplacez <table border=1 width="50%"> par <table border=1 width="50%" height="80%"> et visualisez le résultat.
15. Remplacez <table border=1 width="50%" height="80%"> par <table border=1 width="50%" height="80%" align="center"> et visualisez le résultat.

Exercice 3 :

Donnez les codes HTML permettant d'obtenir les tables suivantes :

table 1

A	B	C	D
E	F		G
H			I
J			K

table 2

table 1 A B C D E F G H I J K	1. Fichiers HTML 2. Fichiers Images 1. Natures mortes 2. Paysages 1. Ete 2. Hiver 3. Personnages 3. Fichiers Animations 1. Dim2 2. Dim3 4. Fichiers Sons

table 3

1					
2		3 5		4	
6		7			
8		9			
10		11		12	
13	14	15	16	17	18
19 23	20	24	21	25	22
26					27
28	29	30	31	32	33

Exercice 4 :

Construire la page entete.html en incluant l'image gif et un fond d'écran image.

- Insérer l'image (ANIM_BOUSOLE.GIF dans le dossier images) ;
- puis ajouter le fond (FOND_GRIS.JPG toujours dans images).

Exercice 5 :

Construire la page tablo.html (dans web) en fonction du modèle suivant :

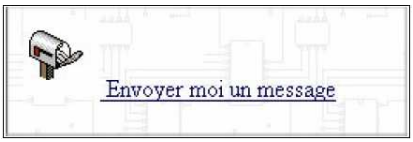
- Insérer l'image (ANIM_NOMBRE.GIF dans le dossier images) ;
- puis ajouter le fond de votre choix (toujours stocké dans images).

<table>
<tr><th colspan="4">Titre du tableau</th></tr>
<tr><td colspan="4">Regroupement horizontal de 4 cellules</td></tr>
<tr><td rowspan="3">Regroupement vertical de 4 cellules</td><td colspan="2">Deux horizontales</td><td>Deux verticales</td></tr>
<tr><td>gauche</td><td>Ca vit juste en dessous</td><td rowspan="2"></td></tr>
<tr><td rowspan="2">deux verticales</td><td rowspan="2">  </td></tr>
<tr><td>droite</td><td colspan="2">Encore deux horizontales</td></tr>
</table>

Exercice 6 :

Construire la page adresse.html (dans files) :

- Insérer l'image (ANIM_EMAIL06.GIF dans le dossier images), vous établirez un lien sur votre propre adresse e_mail (si vous en avez une ou vers une adresse fictive) ;
- Puis ajouter le fond (FOND_CIRCUIT.GIF toujours dans images) ;
- Vous établirez un lien messagerie à votre adresse sur le texte.



Exercice 7 :

Construire la page retour.html (dans files) :

- Insérer l'image (ANIM_INFO1.GIF dans le dossier image).
- Etablir un lien permettant l'appel du fichier exo1.html à partir du texte 'vers EXO1', puis établir le lien inverse vers retour.html à partir du texte 'vers retours'.

Exercice 8 :

Construire la page cadre.html (dans web) :

- La partie haute portera le nom de ‘ENTETE’ ;
- La partie centrale de gauche portera le nom de ‘CORPS’ et celle de droite ‘RETOUR’
- La partie basse portera le nom de ‘ADRESSE’ ;
- Les divisions horizontales seront de 30%,45%, et 25% ;
- Les divisions verticales seront de 80% et 20%.

TP 3

Les Formulaires

Objectifs

- *Se familiariser avec l'environnement du développement web*
- *Connaître les formulaires et ses objets.*

Enoncé du TP :

TP 3

Les Formulaires

Avant propos

La commande Form

Des formulaires peuvent être préparés afin de saisir des données et les traiter au niveau du serveur.

Pour rédiger un questionnaire, il faut:

- Établir une zone d'édition (appelée FORM) en utilisant les commandes <FORM></FORM>.
- Définir la méthode à employer pour transmettre au serveur l'information recueillie dans les champs du formulaire.
- Identifier l'emplacement et le nom du programme qui devra traiter l'information recueillie.
- Fournir, s'il y a lieu, les arguments au programme de traitement des données.

Toute cette information se retrouve dans la commande suivante:

<FORM METHOD="POST" ACTION="/cgi-bin/questionnaire.cmd?xxx">

La méthode utilisée est ***POST***, le programme de traitement se nomme questionnaire.cmd et se retrouve dans le dossier cgi-bin, un seul argument est fourni au programme soit xxx.

La commande INPUT

- ***La commande INPUT="TEXT"***

Parmi les choix disponibles en HTML, un des types d'entrée de données est le champ ***input type="text"***. Dans ce cas, il faut inscrire le type de champ, le nom du champ et ses dimensions à l'écran.

Ainsi, dans la question ci-dessous, le code à utiliser pour entrer le nom de la personne est :

```
NOM: <input type="text" name="name" size=30>.
```

Le type ***text*** est un champ ou l'usager entre de l'information sur son clavier, dans une zone définie à l'écran.

- ***La commande INPUT="RADIO "***

Un autre type de champ est le type ***"input type=radio"*** qui permet d'afficher une série de boutons radio comme choix de réponses.

Il suffit d'abord de poser la question puis de positionner la commande suivante:

```
<input type="radio" name="info" value="OUI">OUI
<input type="radio" name="info" value="NON">NON
```

- ***La commande INPUT=CHECKBOX***

La commande ***input Type=checkbox*** permet d'afficher une liste où plusieurs choix sont possibles en même temps. La commande s'écrit:

```
Texte <input name="nom_du_champ" TYPE=checkbox VALUE="texte"><BR>
ou
<input name="nom_du_champ" TYPE=checkbox VALUE="texte"> Texte <BR>
```

- ***La commande SELECT NAME et OPTION***

La commande ***select name*** et ***Option*** permet d'afficher une liste ou un seul choix est possible et qui s'affichent sous la forme d'un menu ***"pop-up"***. La commande s'écrit:

```
<select name><option selected>option1<option> option2<option>
option3</select>
```

- ***La commande TEXTAREA***

La commande HTML utilisée est:

<TEXTAREA name="nom_du_champ?" rows=n cols=m></TEXTAREA>

Dans ce type, on spécifie d'abord le type soit textarea, ensuite le nom de la rubrique soit name=" *nom_du_champ*" puis les paramètres d'affichage de la zone de dialogue en rangées (n) et en colonnes (m).

```
<TEXTAREA name="zone_libre" rows=5 cols=40></TEXTAREA>
```

- ***La commande INPUT=submit***

Les formulaires doivent être complétés avant fermeture par une commande permettant d'envoyer le contenu des champs remplis au serveur HTTP.

Cette commande s'écrit :

```
<input type="submit" value="Soumettre">
```

On ajoute également une deuxième commande qui permet à l'usager de reprendre le questionnaire s'il s'est trompé.La commande en question s'écrit:

```
<input type="reset" value="effacer et recommencer">
```

Exercices

Exercice 1 :

Écrire un formulaire « calculatrice » : 2 cases pour la saisie des opérandes, un groupe de 4 cases à cocher (ou une liste déroulante) pour le choix de l'opération, et affichage du résultat de l'opération.

Exercice 2 :

1. Editer une page *html* qui permet aux utilisateurs de soumettre par courrier électronique une fiche de renseignement. La page inclut un formulaire *html* qui présente la fiche de renseignement demandée. Cette fiche est composée des éléments suivants :

Champ.	*Domaine de valeurs*
Civilité	{Mme. Mlle. M.}
Nom	Chaîne de caractères alphabétiques
Prénom	Chaîne de caractères alphabétiques
Date de naissance	jj/mm/aaaa
Adresse	Une adresse (voir ci-après)
Téléphone	10 chiffres.
Adresse e-mail	nom.prenon@domain.tld
Site web	URL

Une adresse est composée des champs suivants :

Champ	*Domaine de valeurs*
Numéro	Un nombre
Rue	Chaîne alphanumérique
Code postale	Cinq chiffres
Ville	Chaîne de caractères

a. La fiche est représentée par un formulaire HTML. Pour chacun des éléments cités ci-dessus proposer les éléments de formulaire les plus adaptés à son implémentation.

b. Editer une page HTML qui représente la fiche décrite ci-dessus.

2. Editer une page Html qui représente *un livre de visiteur*. Un livre de visiteurs permet aux visiteurs d'un site de laisser leurs coordonnées ainsi que leurs remarques sur l'organisation et/ou le contenu du site.

3. Une société spécialisée dans la vente des ordinateurs sur mesure voudrait offrir à ses clients la possibilité de demander via le serveur Web la configuration des ordinateurs à acheter. Editer une page Web qui permet d'envoyer par courrier électronique la configuration personnalisée d'un ordinateur. Un client a la possibilité de changer les attributs suivants d'un ordinateur : Le type de processeur, la taille de la mémoire vive, la taille du disque dur, la taille de l'écran, le type du boîtier, le type du modem (si demandé), et le type de la carte réseaux (si demandée). Le choix de chaque attribut se fait parmi un ensemble de choix possibles prédéfinis par la société.

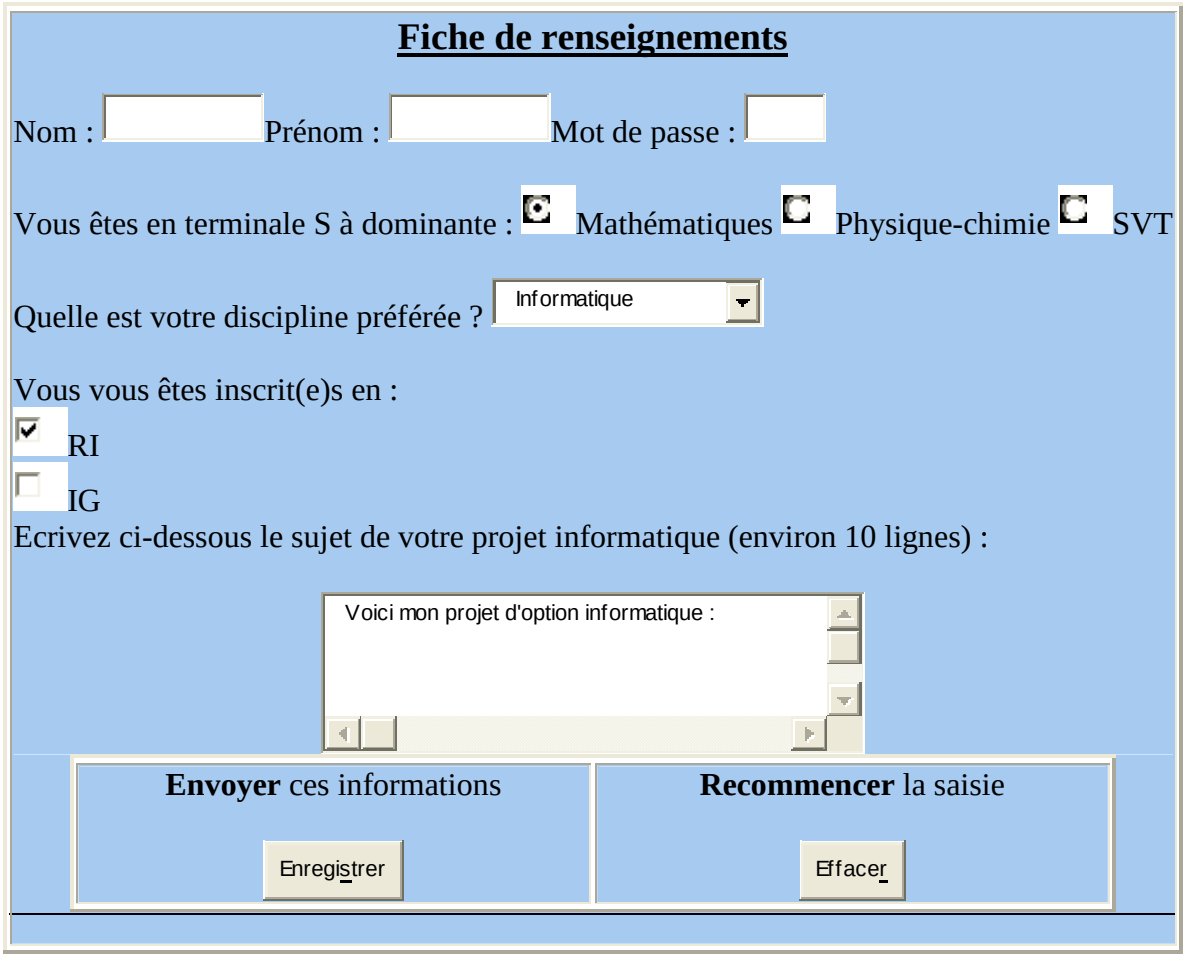
Exercice 3 :

Ecrire le document HTML qui réalise les indications suivant :

- La page doit contenir les champs suivants : *Nom, Prénom, adresse, sexe, Code postal, Téléphone fixe, Téléphone mobile, Types d'activités, Remarques et Contact.*
- La taille de tous les champs (attribut size) est fixée à 40 caractères, exceptés pour les champs *Code postal* (5), *Téléphone fixe* (20) et *Téléphone mobile* (20).
- Le nombre maximum de caractères (attribut maxlength) est fixé à 50, exceptés pour les champs *Code postal* (5), *Téléphone fixe* (20) et *Téléphone mobile* (20).
- Les dimensions du champ *Remarques* sont de 70 colonnes et de 4 lignes.
- Le champ *Types d'activités* est un contrôle de type Select.
- Les champs *Contact* et *sexe* sont des contrôles de type Radio.
- Le contenu du formulaire sera envoyé à votre adresse électronique votre-nom@site.fr. Pour ce faire, les attributs de l'élément FORM seront method="post", action="mailto:votre-nom@site.fr" et enctype="text/plain".

Exercice 4 :

Récupérer le code du formulaire ci-dessous. Les données du formulaire sont expédiées à l'adresse *Boukchim_mossaab@yahoo.fr*



Note Cours 1

Fiches HTML

Objectifs

- *Connaître l'allure d'une page HTML, sa structure générale et la démarche à suivre pour la produire.*
- *Savoir construire des tableaux et des frames, après avoir compris ces nouvelles notions.*
- *Connaître les formulaires et ses objets.*
- *Se familiariser avec l'environnement du développement web*

Enoncé du Cours :

Note Cours 1

Fiches HTML

Allure d'un fichier HTML ou squelette d'un fichier HTML

```
<!-- commentaire -->
<HTML>
    <HEAD>
        ✍✍✍
    </HEAD>
    <BODY>
        ✍✍✍
    <BODY>
</HTML>
```

Commentaires

Les commentaires peuvent être placés n'importe où dans un document HTML à condition de ne pas être imbriqués. Ils sont placés entre les chaînes de caractère **<!--** et **-->**.

Exemple :
<!-- Ceci est un commentaire **-->**

HTML

C'est le premier marqueur ou tag que l'on trouve dans un document HTML. Tout le document qui sera écrit par la suite (texte ou marqueur) sera compris entre le tag <HTML> et </HTML>.

En-tête **<HEAD>**

L'en-tête du document est réservée aux méta-informations (les informations relatives au document) comme son titre. Ces méta-informations doivent être placées entre les tags ***<HEAD>*** de début et ***</HEAD>*** de fin. Même si l'en-tête est optionnelle, il est important de l'inclure dans un document pour éviter que le logiciel serveur n'ait à balayer tout le document pour y retrouver l'information recherchée.

Dans l'en-tête, on trouvera les 6 commandes suivantes :

- ☞ **TITLE** (la seule véritablement essentielle- voir ci-dessous)
- ☞ **ISINDEX** indique au logiciel client que le document est un écran permettant une saisie pour interroger un programme externe (Programme CGI), (Servait à l'origine du HTML aux browser à envoyer de l'info.-Les formulaires ont simplifiés tout),
- ☞ **BASE** permet d'indiquer une adresse de base qui complétera tous les chemins relatifs,
- ☞ **LINK** (Indique un lien entre le présent document et un autre) ; très rarement utilisé. ☞
- **NEXTID** (Indique le document suivant – considéré actuellement comme obsolète-)
- ☞ **META** donne des informations sur le document au serveur. Ces informations sont généralement :
  - **<*META NAME*=...>** le nom,
  - **<*META AUTHOR*=....>** le nom de l'auteur,
  - **<*META CONTENT*=….>** ,
  - **<*META NEXTID*=….>**un identifiant.
- ☞**<BASEFONT SIZE = n>** ou n prend les valeurs 1 à 7.Cette commande fixe la taille de la police de caractères (par défaut : 3) 1, petits caractères … 7, gros caractères.
- ☞ **PROMPT=texte**, permet d'afficher le texte comme message pour **<*ISINDEX*>.**

Titre **<*TITLE*>**

<*TITLE*> Titre **</*TITLE*>**
Chaque document HTML, pour être correct, doit disposer d'un titre. Le titre apparaît le plus souvent dans la barre de titre du navigateur Web. Son but est de pouvoir identifier le document dans un ensemble plus large comme un index WAIS par exemple. Le titre d'un document HTML sert aussi lorsqu'un navigateur World Wide Web dispose d'une fonction “hotlist” ou “bookmark” pour fournir un accès rapide à vos documents favoris.
Le titre est limité à ***64 caractères*** y compris les espaces et ne doit normalement contenir que les 128 premiers caractères ASCII (pas de caractères accentués)

Corps du document **<*BODY*>**

Le corps du document, ce qui sera effectivement affiché par le navigateur Web, est balisé par les commandes :
<*BODY*> et **</*BODY*>**

Tailles de polices :

Il existe 7 tailles de polices. La commande est **<*FONT SIZE = n*> … </*FONT*>** ou n prend les valeurs de 1 à 7.
1, petits caractères
7, gros caractères.
La police par défaut possède la taille 3.
La taille de la police peut être fixée dans l'en-tête du document par la commande :
<*BASEFONT SIZE = n*> ou n prend les valeurs 1 à 7.

Les Entêtes **<*Hx*>** avec x : 1→6

Ces entêtes servent à diviser le texte en section de la même manière qu’un livre. Le langage HTML reconnaît six niveaux de d’en-tête numérotés de 1 à 6. Le niveau le plus élevé est le 1, le plus petit est le 6.

Séparateurs

Contrairement à tous les autres systèmes de traitement de texte, le retour de charriot n'a aucune valeur en HTML. C'est le navigateur Web lui même qui définira le passage à la ligne en fonction de facteurs comme la taille des polices de caractères, la largeur de la fenêtre de visualisation etc.
De même, plusieurs espaces dans un document HTML seront ramenés à un seul.
<*P*> Cette commande <P> (fin de paragraphe) termine un paragraphe et insère une ligne vide après le paragraphe.
<*BR*> Cette commande impose la coupure d'une ligne de texte en rejetant ce qui suit à la ligne suivante.
<*NOBR*>*…*</*NOBR*>Cette commande permet de mettre du texte sans retour à la ligne.
<*WBR*> Cette commande permet de forcer un retour à la ligne dans un texte encadré par **<*NOBR*>**

<*HR*> pour une ligne horizontale de séparation :

WIDTH	fait varier la largeur de la ligne en % soit en pixel (valeur par défaut 100%)	
SIZE	fait varier l'épaisseur de la ligne en pixel (valeur par défaut 1)	
ALIGN	fait un alignement de la ligne suivant les 3 possibilités :	
	CENTER	par rapport au centre de la fenêtre,
	LEFT	par rapport à la gauche de la fenêtre,
	RIGHT	par rapport à la droite de la fenêtre.
NOSHADE	Lorsqu'il est présent dans le marqueur <HR> l'effet est une ligne pleine sans ombrage.	

Le paramètre ***ALIGN*** a un effet que lorsque la longueur de la ligne est inférieure à 100 %.

Pointeurs vers autres documents

Grâce au langage HTML, vous pouvez créer un hyperlien vers un document local, un document distant ou un signet dans ces deux types de documents.

Anchor

L'hyperlien est créé par l'utilisation du tag ***<A>*** pour "anchor" ou ancre.
Les attributs possibles sont :

HREF

La syntaxe complète est ***<A HREF="document">texte</A>***
Où **HREF** est un raccourci pour "Hypertexte référence", ***document*** désigne le document vers lequel on pointe et ***texte*** représente le texte qui sera affiché pour représenter le lien hypertexte.

Chemin d'accès relatif :
Dans ce cas, relativement par rapport à l'endroit ou vous vous situez dans une arborescence, vous exprimez le chemin d'accès vers le fichier lié.

HREF= "fichier2.htm "	Dans le répertoire courant
HREF= "tempo/version1/fichier2.htm "	Dans le répertoire 2 niveaux en dessous
HREF= "../../fichier2.htm "	Dans le répertoire 2 niveaux au dessus

Chemin d'accès absolu :
Dans ce cas, depuis la racine des disques nommés, vous exprimez le chemin d'accès vers le fichier lié.

HREF= "/rep1/fichier2.htm "	Dans le répertoire rep1 de l'unité en cours
HREF= "/d:/html/fichier2.htm "	Dans le répertoire html du disque d
HREF= "/~brassart/fichier2.htm "	Dans le répertoire d'accueil de l'utilisateur brassart sur un système UNIX

Lien vers un autre site distant
Pour cela il suffit de remplacer dans l'exemple précédent ***document*** par une U.R.L.

Création d'un lien interne vers un endroit précis d'un document
Un lien interne pointe vers une ancre, c'est à dire un endroit à l'intérieur d'un document défini par un nom. Une ancre se défini dans le marqueur ***<A>*** en y ajoutant le paramètre ***NAME="nom"***.
Création d'un lien externe vers un endroit précis d'un document
Il est également possible d'utiliser les ancres dans les liens externes. Il faut alors spécifier l'ancre vers laquelle pointe le lien en ajoutant #nom à la fin de l'URL.

Listes

HTML définit 5 types de listes :

- ☞ Les listes numérotées ou ordonnées (ou Ordered (numbered) lists),
- ☞ Les listes non numérotées dites listes à puces dont les entées sont signalées par un signe typographique,
- ☞ Les listes de glossaire,
- ☞ Les listes de menu,
- ☞ Les listes de répertoire.

Listes numérotées ou Ordered (numbered) lists

Les options suivantes sont possibles :

- ☞ TYPE=1 : pour une liste numérotée 1,2,3...
- ☞ TYPE=A : pour un repérage type A,B,C...
- ☞ TYPE=a : pour un repérage type a,b,c...
- ☞ TYPE=I : pour une liste numérotée I,II,III,IV...
- ☞ TYPE=i : pour une liste numérotée i,ii,iii,iv...

START=n fait débuter le repérage (chiffres ou lettres) au rang numéro n.
Remarque : TYPE=1 est utilisé par défaut.

Listes descriptives ou Definition lists <DL> <DT> <DD>

L'environnement <DL> délimite une zone de liste de définition.
La commande <DT> introduit un nouveau terme de définition. C'est en général un élément court.
La commande "dd" introduit une description du terme de définition. Le résultat à l'écran est un décalage du texte vers la droite.

Style

L'utilisation de différents styles de polices de caractères permet de varier la présentation d'un texte. En HTML, on peut utiliser des styles logiques où le nom du style indique la nature du fragment de texte à écrire dans ce style ou des styles physiques où le nom du style indique explicitement le style de police que l'on souhaite voir utiliser.

Styles de caractères :

Styles logiques : à utiliser de préférence car ils sont interprétés par les différents logiciels clients WWW.

Les commandes de styles logiques sont les suivantes :

<*EM*> texte **</*EM*>**	met le texte généralement en italique.
<*STRONG*> texte **</*STRONG*>**	met le texte généralement en gras.
<*CODE*> texte **</*CODE*>**	pour l'utilisation d'une police à chasse fixe (encombrement des caractères constant).
<*SAMP*> caractères **</*SAMP*>**	séquence de caractères littéraux.
<*KBD*> saisie **</*KBD*>**	pour mettre en évidence une saisie d'utilisateur.
<*VAR*> variable **</*VAR*>**	pour indiquer le nom d'une variable.
<*DFN*> définition **</*DFN*>**	pour mettre en évidence la 1ère utilisation d'un terme.
<*CITE*> citation **</*CITE*>**	pour mettre en évidence une citation (généralement en italique).
<*ADDRESS*> adresse **</*ADDRESS*>**	cette commande n'est pas utilisée pour une adresse postale. Cette commande est généralement utilisée pour indiquer l'auteur d'un document ainsi que le moyen de le contacter ou bien elle donne l'adresse du document. Elle est souvent placée en fin de document.
<*BLOCKQUOTE*> texte **</*BLOCKQUOTE*>**	cette commande constitue avec le texte un texte avec retrait à gauche et à droite.
<*BLINK*> texte **</*BLINK*>**	pour faire clignoter le texte.
<*STRIKE*> texte barré **</*STRIKE*>**	cette commande permet de barrer du texte.
<*LISTING*> code informatique **</*LISTING*>**	cette commande permet d'afficher du code informatique
<*BIG*> texte **</*BIG*>**	cette commande permet d'écrire un texte en plus gros caractères que les caractères en cours.
<*SMALL*> texte **</*SMALL*>**	cette commande permet d'écrire un texte en plus petits caractères que les caractères en cours.
<*SUP*> texte **</*SUP*>**	cette commande permet d'écrire un texte en exposant.
<*SUB*> texte **</*SUB*>**	cette commande permet d'écrire un texte en indice.

CENTRAGE D'UN TEXTE <*CENTER*>…</*CENTER*>

La commande **<*CENTER*>** ... **</*CENTER*>** permet de centrer toutes les lignes d'un texte.

TEXTE PRÉ-FORMATÉ <*PRE*> ... </*PRE*>

Les espaces (plusieurs à la suite), tabulations, retour chariot n'ont pas de valeur en HTML.
La commande **<*PRE*> ... </*PRE*>** est utilisée pour inclure un texte pré-formaté dans un document HTML.
Les espacements, tabulations et retour chariot gardent alors leur sens initial.
Attention : les commandes HTML existant dans le texte pré-formaté seront interprétées.
L'option WIDTH=n peut-être utilisée pour limiter la longueur de la ligne.

Les couleurs utilisées dans BODY

Par défaut le fond du document est gris clair, les caractères sont noirs, les prises d'hypertextes sont bleues quand elles n'ont jamais été utilisées, violettes dans le cas contraire, rouges à l'instant de la sélection (dans le cas de NETSCAPE).

Il est possible de modifier ces couleurs en rajoutant à la commande les options suivantes :

☞ BGCOLOR=c	pour le fond du document
☞ TEXT=c	pour la couleur des caractères
☞ LINK=c	pour la couleur des prises d'hypertextes non utilisées
☞ VLINK=c	pour la couleur des prises d'hypertextes utilisées
☞ ALINK=c	pour la couleur des prises d'hypertextes à l'instant de la sélection

La valeur de c est composée de trois nombres hexadécimaux accolés (codés de 00 à FF) représentant le mélange des trois couleurs primaires RGB (Red, Green, Blue). Le nombre obtenu est précédé d'un dièse (#). Le blanc à pour valeur # FFFFFF ; le noir à pour valeur #000000. Toutes les autres couleurs sont obtenues par des dosages précis dans chacune des composantes RVB.

IMAGES DANS LE TEXTE

Des images peuvent être insérées dans le texte d'un document HTML, et elles sont au format **GIF** ou **JPG**.

Elles peuvent servir de prises d'hypertextes :

- ☞ Soit toute l'image réagit à un clic.
- ☞ Soit l'image réagit en fonction de la zone où le clic s'est produit.

Les commandes sont les suivantes :

☞ ***<IMG SRC="nom_image.gif">*** pour insérer une image locale, où nom_image.gif est le nom complet du fichier avec éventuellement son chemin relatif à partir du répertoire du document HTML.

☞ ***<IMG SRC="URL">*** pour insérer une image distante. URL étant l'adresse complète du fichier image.

☞ ***<A HREF="#etiquette"><IMG SRC="nom_image.gif"></A>*** pour effectuer le lien sur le point de branchement de nom : ***etiquette***, du document courant, en cliquant sur l'image nom_image.gif qui est la prise d'hypertexte. (nom_image.gif est le nom complet du fichier image).

☞ ***<IMG SRC="nom_image.gif" ISMAP>*** pour une image cliquable (ou réactive). Les images cliquables permettent des liens selon l'endroit du &quotclic" de la souris dans l'image. La requête est transmise à un programme exécutable (souvent IMAGEMAP.EXE) qui associe les coordonnées du &quotclic" à l'adresse correspondante.

Pour ces commandes les options suivantes sont possibles :

ALIGN=left/right/top/texttop/middle/absmiddle/baseline/bottom/absbottom

- ☞ left : positionne l'image à gauche de l'écran.
- ☞ right : positionne l'image à droite de l'écran.
- ☞ top : positionne le sommet de la ligne de texte au sommet de l'image.
- ☞ texttop : positionne l'axe horizontal de la ligne de texte au sommet de l'image.
- ☞ middle : positionne la base de la ligne de texte au milieu de l'image.
- ☞ absmiddle : positionne l'axe horizontal de la ligne de texte au milieu de l'image.
- ☞ baseline : positionne la base de la ligne de texte à la base de l'image.
- ☞ bottom : positionne la base de la ligne de texte à la base de l'image. ☞

absbottom : positionne la ligne de texte au bas de l'image.

WIDTH=n ou n% HEIGHT=m ou m% redimentionnent l'image : n et m valeurs de la largeur et de la hauteur de l'image en pixels ou n% et m% valeurs relatives par rapport à la taille normale de l'image.
VSPACE=n HSPACE=m éloignent le texte de l'image de n pixels au-dessus et au-dessous de l'image et de m pixels à gauche et à droite de l'image.

BORDER=n trace un cadre autour de l'image avec un trait de n pixels de large.

ALT="texte" le "texte" est une alternative à l'image lorsque le logiciel client WWW utilisé ne peut afficher l'image.

TABLEAUX *<TABLE> ... </TABLE>*

Les commandes sont les suivantes :

<TABLE> ... </TABLE> pour créer un tableau.

Pour cette commande les options suivantes sont possibles :

BORDER : trace un cadre en trait fin
BORDER=n : trace un cadre en trait de n pixels d'épaisseur
CELLSPACING=n : espacement de n pixels entre les cellules.
CELLPADDING=n : espacement autour de l'écriture dans les cellules. ***WIDTH=n ou n%*** : largeur en pixels ou largeur relative du tableau.

Remarque

cellspacing et cellpadding ont priorité sur width.

<CAPTION> nomtableau</CAPTION> pour indiquer le nom du tableau.

<TR> ... </TR> encadre une ligne du tableau.

Pour cette commande les options suivantes sont possibles :

ALIGN=left/right/center : position horizontale (gauche/droite/centre) du texte dans les cellules de la ligne.
VALIGN=top/middle/bottom/baseline : position verticale (haut/milieu/bas/bas) du texte dans les cellules de la ligne.

<TH> ... </TH> encadre une cellule d'en-tête du tableau. (cellule pouvant contenir un texte alphanumérique, une image, une liste, un lien, un autre tableau ou rien)(texte en gras).

Pour cette commande les options suivantes sont possibles :

ALIGN=left/right/center : position horizontale (gauche/droite/centre) du texte dans la cellule d'en- tête.
COLSPAN=n : fusionne n cellules horizontalement.
ROWSPAN=n : fusionne n cellules verticalement.
NOWRAP : supprime la césure éventuelle du texte de la cellule d'en-tête.

<TD> ... </TD> encadre une cellule du tableau. (cellule pouvant contenir un texte alphanumérique, une image, une liste, un lien, un autre tableau ou rien)

Pour cette commande les options suivantes sont possibles :

ALIGN=left/right/center : position horizontale (gauche/droite/centre) du texte dans la cellule.
COLSPAN=n : fusionne n cellules horizontalement.
ROWSPAN=n : fusionne n cellules verticalement.
NOWRAP : supprime la césure éventuelle du texte de la cellule.

Formulaires

La commande FORM

Des formulaires peuvent être préparés afin de saisir des données et les traiter au niveau du serveur.

Pour rédiger un questionnaire, il faut:

☞1.Établir une zone d'édition (appelée FORM) en utilisant les commandes <FORM></FORM>. ☞2.Définir la méthode à employer pour transmettre au serveur l'information recueillie dans les champs du formulaire.
☞3.Identifier l'emplacement et le nom du programme qui devra traiter l'information recueillie.
☞4.Fournir, s'il y a lieu, les arguments au programme de traitement des données.
Toute cette information se retrouve dans la commande suivante:

```
<FORM METHOD="POST" ACTION="/cgi-bin/questionnaire.cmd?xxx">
```

La méthode utilisée est ***POST***, le programme de traitement se nomme questionnaire.cmd et se retrouve dans le dossier cgi-bin, un seul argument est fourni au programme soit xxx.

Il est à noter que le programme de traitement des données (questionnaire.cmd en l'occurrence) doit être fourni par l'administrateur du serveur et créé en fonction de l'application à supporter.

Les commandes INPUT

Syntaxe : ***INPUT="TEXT"***

Parmi les choix disponibles en HTML, un des types d'entrée de données est le champ ***input type="text"***. Dans ce cas, il faut inscrire le type de champ, le nom du champ et ses dimensions à l'écran.

Exemple:

```
NOM: <input type="text" name="name" size=30>.
```

Le type ***text*** est un champ ou l'usager entre de l'information sur son clavier, dans une zone définie à l'écran par la commande ***size***.

Ainsi, une commande ***size=30*** est un champ d'une longueur de 30 espacements de largeur. Pour un champ plus long, entrez une valeur de 50 ou 70, selon l'espace requis pour couvrir toutes les possibilités.
La commande ***name="name"*** précise que l'on désire enregistrer le contenu du champ dans la rubrique ***"name"***.

La commande INPUT="RADIO

Un autre type de champ est le type ***"input type=radio"*** qui permet d'afficher une série de boutons radio comme choix de réponses.

Il suffit d'abord de poser la question puis de positionner la commande suivante:

```
<input type="radio" name="info" value="OUI">OUI
<input type="radio" name="info" value="NON">NON
```

La commande INPUT=CHECKBOX

La commande ***input Type=checkbox*** permet d'afficher une liste ou plusieurs choix sont possibles en même temps.
La commande s'écrit:

```
Texte <input name="nom_du_champ" TYPE=checkbox VALUE="texte"><BR>
ou
<input name="nom_du_champ" TYPE=checkbox VALUE="texte"> Texte <BR>
```

La commande SELECT NAME et OPTION

La commande ***select name*** et ***Option*** permet d'afficher une liste ou un seul choix est possible et qui s'affichent sous la forme d'un menu ***"pop-up"***. La commande s'écrit:

```
<select name><option selected>option1<option> option2<option>
option3</select>
```

La commande TEXTAREA.

La commande HTML utilisée est:
<***TEXTAREA name="nom_du_champ?" rows=n cols=m></TEXTAREA***>

Dans ce type, on spécifie d'abord le type soit textarea, ensuite le nom de la rubrique soit name=" *nom_du_champ* " puis les paramètres d'affichage de la zone de dialogue en rangées (n) et en colonnes (m).

```
<TEXTAREA name="zone_libre" rows=5 cols=40></TEXTAREA>
```

Les formulaires doivent être complétés avant fermeture par une commande permettant d'envoyer le contenu des champs remplis au serveur HTTP.

Cette commande s'écrit :

```
<input type="submit" value="Soumettre">
```

On ajoute également une deuxième commande qui permet à l'usager de reprendre le questionnaire s'il s'est trompé.
La commande en question s'écrit:

```
<input type="reset" value="effacer et recommencer">
```

Note Cours 2

Concepts de base du Java Script

Objectifs

- *Se familiariser avec l'environnement du développement web*
- *Connaître et comprendre les objets, les structures de données, les événements et les opérations du Java Script.*

Enoncé du cours :

Note 2

Concepts de base du Java Script

Bref préambule

A l'origine le Web n'était qu'une sorte de lieu de stockage. Il est aujourd'hui bien plus que cela: on s'y montre, s'y informe, s'y instruit et s'y amuse. Les outils du Web ont évolué en conséquence. Le HTML simple chaîne de balisage a été complété par différents langages de programmation.

Java Script est un langage de programmation très simple et constitue une excellente initiation à la programmation web. La création de programmes Java Script requiert très peu de connaissances.

Il est facile d'utiliser Java Script pour améliorer les pages Web crées en HTML. Les programmes en Java Script peuvent être composées d'une seule ligne. Cependant il est possible d'utiliser Java Script pour créer des applications complexes.

Partie I : Les bases de Java Script

I. Qu'est ce que Java Script.

Javascript est donc une extension du code Html des pages Web. Les scripts, qui s'ajoutent ici aux balises Html, peuvent en quelque sorte être comparés aux macros d'un traitement de texte.
Ces scripts vont être gérés et exécutés par le browser lui-même sans devoir faire appel aux ressources du serveur. Ces instructions seront donc traitées en direct et surtout sans retard par le navigateur.

Javascript a été initialement développé par Netscape et s'appelait alors LiveScript. Adopté à la fin de l'année 1995, par la firme Sun (qui a aussi développé Java), il prit alors son nom de Javascript.

II. Intégrer un script Java Script à une page Web

Le principe est simple. Il suffit de respecter les deux principes suivants :

- n'importe où.
- mais là où il le faut.

Le browser traite votre page Html de haut en bas (y compris vos ajoutes en Javascript). Par conséquent, toute instruction ne pourra être exécutée que si le browser possède à ce moment précis tous les éléments nécessaires à son exécution. Ceux-ci doivent donc être déclarés avant ou au plus tard lors de l'instruction.
Pour s'assurer que le programme script est chargé dans la page et prêt à fonctionner à toute intervention de votre visiteur (il y a des impatients) on prendra l'habitude de déclarer systématiquement (lorsque cela sera possible) un maximum d'éléments dans les balises d'en-tête soit entre <HEAD> et </HEAD> et avant la balise
<BODY>. Ce sera le cas par exemple pour les fonctions.
Rien n'interdit de mettre plusieurs scripts dans une même page Html.
Il faut noter que l'usage de la balise script n'est pas toujours obligatoire. Ce sera le cas des événements Javascript (par exemple onClick) où il faut simplement insérer le code à l'intérieur de la commande Html comme un attribut de celle-ci. L'événement fera appel à la fonction Javascript lorsque la commande Html sera activée. Javascript fonctionne alors en quelque sorte comme une extension du langage Html.

1. Une première instruction Javascript

Sans vraiment entrer dans les détails, voyons une première instruction Javascript (en fait une méthode de l'objet window) soit l'instruction alert().

```
alert("votre texte");
```

Cette instruction affiche un message (dans le cas présent votre texte entre les guillemets) dans une boite de dialogue pourvue d'un bouton OK. Pour continuer dans la page, le lecteur devra cliquer ce bouton.

Vous remarquerez des points-virgules à la fin de chaque instruction Javascript (ce qui n'est pas sans rappeler le C et le C++). Le Javascript, bon enfant, est moins strict que ces autres langages et ne signale généralement pas de message d'erreur s'ils venaient à manquer. On peut considérer que le point-virgule est optionnel et qu'il n'est obligatoire que lorsque vous écrivez plusieurs instructions sur une même ligne. On recommande quand même vivement dans la littérature d'en mettre de façon systématique.

Javascript est "bon enfant" car il n'est pas toujours trop strict sur la syntaxe et passe au-dessus de certaines libertés prises avec celle-ci. Très bien! Mais ce caractère "bon enfant" est à double tranchant car parfois, pour une raison indéterminée, il devient dans certaines situations plus rigoureux et alors bonne chance pour débugger votre script.

2. Remarques

Javascript est case sensitive. Ainsi il faudra écrire alert() et non Alert(). Pour l'écriture des instructions Javascript, on utilisera l'alphabet ASCII classique (à 128 caractères) comme en Html. Les caractères accentués comme é ou à ne peuvent être employés que dans les chaînes de caractères c.-à-d. dans votre texte de notre exemple.

Les guillemets " et l'apostrophe ' font partie intégrante du langage Javascript. On peut utiliser l'une ou l'autre forme à condition de ne pas les mélanger. Ainsi alert("...') donnera un message d'erreur. Si vous souhaiter utiliser des guillemets dans vos chaînes de caractères, tapez \" ou \' pour les différencier vis à vis du compilateur.

Partie II : Les opérateurs

Les variables, c'est bien mais encore faut-il pouvoir les manipuler ou les évaluer. Voyons (et ce n'est peut-être pas le chapitre le plus marrant de ce tutorial) les différents opérateurs mis à notre disposition par Javascript.

1. Les opérateurs de calcul

Signe	Nom
+	plus
-	Moins
*	multiplié par
/	Divisé par
%	Modulo
=	a la valeur

2. Les opérateurs de comparaison

Signe	Nom
==	Egal
<	Inférieur
<=	inférieur ou égal
=<	supérieur ou égal
!=	différent

Important : On confond souvent le = et le == (deux signes =). Le = est un opérateur d'attribution de valeur tandis que le == est un opérateur de comparaison. Cette confusion est une source classique d'erreur de programmation.

3. Les opérateurs logiques

Aussi appelés opérateurs booléens, ses opérateurs servent à vérifier deux ou plusieurs conditions.

Signe	Nom
&&	et
||	ou

Partie III : Les fonctions

1. Définition

Une fonction est un groupe de ligne(s) de code de programmation destiné à exécuter une tâche bien spécifique et que l'on pourra, si besoin est, utiliser à plusieurs reprises. De plus, l'usage des fonctions améliorera grandement la lisibilité de votre script.

En Javascript, il existe deux types de fonctions :

- les fonctions propres à Javascript. On les appelle des "méthodes". Elles sont associées à un objet bien particulier comme c'était le cas de la méthode Alert() avec l'objet window.
- les fonctions écrites par vous-même pour les besoins de votre script. C'est à celles-là que nous nous intéressons maintenant.

2. Déclaration des fonctions

Pour déclarer ou définir une fonction, on utilise le mot (réservé) function. La syntaxe d'une déclaration de fonction est la suivante :

function nom_de_la_fonction(arguments) {
... code des instructions ...}

Le nom de la fonction suit les mêmes règles que celles qui régissent le nom de variables (nombre de caractères indéfini, commencer par une lettre, peuvent inclure des chiffres...). Pour rappel, Javascript est sensible à la case. Ainsi fonction() ne sera pas égal à Fonction(). En outre, Tous les noms des fonctions dans un script doivent être uniques.

La mention des arguments est facultative mais dans ce cas les parenthèses doivent rester. C'est d'ailleurs grâce à ces parenthèses que l'interpréteur Javascript distingue les variables des fonctions. Nous reviendrons plus en détail sur les arguments et autres paramètres dans la partie Javascript avancé.

Lorsque une accolade est ouverte, elle doit impérativement, sous peine de message d'erreur, être refermée. Prenez la bonne habitude de fermer directement vos accolades et d'écrire votre code entre elles.

Le fait de définir une fonction n'entraîne pas l'exécution des commandes qui la composent. Ce n'est que lors de l'appel de la fonction que le code de programme est exécuté.

3. L'appel d'une fonction

L'appel d'une fonction se fait le plus simplement du monde par le nom de la fonction (avec les parenthèses).

Soit par exemple nom_de_la_fonction();

Il faudra veuiller en toute logique (car l'interpréteur lit votre script de haut vers le bas) que votre fonction soit bien définie avant d'être appelée.

4. Les fonctions dans <HEAD>...<HEAD>

Il est donc prudent ou judicieux de placer toutes les déclarations de fonction dans l'en-tête de votre page c.-à-d

.dans la balise <HEAD> ... <HEAD>. Vous serez ainsi assuré que vos fonctions seront déjà prises en compte par l'interpréteur avant qu'elles soient appelées dans le <BODY>.

5. Exemple

Dans cet exemple, on définit dans les balises HEAD, une fonction appelée message() qui affiche le texte
"Bienvenue à ma page". cette fonction sera appelée au chargement de la page voir onLoad=.... dans le tag
<BODY>.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="Javascript">
<--
function message() {
document.write("Bienvenue à ma page");
}
//-->
</SCRIPT>
</HEAD>
<BODY onLoad="message()">
</BODY>
</HTML>
```

6. Passer une valeur à une fonction

On peut passer des valeurs ou paramètres aux fonctions Javascript. La valeur ainsi passée sera utilisée par la fonction.
Pour passer un paramètre à une fonction, on fournit un nom d'une variable dans la déclaration de la fonction. Un exemple un peu simplet pour comprendre. J'écris une fonction qui affiche une boite d'alerte dont le texte peut changer.
Dans la déclaration de la fonction, on écrit :

```
function Exemple(Texte) {
alert(texte);}
```

Le nom de la variable est Texte et est définie comme un paramètre de la fonction. Dans l'appel de la fonction, on lui fournit le texte :

```
Exemple("Salut à tous");
```

7. Passer plusieurs valeurs à une fonction

On peut passer plusieurs paramètres à une fonction. Comme c'est souvent le cas en Javascript, on sépare les paramètres par des virgules.

```
function nom_de_la_fonction(arg1, arg2, arg3) {
... code des instructions ...
}
```

Notre premier exemple devient pour la déclaration de fonction :

```
function Exemplebis(Texte1, Texte2){...}
```

et pour l'appel de la fonction

```
Exemplebis("Salut à tous", "Signé Luc")
```

8. Retourner une valeur

Le principe est simple (la pratique parfois moins). Pour renvoyer un résultat, il suffit d'écrire le mot clé return suivi de l'expression à renvoyer. Notez qu'il ne faut pas entourer l'expression de parenthèses. Par exemple :

```
function cube(nombre) {
var cube = nombre*nombre*nombre return cube;
}
```

Précisons que l'instruction return est facultative et qu'on peut trouver plusieurs return dans une même fonction. Pour exploiter cette valeur de la variable retournée par la fonction, on utilise une formulation du type document.write(cube(5)).

Partie IV : Les conditions

1. l'expression if

if (condition vraie) {
instructions1;
}
else {
instructions2;
}

Pour être complet (et pour ceux qui aiment les écritures concises), il y a aussi :
(expression) ? instruction a : instruction b
Si l'expression entre parenthèse est vraie, l'instruction a est exécutée.Si l'expression entre parenthèses retourne faux, c'est l'instruction b qui est exécutée.

2. L'expression for

L'expression for permet d'exécuter un bloc d'instructions un certain nombre de fois en fonction de la réalisation d'un certain critère. Sa syntaxe est :

for (valeur initiale ; condition ; progression) {
instructions;
}

Prenons un exemple concret

```
for (i=0, i<10, i++) {
document.write(i + "<BR>")
}
```

3. While

L'instruction while permet d'exécuter une instruction (ou un groupe d'instructions) un certain nombre de fois.

while (condition vraie){
continuer à faire quelque chose
}

Aussi longtemps que la condition est vérifiée, Javascript continue à exécuter les instructions entre les accolades. Une fois que la condition n'est plus vérifiée, la boucle est interrompue et on continue le script.
Prenons un exemple.

```
compt=1;
while (compt<5) {
document.write ("ligne : " + compt + "<BR>");
compt++;
}
document.write("fin de la boucle");
```

4. Break

L'instruction break permet d'interrompre prématurément une boucle for ou while.
Pour illustrer ceci, reprenons notre exemple :

```
compt=1;
while (compt<5) {
if (compt == 4)
break;
document.write ("ligne : " + compt + "<BR>") ;
compt++;
}
document.write("fin de la boucle");
```

Le fonctionnement est semblable à l'exemple précédent sauf lorsque le compteur vaut 4. A ce moment, par le break, on sort de la boucle et "fin de boucle" est affiché.

Ce qui donnerait à l'écran :

ligne : 1
ligne : 2
ligne : 3
fin de la boucle

5. Continue

L'instruction continue permet de sauter une instruction dans une boucle for ou while et de continuer ensuite le bouclage (sans sortir de celui-ci comme le fait break).
Reprenons notre exemple ;

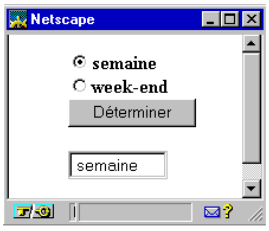
```
compt=1;
while (compt<5) { if (compt == 3){ compt++ continue;}
document.write ("ligne : " + compt + "<BR>");
compt++;
}
document.write("fin de la boucle");
```

Ici, la boucle démarre. Lorsque le compteur vaut 3, par l'instruction continue, on saute l'instruction document.write (ligne : 3 n'est pas affichée) et on continue la boucle. Notons qu'on a dû ajouter compt++ avant continue; pour éviter un bouclage infini et un plantage du navigateur (compt restant à 3).
Ce qui fait à l'écran : ligne : 1 ligne : 2 ligne : 4
fin de la boucle

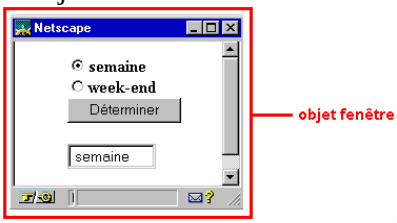
Partie V: Un peu de théorie objet

1. Les objets et leur hiérarchie

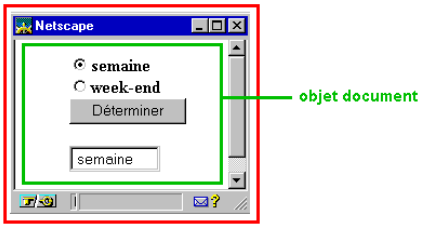
Javascript divise la page web en objets et surtout va nous permettre d'accéder à ces objets, d'en retirer des informations et de les manipuler.
Voyons d'abord une illustration des différents objets qu'une page peut contenir. On a chargé la page suivante :



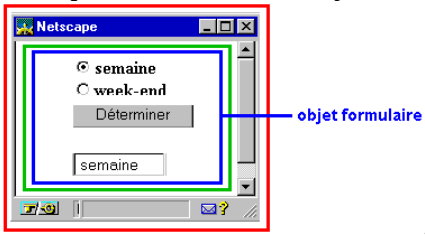
Cette page s'affiche dans une fenêtre. C'est l'objet fenêtre.



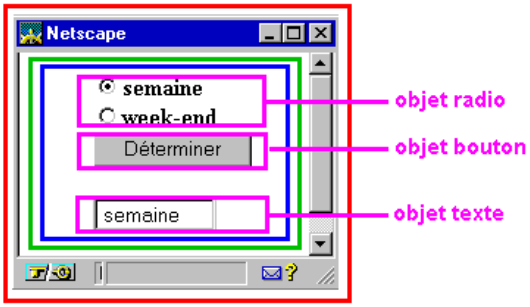
Dans cette fenêtre, il y a un document Html. C'est l'objet document. Autrement dit (et c'est là que l'on voit apparaître la notion de la hiérarchie des objets Javacript), l'objet fenêtre contient l'objet document.



Dans ce document, on trouve un formulaire au sens Html. C'est l'objet formulaire. Autrement dit, l'objet fenêtre contient un objet document qui lui contient un objet formulaire.



Dans ce document, on trouve trois objets. Des boutons radio, un bouton classique et une zone de texte. Ce sont respectivement l'objet radio, l'objet bouton, l'objet texte. Autrement dit l'objet fenêtre contient l'objet document qui contient l'objet formulaire qui contient à son tour l'objet radio, l'objet fenêtre contient l'objet document qui contient l'objet formulaire qui contient à son tour l'objet bouton et l'objet fenêtre contient l'objet document qui contient l'objet formulaire qui contient à son tour l'objet texte.



2. Les propriétés des objets

Une propriété est un attribut, une caractéristique, une description de l'objet. Par exemple, l'objet volant d'une voiture a comme propriétés qu'il peut être en bois ou en cuir. L'objet livre a comme propriétés son auteur, sa maison d'édition, son titre, son numéro ISBN, etc.
De même les objets Javascript ont des propriétés personnalisées. Dans le cas des boutons radio, une de ses propriétés est, par exemple, sa sélection ou sa non-sélection (checked en anglais).
En Javascript, pour accéder aux propriétés, on utilise la syntaxe :
nom_de_l'objet.nom_de_la_propriété
Dans le cas du bouton radio "semaine", pour tester la propriété de sélection, on écrira
document.form.radio[0].checked

Partie VI : Afficher du texte

1. La méthode write()

A la méthode document, Javascript a dédié la méthode "écrire dans le document", c'est la méthode write().
La syntaxe est assez simple soit write("votre texte");
On peut aussi écrire une variable, soit la variable resultat,
write(resultat);
Pour associer du texte (chaînes de caractères) et des variables, on utilise l'instruction write("Le résultat est " +
resultat);
On peut utiliser les balises Html pour agrémenter ce texte write("<B>Le résultat est</B>" + resultat); ou
write ("<B>" + "Le résultat est " + "</B>" + resultat)

2. L'instruction writeln()

La méthode writeln() est fort proche de write() à ceci près qu'elle ajoute un retour chariot à la fin des caractères affichés par l'instruction. Ce qui n'a aucun effet en Html. Pour faire fonctionner writeln() Il faut l'inclure dans des balises <PRE>.

```
<PRE>
<SCRIPT LANGUAGE="Javascript">
<--
document.writeln("Ligne 1");
document.writeln("Ligne 2");
//-->
</SCRIPT>
</PRE>
```

Autrement dit l'emploi de writeln() est anecdotique et on utilise simplement le tag
 avec la méthode write().

Partie VII : Les événements

1. Généralités

Avec les événements et surtout leur gestion, nous abordons le côté "magique" de Javascript.
En Html classique, il y a un événement que vous connaissez bien. C'est le clic de la souris sur un lien pour vous transporter sur une autre page Web. Hélas, c'est à peu près le seul. Heureusement, Javascript va en ajouter une bonne dizaine, pour votre plus grand plaisir.
Les événements Javascript, associés aux fonctions, aux méthodes et aux formulaires, ouvrent grand la porte pour une réelle interactivité de vos pages.

2. Les événements

Passons en revue différents événements implémentés en Javascript.

Description	Evénement	Objets
Lorsque l'utilisateur clique sur un bouton, un lien ou tout autre élément.	Clik	Lien hypertexte, Elément bouton, Case à cocher, Bouton Radio, Bouton Submit, Bouton Reset
Lorsque la page est chargée par le browser ou le navigateur.	Load	Fenêtre
Lorsque l'utilisateur quitte la page.	Unload	Fenêtre
Lorsque l'utilisateur place le pointeur de la souris sur un lien ou tout autre élément.	MouseOver	Lien hypertexte
Lorsque le pointeur de la souris quitte un lien ou tout autre élément.	MouseOut	Lien hypertexte
Lorsque un élément de formulaire a le focus c-à-d devient la zone d'entrée active.	Focus	Elément de texte , Liste de sélection
Lorsque un élément de formulaire perd le focus c-à-d que l'utilisateur clique hors du champs et que la zone d'entrée n'est plus active.	Blur	Elément de texte, Liste de sélection
Lorsque la valeur d'un champ de formulaire est modifiée.	Change	Elément de texte, Liste de sélection
Lorsque l'utilisateur sélectionne un champ dans un élément de formulaire.	Select	Elément de texte
Lorsque l'utilisateur clique sur le bouton Submit pour envoyer un formulaire.	Submit	formulaire

3. Les gestionnaires d'événements

Pour être efficace, il faut qu'à ces événements soient associées les actions prévues par vous. C'est le rôle des gestionnaires d'événements. La syntaxe est
onévénement="fonction()"
Par exemple, onClick="alert('Vous avez cliqué sur cet élément')".
De façon littéraire, au clic de l'utilisateur, ouvrir une boite d'alerte avec le message indiqué.

3.1 onclick

Evénement classique en informatique, le clic de la souris.
Le code de ceci est :

```
<FORM>
<INPUT TYPE="button" VALUE="Cliquez ici" onClick="alert('Vous avez bien cliqué ici')">
</FORM>
```

3.2 onLoad et onUnload

L'événement Load survient lorsque la page a fini de se charger. A l'inverse, Unload survient lorsque l'utilisateur quitte la page.

Les événements onLoad et onUnload sont utilisés sous forme d'attributs de la balise <BODY> ou

<FRAMESET>. On peut ainsi écrire un script pour souhaiter la bienvenue à l'ouverture d'une page et un petit mot d'au revoir au moment de quitter celle-ci.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE='Javascript'> function bienvenue() { alert("Bienvenue à cette page");
}
function au_revoir() {
alert("Au revoir");
}
</SCRIPT>
</HEAD>
<BODY onLoad='bienvenue()' onUnload='au_revoir()'> Html normal
</BODY>
</HTML>
```

3.3 onmouseOver et onmouseOut

L'événement onmouseOver se produit lorsque le pointeur de la souris passe au dessus (sans cliquer) d'un lien ou d'une image. Cet événement est fort pratique pour, par exemple, afficher des explications soit dans la barre de statut soit avec une petite fenêtre genre infobulle.

L'événement onmouseOut, généralement associé à un onmouseOver, se produit lorsque le pointeur quitte la zone sensible (lien ou image).

Notons qui si onmouseOver est du Javascript 1.0, onmouseOut est du Javascript 1.1. En clair, onmouseOut ne fonctionne pas avec Netscape 2.0 et Explorer 3.0.

3.4 onFocus

L'événement onFocus survient lorsqu'un champ de saisie a le focus c.-à-d. quand son emplacement est prêt à recevoir ce que l'utilisateur à l'intention de taper au clavier. C'est souvent la conséquence d'un clic de souris ou de l'usage de la touche "Tab".

3.5 onBlur

L'événement onBlur a lieu lorsqu'un champ de formulaire perd le focus. Cela se produit quand l'utilisateur ayant terminé la saisie qu'il effectuait dans une case, clique en dehors du champ ou utilise la touche "Tab" pour passer à un champ. Cet événement sera souvent utilisé pour vérifier la saisie d'un formulaire.

Le code est :

```
<FORM>
<INPUT TYPE=text onBlur="alert('Ceci est un Blur')">
</FORM>
```

3.6 onchange

Cet événement s'apparente à l'événement onBlur mais avec une petite différence. Non seulement la case du formulaire doit avoir perdu le focus mais aussi son contenu doit avoir été modifié par l'utilisateur.

3.7 onselect

Cet événement se produit lorsque l'utilisateur a sélectionné (mis en surbrillance ou en vidéo inverse) tout ou partie d'une zone de texte dans une zone de type text ou textarea

Partie VIII: Les formulaires

1. Généralités

Avec Javascript, les formulaires Html prennent une toute autre dimension. Mettons au point le vocabulaire que nous utiliserons. Un formulaire est l'élément Html déclaré par les balises <FORM></FORM>. Un formulaire contient un ou plusieurs éléments que nous appellerons des contrôles. Ces contrôles sont notés par exemple par la balise <INPUT TYPE= ...>.

2. Déclaration d'un formulaire

La déclaration d'un formulaire se fait par les balises <FORM> et </FORM>. Il faut noter qu'en Javascript, l'attribut NAME="nom_du_formulaire" a toute son importance pour désigner le chemin complet des éléments. En outre, les attributs ACTION et METHOD sont facultatifs pour autant que vous ne faites pas appel au serveur.
Une erreur classique en Javascript est, emporté par son élan, d'oublier de déclarer la fin du formulaire
</FORM> après avoir incorporé un contrôle.

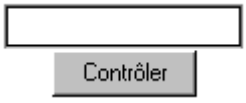
3. Le contrôle ligne de texte

La zone de texte est l'élément d'entrée/sortie par excellence de Javascript. La syntaxe Html est <INPUT TYPE="text" NAME="nom" SIZE=x MAXLENGTH=y> pour un champ de saisie d'une seule ligne, de longueur x et de longueur maximale de y.
L'objet text possède trois propriétés :

Propriété	Description
name	indique le nom du contrôle par lequel on pourra accéder.
Defaultvalue	indique la valeur par défaut qui sera affichée dans la zone de texte.
Value	indique la valeur en cours de la zone de texte. Soit celle tapée par l'utilisateur ou si celui-ci n'a rien tapé, la valeur par défaut.

Exemple « Lire une valeur dans une zone de texte »

Voici un exemple que nous détaillerons :
Voici une zone de texte. Entrez une valeur et appuyer sur le bouton pour contrôler celle-ci.



Le script complet est celui-ci :

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="javascript">
function controle(form1) {
var test = document.form1.input.value;
alert("Vous avez tapé : " + test);
}
</SCRIPT>
</HEAD>
<BODY>
<FORM NAME="form1">
<INPUT TYPE="text" NAME="input" VALUE=""><BR>
<INPUT TYPE="button" NAME="bouton" VALUE="Contrôler" onClick="controle(form1)">
</FORM>
</BODY>
</HTML>
```

4. Les boutons radio

Les boutons radio sont utilisés pour noter un choix, et seulement un seul, parmi un ensemble de propositions.

Propriété	Description
name	indique le nom du contrôle. Tous les boutons portent le même nom.
index	l'index ou le rang du bouton radio en commençant par 0.
checked	indique l'état en cours de l'élément radio
value	indique la valeur de l'élément radio.

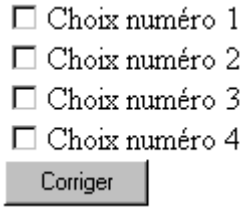
5. Les boutons case à cocher (checkbox)

Les boutons case à cocher sont utilisés pour noter un ou plusieurs choix (pour rappel avec les boutons radio un seul choix) parmi un ensemble de propositions. A part cela, sa syntaxe et son usage est tout à fait semblable aux boutons radio sauf en ce qui concerne l'attribut name.

Propriété	Description
name	indique le nom du contrôle. Toutes les cases à cocher portent un nom différent.
checked	indique l'état en cours de l'élément case à cocher.
value	indique la valeur de l'élément case à cocher.

Exemple :

Entrez votre choix :
Il faut sélectionner les numéros 1,2 et 4 pour avoir la bonne réponse.




```
<HTML>
<HEAD>
<script language="javascript">
function reponse(form4) {
if ( (form4.check1.checked) == true && (form4.check2.checked) == true && (form4.check3.checked) == false
&& (form4.check4.checked) == true)
{ alert("C'est la bonne réponse! ") }
else
{alert("Désolé, continuez à chercher.")}
}
</SCRIPT>
</HEAD>
<BODY>
Entrez votre choix :
<FORM NAME="form4">
<INPUT TYPE="CHECKBOX" NAME="check1" VALUE="1">Choix numéro 1<BR>
<INPUT TYPE="CHECKBOX" NAME="check2" VALUE="2">Choix numéro 2<BR>
<INPUT TYPE="CHECKBOX" NAME="check3" VALUE="3">Choix numéro 3<BR>
<INPUT TYPE="CHECKBOX" NAME="check4" VALUE="4">Choix numéro 4<BR>
<INPUT TYPE="button"NAME="but" VALUE="Corriger" onClick="reponse(form4)">
</FORM>
</BODY>
</HTML>
```

6. Liste de sélection

Le contrôle liste de sélection vous permet de proposer diverses options sous la forme d'une liste déroulante dans laquelle l'utilisateur peut cliquer pour faire son choix. Ce choix reste alors affiché.
La boite de la liste est crée par la balise <SELECT> et les éléments de la liste par un ou plusieurs tags
<OPTION>. La balise </SELECT> termine la liste.

Propriété	Description
name	indique le nom de la liste déroulante.
length	indique le nombre d'éléments de la liste. S'il est indiqué dans le tag <SELECT>, tous les éléments de la liste seront affichés. Si vous ne l'indiquez pas un seul apparaîtra dans la boite de la liste déroulante.
selectedIndex	indique le rang à partir de 0 de l'élément de la liste qui a été sélectionné par l'utilisateur

Un petit exemple comme d'habitude :
Entrez votre choix :


```
<HTML>
<HEAD>
<script language="javascript"> function liste(form5) {
alert("L\'élément " + (form5.list.selectedIndex + 1)); }
</SCRIPT>
</HEAD>
<BODY>
Entrez votre choix : <FORM NAME="form5">
<SELECT NAME="list">
<OPTION VALUE="2">Elément 2
<OPTION VALUE="3">Elément 3
</SELECT>
<INPUT TYPE="button"NAME="b" VALUE="Quel est l'élément retenu?" onClick="liste(form5)"> </FORM>
</BODY>
</HTML>
```

Partie IX : L'objet String

Revenons à l'objet String [Afficher du texte -- Avancé --] pour nous intéresser à la manipulation des caractères si utile pour l'aspect programmation de Javascript.

Instruction	Description
length	C'est un entier qui indique la longueur de la chaîne de caractères.
charAt()	Méthode qui permet d'accéder à un caractère isolé d'une chaîne.
indexOf()	Méthode qui renvoie la position d'une chaîne partielle à partir d'une position déterminée (en commençant au début de la chaîne pricipale soit en position 0).
LastIndexOf()	Méthode qui renvoie la position d'une chaîne partielle à partir d'une position déterminée (en commençant à la fin soit en position length moins 1).
substring(x,y)	Méthode qui renvoie un string partiel situé entre l position x et la position y-1.
toLowerCase()	Transforme toutes les lettres en minuscules
toUpperCase()	Transforme toutes les lettres en Majuscules.

PartieX : L'objet Math

Instruction	Description
abs()	x=Math.abs(y); La méthode abs() renvoie la valeur absolue (valeur positive) de y. Il supprime en quelque sorte le signe négatif d'un nombre.
max()	x=Math.max(y,z); La méthode max(y,z) renvoie le plus grand des 2 nombres y et z.
min()	x=Math.min(y,z); La méthode min(y,z) renvoie le plus petit des 2 nombres y et z.
pow()	x=Math.pow(y,z); La méthode pow() calcule la valeur d'un nombre y à la puissance z.
sqrt()	x=Math.sqrt(y); La méthode sqrt() renvoie la racine carrée de y.
parseFloat()	x=parseFloat("variable"); Cette fonction convertit une chaîne contenant un nombre en une valeur à virgule flottante.
parseInt()	x=parseInt(variable); Retourne la partie entière d'un nombre avec une virgule.

Partie XI : L'objet Date

Instruction	Description
new Date()	Cette méthode renvoie toutes les informations « date et heure » de l'ordinateur de l'utilisateur. Variable=new Date() ;
getYear()	variable_date=new Date(); an=variable_date.getYear() ; Retourne les deux derniers chiffres de l'année dans variable_date.
getMonth()	variable_date=new Date(); mois=variable_date.getMonth(); Retourne le mois dans variable_date sous forme d'un entier compris entre 0 et 11 (0 pour janvier, 1 pour février, 2 pour mars, etc.).
getHours()	variable_date=new Date(); hrs=variable_date.getHours(); Retourne l'heure dans variable_date sous forme d'un entier compris entre 0 et 23.
getMinutes()	variable_date=new Date(); min=variable_date.getMinutes(); Retourne les minutes dans variable_date sous forme d'un entier compris entre 0 et 59.

Partie XII : L'objet Array

1. Généralités

L'objet Array (ou tableaux) est une liste d'éléments indexés dans lesquels on pourra ranger (écrire) des données ou aller reprendre ces données (lire).

2. Tableau à une dimension

Pour faire un tableau, il faut procéder en deux étapes :

- d'abord construire la structure du tableau. A ce stade, les éléments du tableau sont vides.
- ensuite affecter des valeurs dans les cases ainsi définies.

On commence par définir le tableau :
nom_du_tableau = new Array (x);
où x est le nombre d'élément du tableau.
Ensuite, on va alimenter la structure ainsi définie :
nom_du_tableau[i] = "elément";
où i est un nombre compris entre 0 et x moins 1.
Exemple : un carnet d'adresse avec 3 personnes construction du tableau :
carnet = new Array(3); ajout des données : carnet[0]="Dupont"; carnet[1]="Médard";
carnet[2]="Van Lancker";
pour accéder un élément, on emploie :
document.write(carnet[2]);

Remarques :

- Quand on en aura l'occasion, il sera pratique de charger le tableau avec une boucle. Supposons que l'on ait à charger 50 images. Soit on le fait manuellement, il faut charger 0.gif, 1.gif, 2.gif... Soit on utilise une

boucle du style :

```
function gifs() {
gif = new Array(50);
for (var=i;i<50;i++)
{gif[i] =i+".gif";}}
```

- Javascript étant un langage peu typé, il n'est pas nécessaire de déclarer le nombre d'élément du tableau (soit x). Javascript prendra comme nombre d'éléments, le nombre i le plus élevé lors de "l'alimentation" de la structure (en fait i + 1).Ainsi la formulation suivante serait aussi correcte pour un tableau à 3 dimensions. carnet = new Array();

carnet[2]="Van Lancker";

3. Propriétés et méthodes

Elément	Description
length	Retourne le nombre d'éléments du tableau.
join()	Regroupe tous les éléments du tableau dans une seule chaîne. Les différents éléments sont séparés par un caractère séparateur spécifié en argument. Par défaut, ce séparateur est une virgule.
reverse()	Inverse l'ordre des éléments (ne les trie pas).
sort()	Retourne les éléments par ordre alphabétique (à condition qu'ils soient de même nature)

Partie XIII : Un peu de tout

1. Les boite de dialogue ou de message

Javascript met à votre disposition 3 boites de message :

- alert()
- prompt()
- confirm()

Ce sont toutes trois des méthodes de l'objet window.

2. La méthode alert()

La méthode alert() doit, à ce stade de votre étude, vous être familière car nous l'avons déjà souvent utilisée.
La méthode alert() affiche une boite de dialogue dans laquelle est reproduite la valeur (variable et/ou chaîne de caractères) de l'argument qui lui a été fourni. Cette boite bloque le programme en cours tant que l'utilisateur n'aura pas cliqué sur "OK".
Alert() sera aussi très utile pour vous aider à débbuger les scripts.
Sa syntaxe est :
alert(variable);
alert("chaîne de caractères");
alert(variable + "chaîne de caractères");
Si vous souhaitez écrire sur plusieurs lignes, il faudra utiliser le signe \n.

3. La méthode prompt()

Dans le même style que la méthode alert(), Javascript vous propose une autre boite de dialogue, dans le cas présent appelée boite d'invite, qui est composée d'un champ comportant une entrée à compléter par l'utilisateur. Cette entrée possède aussi une valeur par défaut.
La syntaxe est :
prompt("texte de la boite d'invite","valeur par défaut");

En cliquant sur OK, la méthode renvoie la valeur tapée par l'utilisateur ou la réponse proposée par défaut. Si l'utilisateur clique sur Annuler ou Cancel, la valeur null est alors renvoyée.
Prompt() est parfois utilisé pour saisir des données fournies par l'utilisateur. Selon certaines sources, le texte ne doit cependant pas dépasser 45 caractères sous Netscape et 38 sous Explorer 3.0.

4. La méthode confirm()

Cette méthode affiche 2 boutons "OK" et "Annuler". En cliquant sur OK, continue() renvoie la valeur true et bien entendu false si on a cliqué sur Annuler. Ce qui peut permettre, par exemple, de choisir une option dans un programme.
La syntaxe de l'exemple est :
confirm("Voulez-vous continuer ?")

TP 4

TP Java Script

Objectifs

- *Se familiariser avec l'environnement du développement web*
- *Connaître et comprendre les objets, les structures de données, les événements et les opérations du Java Script.*

Enoncé du TP :

TP 4

TP Java Script

Exercice 1 :

Reprendre l'exercice de calculatrice en ajoutant de la java script.

Exercice 2 :

Créer un document HTML qui déclare les cinq variables globales suivantes dans la section <script> contenue dans la section <head> du document : nom, age, moisNaissance, dateNaissance et anneeNaissance. Dans une autre section <script>, contenue dans la section <body> du document, déclarez une variable nommée infosNaissance. Combinez les cinq variables globales dans la variable infosNaissance, puis affichez la variable infosNaissance en utilisant la méthode document.write (). Enregistrez le fichier sous le nom Info.html. Ouvrez ensuite le document HTML pour voir à quoi il ressemble. Etes vous satisfait de l'affichage ? Comment pouvez vous l'améliorer.

Exercice 3 :

Créer une page Web avec une commande if demandant à l'utilisateur s'il souhaite un accueil personnalisé. S'il répond par l'affirmative, afficher une boite de dialogue qui lui demande son nom, puis affichez le nom dans une boite de dialogue alert. S'il répond par la négative, affichez une message d'accueil générique dans une boite de dialogue alert. Enregistrez le fichier sous le nom Accueil.html

Exercice 4 :

Question 1

Ecrire un convertisseur euros-francs, en utilisant une fonction javascript pour calculer la conversion. La conversion se fait lorsqu'on clique sur le bouton convertir.

Question 2

Proposer une autre solution, qui convertit, dès qu'on change la valeur dans Euros. Quelles en sont les limites ?

Exercice 5 :

Proposez un formulaire de saisie de nom, prénom et ville d'habitation. Le nom et la ville sont obligatoires et sont symbolisés par un « surlignement ».

Dès que les champs obligatoires sont remplis, le « surlignement » s'enlève.

Exercice 6 :

A chaque élément d'interface d'un formulaire HTML correspond un type d'objet JavaScript particulier. Celui-ci contient des propriétés qui permettent de connaître son contenu et gère différents évènements auxquels on peut associer des fonctions comme nous l'avons déjà fait pour l'évènement *onclick* des boutons.

L'objet représentant le formulaire a une propriété nommée *elements* qui est un tableau contenant tous les éléments d'interface du formulaire.

Les objets représentant des éléments d'interface ont tous les propriétés :

- form : qui représente le formulaire parent
- name : qui est le nom de l'élément
- type : qui est le type de l'élément
- value : qui est une valeur en général librement utilisable

➢ *Listes de choix*

Les listes de choix correspondent à la balise HTML *select*. L'objet correspondant en JavaScript est du type *Select*. Il possède les propriétés suivantes :

- *length : contient le nombre d'éléments de la liste*
- *selectedIndex : contient le numéro de l'option sélectionnée (la numérotation commence à 0)*
- *options : c'est un tableau contenant des objets de type Option qui ont une propriété text contenant le libellé de l'option et une propriété value.*

Les objets de type *Select* gèrent les évènements :

- *onblur*
- *onclick*
- *onfocus*
- *onchange*

➢ *Boutons radio*

Les boutons radio correspondent à la balise HTML *input type="radio"*. L'objet correspondant en JavaScript est du type *Radio*. Il possède :

- *une propriété checked qui est de type booléen et indique s'il a été coché*
- *une propriété value*

Pour que plusieurs boutons radio fonctionnent en groupe (l'activation de l'un désactive l'autre), il suffit de leur donner le même nom. Pour JavaScript ils formeront un tableau portant le même nom.

Les objets de type Radio gèrent les évènements :

- *onblur*
- *onclick*
- *onfocus*

➢ *Cases à cocher*

Les cases à cocher correspondent à la balise HTML *input type ="checkbox"*. L'objet correspondant en JavaScript est du type *Checkbox*. Il possède :

- *une propriété checked qui est de type booléen et indique s'il a été coché*
- *une propriété value*

Comme pour les boutons radio, les cases à cocher peuvent être regroupées dans un tableau en leur attribuant le même nom.

Les objets de type Radio gèrent les évènements :

- *onblur*
- *onclick*
- *onfocus*

➢ *Autres éléments d'interface*

- *Les zones de texte libre correspondent à la balise HTML textarea. En JavaScript, on utilise surtout leur propriété value qui représente le texte inscrit sous la forme d'une chaîne de caractères. On peut forcer des passages à la ligne en utilisant le caractère \n.*
- *Les éléments cachés correspondent à la balise HTML input type="hidden". En JavaScript on utilise surtout leur propriété value, cela permet d'envoyer des données sans le montrer à l'utilisateur.*
- *Les mots de passe correspondent à la balise HTML input type="password". En JavaScript on les utilise comme les lignes de saisie classiques.*

Question 1 :

Ecrire un texte à trous, où les réponses possibles sont données dans une liste de choix. Un bouton vérifier permet de vérifier la validité des réponses.

Question 2 :

Reprendre la question 1 mais en permettant à l'utilisateur de saisir au clavier dans une zone de texte, les mots manquants.

Exercice 7

Créer un formulaire de saisie d'un CV « light ». Nom, Prénom, date de naissance, adresse email, niveau d'études de l'année en cours.

- Pour vérifier une adresse mail vous pouvez chercher le '@' puis vérifier qu'il y a bien au moins quatre caractères après, suivis d'un « . » et de deux lettres au moins.

  On récupère la taille d'une chaîne avec

```
var machaine = "un exemple";
int longueur = machaine.length() ;
```

  On peut aussi accéder aux différents caractères par :

```
machaine.chatAt(4) pour avoir le 5ème.
```

  On peut également demander l'index d'un caractère donné accéder par :

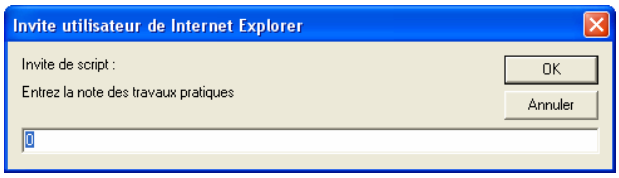
```
machaine.indexOf("x")renvoie -1 si pas trouvé, de 0 à n sinon
```

Exercice 8

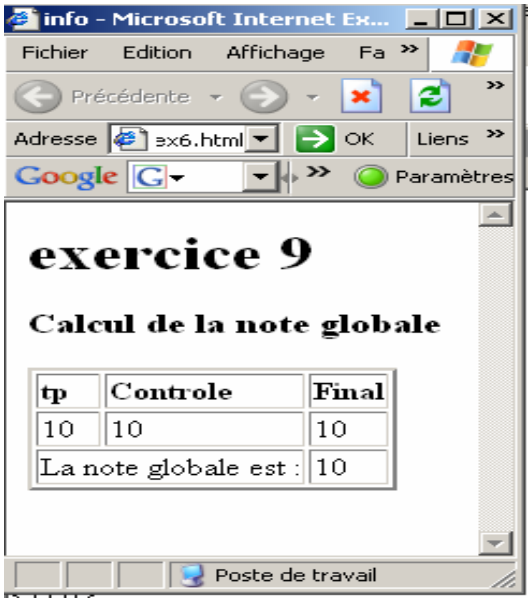
Écrivez le code **JavaScript** qui va simuler un tirage d'une loterie de 3 chiffres entre 0 et 9 avec répétition permise et affichera le résultat dans une page web, nommée **tp4Num8.html**.

Exercice 9

Vous allez produire une page web, nommée **tp4Num9.html**, utilisant du code **JavaScript**, pour saisir les notes d'un étudiant et calculer sa note globale.
Voici l'aspect visuel de votre application :



Le résultat de l'appel de prompt pour la note des TPs.



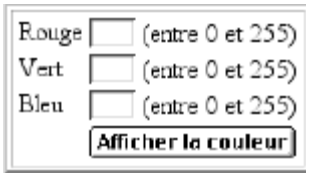
Le résultat après la saisie des valeurs 10, 10 et 10 comme notes de TP, Controle et Final.

Votre page web doit faire :

- Saisir les notes de tp, ainsi que des examens du contrôle et final en utilisant la fonction **prompt** et les conserver dans des variables.
- Calculer et afficher la note globale.
- La note du tp et d'examens sont sur 20. Vous n'êtes pas tenu de tester que les notes entrées sont entre 0 et 20.
- globale = tp * 30% + contrôle * 30% + final * 40%

Exercice 10

Dans l'extrait de page ci-dessous :



On voit un tableau à l'aide duquel l'utilisateur choisit la couleur de fond, en saisissant des valeurs entre 0 et 255 dans les champs correspondant aux trois couleurs basiques rouge, vert, bleu.

On vous fournit gracieusement le code de la fonction hexa() qui convertit un nombre décimal entre 0 et 255 en nombre hexadécimal à deux chiffres.

Note Cours 3

Cours PHP

Objectifs

- *Etudier le langage PHP.*
- *Connaître le principe de manipulation des bases de données par le php.*
- *Se familiariser avec l'environnement du développement web*

Enoncé du cours :

Note 3

Cours PHP

I- Introduction

1. Définition

PHP est un langage de script qui est principalement utilisé pour être exécuté par un serveur HTTP, mais il peut fonctionner comme n'importe quel langage interprété en utilisant les scripts et son interpréteur sur un ordinateur. On désigne parfois PHP comme une plate-forme plus qu'un simple langage.
Sa syntaxe et sa construction ressemblent à celles des langages C++ et Perl, à la différence que le
PHP peut être directement intégré dans du code HTML.

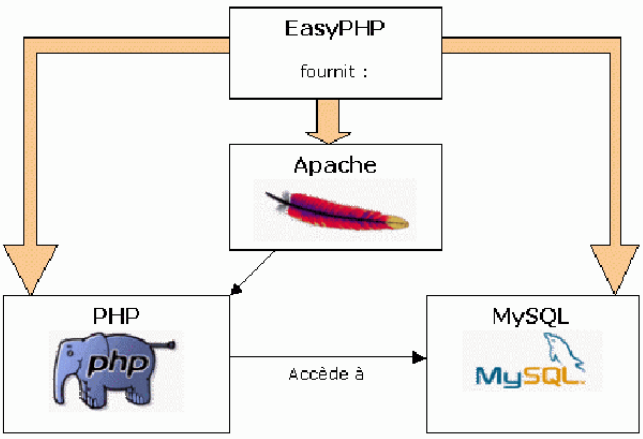
Exemple :

```
<html>
<head>
<title>Exemple</title>
</head>
<body>
<?
        echo "Bonjour, je suis un script PHP!";
?>
</body>
</html>
```

L'objet de ce langage est de permettre aux développeurs web d'écrire des pages dynamiques rapidement.

2. Mise en œuvre et déploiement (EasyPHP)

EasyPHP est un paquetage contenant à la fois Apache, PHP et MySQL
EasyPHP permet d'installer automatiquement et facilement une plateforme permettant l'exploitation d'un site web en PHP qui éventuellement aurait besoin d'un accès a une base de données. De la sorte on se libère des complications d'une installation manuelle de chacune de composantes séparément.



Procédure d'installation :

• Télécharger EasyPHP sur le site www.easyphp.org
• Double cliquer sur l'exécutable téléchargé
• Sélectionner le répertoire d'installation et suivre la procédure

Procédure de lancement :
Lorsqu'EasyPHP est lancé, les serveurs Apache et MySQL sont automatiquement lancés (il est même possible de le faire automatiquement au démarrage de Windows). Une petite icône s'installe dans la barre des tâches, à côté de l'horloge, permettant un accès rapide aux fonctions proposées par EasyPHP :
• Fichier Log : renvoie aux erreurs générées par Apache et MySQL
• Configuration : donne accès aux différentes configurations d'EasyPHP
• Administration : gestion des répertoires virtuels, des extensions, de PHPMyadmin
• Web local : ouvre la page http://localhost/ répertoire racine du site
• Démarrer/Arrêter : démarre/arrête Apache et MySQL
• Quitter : ferme EasyPHP
Pour que vos pages PHP soient interprétées, il est impératif de placer vos fichiers dans le répertoire www (sus le répertoire d'installation de easyphp). Ce répertoire est configuré par défaut comme répertoire racine de votre serveur web. Le serveur Apache est configuré pour ouvrir automatiquement un fichier index lorsque vous saisissez l'adresse 'http://localhost/' (à condition évidemment que le serveur Apache soit en route). Cette page sert de page d'accueil au web local et permet de vérifier le bon fonctionnement d'EasyPHP.
Il est conseillé de créer un répertoire par projet dans le répertoire www afin d'avoir une vision plus claire des développements.

II- Les fonctionnalités du langage

1. Premiers éléments du langage

Lorsque vous créez un code, il doit être placé entre balises php pour que celui- ci soit interprété, comme ceci:

```
<?
echo 'bonjour';
?>
ou encore
<?PHP
echo 'bonjour';
?>
Ce qui affichera à l'écran ' bonjour '.
```

Une syntaxe PHP se termine TOUJOURS par un point-virgule, si vous l'oubliez vous verrez apparaître une PARSE ERROR lors de l'exécution de votre fichier.

2. Intégration de PHP dans une page HTML

L'un des avantages du PHP est qu'il s'intègre facilement dans du code HTML classique. Chacun peut à sa guise inclure quelques parties en PHP dans des parties de code HTML.
Remarque importante :
Du moment que des parties de code PHP ont été intégrées dans une page HTML cette dernière doit impérativement être renommée en extension .php

Exemple

```
<html>
<body>

<font size="2" face="Arial">Le texte en HTML</font> <br>

<?
echo "<font size=\"2\" face=\"Arial\"> Le texte est en PHP.</font>";
?>

<br>
<font size="2" face="Arial"> < ? echo 'Encore du texte en PHP' ?></font>

</body>
</html>
```

Le résultat obtenu sera :
Le texte est en HTML
Le texte est en PHP
Encore du texte en PHP

3. Variables

Une variable est définie sous la forme $variable_nom.
L'affectation d'une variable se fait de la manière suivante : $variable_nom = variable_valeur.
L'appel de la valeur affectée à une variable se fait par son nom.
Syntaxe

```
<?
$variable1 = 'Bonjour 1';          //Affectation d'une chaîne avec quote simple

$variable2 = "Bonjour 2";          // Affectation d'une chaîne avec quote double

$variable3 = 5;                    // Affectation d'un entier

$variable4 = 2 + (3 * 5);          // Affectation d'un résultat d'opération

$variable5 = true;                 // Affectation Booléenne

?>
```

L'affichage des variables combinées à des chaînes de caractères peut se faire de plusieurs manières en utilisant les cotes simples (') ou les doubles cotes (").

```
<?
$nom = 'visiteur';
echo "bonjour $nom";
?>

Le résultat obtenu sera :
bonjour visiteur
```

```
<?
$nom = 'visiteur';
echo 'bonjour '.$nom;
?>

Le résultat obtenu sera :
bonjour visiteur
```

Remarque : Il y'a un . entre le ' et la variable $nom. Le point est un opérateur de concaténation pour les chaînes de caractères.

```
<?
$nom = 'visiteur';
echo 'bonjour $nom';   // La variable $nom ne sera pas interprétée
?>

Le résultat obtenu sera :
bonjour $nom
```

Remarques : Si vous utilisez les ' au lieu des " , la variable n'est pas interprétée comme une variable mais comme une chaîne de caractère.
Remarque sur l'usage des cotes simples ou doubles
Lors de l'usage des ' pour l'affichage d'une chaîne de caractère contenant des apostrophes, vous devez impérativement faire précéder ces apostrophes d'antislash. De la sorte l'apostrophe ne sera pas confondue avec le caractère de fin de la chaîne à afficher. Dans le cas contraire un message d'erreur sera affiché.

```
<?
echo 'vous n\'êtes pas inscrit';
?>

Le résultat obtenu sera :
Vous n'êtes pas inscrit
```

4. Conditions et boucles

➢ Conditions if

La syntaxe de base d'une instruction conditionnelle est la suivante :

```
<?
if($var == 'condition')
{
// 'condition vérifiée';
}
else
{
//'condition non vérifiée';
}
?>
```

Les opérateurs de contrôle sont les suivants :

==	strictement égale
!=	différent
>	plus grand que
<	inférieur à
>=	supérieur à
<=	inférieur à
&&	et
||	ou
AND	et
OR	ou
TRUE	1 ou oui
FALSE	0 ou non

Exemple

```
<?
$montant='100';
if ($montant <1000)
{
echo 'Montant inférieur à 1000';
}
else
{
echo 'Montant supérieur à 1000';
}
?>
```

Le résultat obtenu sera :
Montant inférieur à 1000

➢ Conditions elseif

```
<?
If ($var == 'condition1')
{
        // 'condition1 vérifiée';
}
elseif ($var == 'condition2')
{
        // 'condition2 vérifiée';
}
...
elseif ($var == 'conditionN)
{
        // 'conditionN vérifiée';
}
...
else
{
        echo 'Aucune condition n'est vérifiée';
}
?>
```

Exemple

```
<?
$montant='100' ;
if ($montant >=0 && $montant<1000)
{
        echo ' Votre montant '. $montant. ' est compris entre 0 et 1000';
}
elseif ($montant >=1000 && $montant<5000)
{
        echo ' Votre montant '. $montant. '  est compris entre 1000 et 5000';
}
else
{
        echo ' Votre montant '.$montant. ' est supérieur à 5000';
}
?>
```

Le résultat obtenu sera :
Votre montant 100 est compris entre 0 et 1000

- **Conditions SWITCH**

Syntaxe

```
<?
switch ($variable)
  {
  case condition1:
    //Traitement de la condition 1
    break;
  case condition2:
    //Traitement de la condition 2
    break;

 ....

  case conditionN:
    //Traitement de la condition N
    break;
  default:
    //Traitement par défaut
  }
?>
```

Exemple

```
<?
$variable = 3;

switch ($variable)
{
   case 1:
      echo 'La variable a pour valeur 1';
       break;
case 2:
      echo 'La variable a pour valeur 2';
       break;
case 3:
      echo 'La variable a pour valeur 3';
       break;
case 4:
      echo 'La variable a pour valeur 4';
       break;
case 5:
      echo 'La variable a pour valeur 5';
       break;
default:
        echo 'La variable n'appartient pas à l'intervalle [1,5]';
   }
?>
```

Le résultat obtenu sera :
La variable a pour valeur 3

- **Itération avec WHILE**

Syntaxe

```
<?
 While (condition)
   {
   //Traitements
   }
?>
```

Exemple

```
<?
$nbre_maximum = 6;
$i = 0;                 //initialisation de l'indice d'incrémentation
while ($i < $nbre_maximum)   //condition
    {
    echo  $i.' est inférieur à '. $nbre_maximum.'<br>';
    $i++;  // $i++ est équivalent à ($i+1)
    }
echo  $i.' est égal à '. $nbre_maximum;
?>
```

Le résultat obtenu sera :

0 est inférieur à 6
1 est inférieur à 6
2 est inférieur à 6
3 est inférieur à 6
4 est inférieur à 6
5 est inférieur à 6
6 est égal à 6

- **Itération avec FOR**

Syntaxe

```
<?
for($i=0; $i != condition ; $i++)
    {
    //Traitements réalisés
    }
?>
```

Exemple

```
<?
$nbre_maximum = 6;

//----------DEBUT BOUCLE----------
for($i=0; $i != $nbre_maximum ; $i++)
    {
    echo  $i.'est inférieur à '. $nbre_maximum.'<br>';
    }
//----------FIN BOUCLE----------

echo  $i.' est égal à '. $nbre_maximum;
?>
```

Le résultat obtenu sera :

0 est inférieur à 6
1 est inférieur à 6
2 est inférieur à 6
3 est inférieur à 6
4 est inférieur à 6
5 est inférieur à 6
6 est égal à 6

5. Tableaux

Il existe deux types de tableaux de variables sous PHP :

- ✓ Les tableaux à index numériques (tableaux numérotés) dans lesquels l'accès à la valeur de la variable passe par un index numérique

Ex : $tableau[0], $tableau[1], $tableau[2], …

- ✓ Les tableaux à index associatifs (ou tableaux associatifs) dans lesquels l'accès à la valeur de la variable passe par un index nominatif

Ex : $tableau[nom], $tableau[prénom], $tableau[adresse], …

Syntaxe

```
//Tableau à index numéroté
$tableau = array (valeur0,valeur1,valeur2, ...) ;

//Accès à chacune des valeurs
$tableau[0] donnera valeur0
$tableau[1] donnera valeur1
...

//Tableau à index associatif
$tableau = array (variable1 => valeur1, variable2 => valeur2, ...) ;

//Accès à chacune des valeurs
$tableau[variable1] donnera valeur1
$tableau[variable2] donnera valeur2
...
```

Exemple

```
<?
//Tableau à index numéroté
$tableau1 = array ('château','maison','bateau') ;
echo "Contenu du tableau 1 :<br>";
echo $tableau1[0]."<br>";
echo $tableau1[1]."<br>";
echo $tableau1[2]."<br>";

//Tableau à index associatif
$tableau2 = array ('prenom' =>'Tarak','nom' =>'Joulak','ville' =>'Tunis') ;
echo "Contenu du tableau 2 :<br>";
echo $tableau2['prenom']."<br>";
echo $tableau2['nom']."<br>";
echo $tableau2['ville']."<br>";
?>
```

Le résultat obtenu sera :

```
Contenu du tableau 1 :
château
maison
bateau
Contenu du tableau 2 :
Tarak
Joulak
Tunis
```

- **Quelques fonctions manipulant les tableaux**
  - ✓ PHP intègre une structure de langage qui permet de parcourir un à un les élément d'un tableau : **foreach().**

Syntaxe

```
$tableau = array (valeur0,valeur1,valeur2, …) ;
foreach ( $tableau as $valeur )
{
//Appeller ici la valeur courante par $valeur
…
```

Exemple

```
<?
$tableau1 = array ('chateau','maison','bateau') ;
foreach ( $tableau1 as $valeur )
{
echo $valeur."<br>";
}
?>
```

Le résultat obtenu sera :

château
maison
bateau

- ✓ **in_array()** permet de déterminer si une valeur existe dans le tableau. Elle retourne donc une valeur booléenne (True ou False).

Syntaxe

```
$resultat= in_array ( nom_de_la_valeur, tableau) ;
```

- ✓ **array_search** fonctionne comme in_array : il travaille sur les valeurs d'un tableau. Si il a trouvé la valeur, array_search renvoit la clé correspondante (c'est-à-dire le numéro si c'est un tableau numéroté, ou le nom de la variable si c'est un tableau associatif). Si il n'a pas trouvé la valeur, array_search renvoit false (comme in_array).

Syntaxe

```
$resultat= array_search ( nom_de_la_valeur, tableau) ;
```

6. Passage et transmission de variables

Passage et transmission de variables par formulaire

Quand dans un site web un formulaire est rempli et envoyé, le contenu des champs saisis est transféré à la page destination sous forme de variables. Ce passage de variables ou de paramètres peut se faire de deux manières : en GET ou en POST.

Syntaxe

```
<html>
<body>
        <!—Envoi d'un formulaire en POST  -->
        <form method="post" action="destination.php">
                <input type="text" name="nom" size="12"><br>
                <input type="submit" value="OK">
        </form>

        <!—Envoi d'un formulaire en GET  -->
        <form method="get" action=" destination.php">
                ...
        </form>
</body>
</html>
```

En GET les paramètres apparaissent associés à l'url sous formes de variables séparées par des &
(http://localhost/destination.php?nom=Tarak&prenom=Joulak).
En POST le passage de paramètre se fait de manière invisible.
Selon que la méthode d'envoi a été du GET ou du POST la récupération du contenu des variables est faite selon une syntaxe différente :

Syntaxe

```
<?
//Dans le cas d'un envoi des paramètres en POST
$variable1=$_POST['nom_du_champ'] ;

//Dans le cas d'un envoi des paramètres en GET
$variable1=$_GET['nom_du_champ'] ;
?>
```

Exemple

Dans l'exemple suivant le fichier formulaire.html contient le script html permettant d'afficher un formulaire et d'envoyer les résultats de la saisie à la page resultat.php qui elle les affichera.

Fichier formulaire.html

```
<html>
<body>
        <form method="post" action="resultat.php">
        Nom : <input type="text" name="nom" size="12"><br>
        Prénom : <input type="text" name="prenom" size="12">
        <input type="submit" value="OK">
        </form>
</body>
</html>
```

Fichier resultat.php

```
<?
//Récupération des paramètres passés
$prenom = $_POST['prenom'];
$nom = $_POST['nom'];

//affichage des paramètres
echo "<center>Bonjour $prenom $nom</center>";
?>
```

En exécutant à travers le serveur web le fichier formulaire.html , en remplissant le formulaire et en cliquant sur OK, nous sommes emmenés vers la page resultat.php qui nous affiche une phrase composée des champs saisis dans le formulaire. Les champs saisis sont donc passés de formulaire.html vers resultat.php.

- **Passage et transmission de variables par hyperlien**

Des paramètres ou variables peuvent passer d'une page source vers une page destination sans transiter par un formulaire pour leur envoi. Les hyperliens peuvent être des vecteurs de passage de paramètre.
Syntaxe

```
<!—Syntaxe d'envoi -->
<a href=destination.php ?variable1=contenu1&variable2=contenu2&...> Lien </a>
```

La récupération des paramètres dans la page destination se fait par le tableau $_GET

```
<?
$variable1=$_GET['variable1'] ;
$variable2=$_GET['variable2'] ;
...
?>
```

7. Fonctions Prédéfinies

- Quelques fonctions nous permettent de vérifier qu'une variable possède une valeur ou au contraire, n'en a pas.

Fonction	Signification	Exemple
isset()	détermine si une variable possède une valeur	`isset ($var)`
unset()	détruit une variable, libérant la mémoire qu'elle occupe	`unset ($var)`
empty()	vérifie si une variable n'est pas définie, vide ("") ou égale à 0	`empty ($var)`

- Il existe également de nombreuses fonctions qui permettent d'exploiter les chaînes de caractères, voici les principales :

Fonction	Signification	Exemple
strlen()	renvoie le nombre de caractères contenus dans la chaîne	`$nbcar = strlen($var);`
substr()	extrait un nombre de caractères d'une chaîne à partir d'une position	`$extrait = substr($var,$pos,$nb);`
trim()	supprime les espaces avant et après la chaîne	`$var = trim($var);`

- Manipulation des dates « Code à utiliser avec la fonction date () » :

format	**description**	**Exemple**
a	"am" ou "pm" minuscules	pm
d	jour du mois	01 /20
D	jour de la semaine en 3 lettres	mon
F	nom du mois	Janvier
h	heure (format 12 heures avec 0 en entête)	12
H	heure (format 24 heures avec 0 en entête)	08
g	heure (format 12 heures sans 0 en entête)	4
G	heure (format 24 heures sans 0 en entête)	10
i	minutes	44
j	jours du mois (pas de 0 en entête)	3
m	mois de l'année (0 en entête)	04
M	mois de l'année en 3 lettres	jui
n	mois de l'année; pas de 0 en entête	4
s	secondes	30
y	année à 2 chiffres	02
Y	année en 4 chiffres	2002
d-m-Y	Date du jour de la forme dd-mm-aaaa	02-04-2002

III- Utilisation d'une base de données MySql

PHP fonctionne nativement avec une base de données MYSQL. MYSQL est un système de gestion de base de données (SGBD) qui permet d'entreposer des données de manière structurée (Base, Tables, Champs, Enregistrements). Le noyau de ce système permet d'accéder à l'information entreposée via un langage spécifique le SQL. Ainsi, dans les chapitres précédents nous avions vu que l'information au mieux pouvait être stockée dans des fichiers accessibles en lecture et écriture par des scripts PHP. MYSQL vient ajouter une couche supplémentaire de stockage des données qui est plus commode, rapide et puissante d'utilisation.

Voici ce qu'il peut se passer lorsque le serveur reçoit une demande d'un client de consultation d'une page en PHP qui fait appel à des données stockées sous MYSQL:

1. Le serveur WEB envoie le nom de la page PHP demandée à l'interpréteur PHP.
2. PHP exécute le script existant dans la page. Sitôt que des instructions relatives à la connexion à une base de données trouvées, PHP se charge d'envoyer les requêtes d'exécution à MYSQL.
3. MySQL exécute la requête et renvoie à PHP le jeu de données résultat.
4. PHP termine son traitement et renvoie la page HTML générée au serveur web qui la transmet à l'internaute.

1. SQL : petit récapitulatif du langage

Structured Query Langage est langage standard pour communiquer avec une base de données. Il permet la création de bases, la définition de tables, la consultation des enregistrements ainsi que leur mise à jour.

➢ Ajout d'un enregistrement

INSERT INTO t_personne (id, Nom, Prenom, Age) VALUES (NULL, 'Joulak', 'Tarak', 32)
Cette requête SQL va permettre d'ajouter une ligne d'enregistrement à la table t_personne encore vide.
Le champ Nom prendra pour valeur Joulak
Le champ Prenom prendra pour valeur Tarak
Le champ Age prendra pour valeur 32
Il est à noter que le champ id étant en mode auto incrémental il ne faut pas lui assigner de valeur.
Le compteur s'incrémentant tout seul à chaque nouvel enregistrement.

➢ Consultation d'un enregistrement

SELECT * FROM t_personne WHERE id=1
Cette requête va extraire de la base de données la ligne d'enregistrement ayant pour identifiant 1
(id=1)

➢ Mise à jour d'un enregistrement

UPDATE t_personne SET Age= 35 WHERE id=1
Cette requête va mettre à jour dans la table t_personne l'enregistrement dont le champs id est égal à
1 (id=1) en modifiant le champs Age à 35.

- **Suppression d'un enregistrement**

DELETE FROM t_personne WHERE id=1
Cette requête SQL va supprimer de la table t_personne l'enregistrement ayant pour identifiant 1
(id=1).

2. Accéder à MYSQL via PHP

Dans ce paragraphe nous allons voir comment combiner le langage SQL aux fonctions spécifiques de PHP pour exploiter le contenu d'une base de données MYSQL.

- **Connection à un serveur MYSQL**

La première opération a réaliser pour accéder à MYSQL via un script PHP correspond à la connection à un serveur de base de données MYSQL.

Syntaxe

```
<?
mysql_connect("$nom_serveur_MYSQL", "$utilisateur", "$mot_de_passe");
?>
```

Exemple

```
<?
$nom_serveur_MYSQL="localhost" ;
$utilisateur="root" ;
$mot_de_passe="" ;
mysql_connect("$nom_serveur_MYSQL", "$utilisateur", "$mot_de_passe");
?>
```

La fonction mysql_connect() retourne un booléen ; True si la connection est possible False si elle ne l'est pas.

- **Connection à une base de données MYSQL**

Suite à la Connection au serveur MYSQL, il faut choisir quelle est la base du serveur MYSQL sur laquelle nous désirons nous connecter. Un serveur MYSQL pouvant contenir plusieurs bases de données.

Syntaxe

```
<?
mysql_select_db("$nom_de_la_base");
?>
```

Exemple

```
<?
$adresse_serveur_MYSQL="localhost" ;
$utilisateur="root" ;
$mot_de_passe="" ;
$nom_de_la_base="exercice" ;
mysql_connect("$adresse_serveur_MYSQL", "$utilisateur", "$mot_de_passe");
mysql_select_db("$nom_de_la_base");
?>
```

De même la fonction mysql_select_db() retourne un booléen ; True si la connection est réussie, False si elle a échoué.

➢ Exécution d'une requête SQL via PHP

Le lancement d'une requête SQl au travers d'un script PHP se fait par le biais de la fonction mysql_query() qui prend en paramètre la requête SQL et retourne true ou false selon que la requête ait réussi ou échoué. Pour le cas particulier d'une requête avec SELECT et si la requête a réussi alors un identifiant est retourné afin d'être exploité par d'autres fonctions.

Syntaxe

```
<?
...
$res = mysql_query("$requete_sql");
?>
```

Exemple

```
<?
$adresse_serveur_MYSQL="localhost" ;
$utilisateur="root" ;
$mot_de_passe="" ;
$nom_de_la_base="exercice" ;
$requete_sql="INSERT INTO t_personne (id, Nom, Prenom, Age) VALUES (NULL,
'Joulak', 'Tarak', 32)" ;
mysql_connect("$adresse_serveur_MYSQL", "$utilisateur", "$mot_de_passe");
mysql_select_db("$nom_de_la_base");
$res = mysql_query("$requete_sql");
?>
```

Dans l'exemple précédent une ligne d'enregistrement a été ajoutée à la table t_personne contenant Tarak Joulak 32.

➢ Parcourir le résultat d'un SELECT

Une requête SQL de consultation peut retourner plusieurs enregistrements. De ce fait il y a besoin de pouvoir les consulter enregistrement par enregistrement et champs par champs.

mysql_fetch_array() est la fonction PHP destinée pour ce type de traitements. Cette fonction prend en entrée l'identifiant retourné par mysql_query(). Elle retourne un tableau contenant la ligne demandée ou false s'il n'y a plus d'enregistrement. Elle est généralement exploitée dans une boucle WHILE.

Syntaxe

```
<?
...
$data = mysql_fetch_array($resultat_de_mysql_query) ;
?>
```

Exemple

```
<?
mysql_connect("localhost" , "root" , "");
mysql_select_db("exercice");
$requete_sql="SELECT * FROM  t_personne" ;
$res = mysql_query("$requete_sql");
while ($data= mysql_fetch_array($res))
{
echo $data['Nom'].' '.$data['Prenom']. '  ' .$data['Age']. '<br>';
}
?>
```

L'exécution de ce script affichera l'ensemble du contenu de la table t_personne avec un enregistrement par ligne.

➢ Les principales fonctions MySQL en PHP

Fonction	Description
mysql_close()	ferme la connexion MySQL
mysql_connect()	ouvre une connexion à un serveur MySQL
mysql_error()	renvoie l'erreur MySQL rencontrée
mysql_fetch_array()	retourne une ligne de résultat sous la forme d'un tableau associatif
mysql_fetch_assoc()	lit une ligne de résultat dans un tableau associatif
mysql_fetch_row()	découpe une ligne de résultat en un tableau
mysql_free_result()	libère la mémoire du résultat de la dernière requête
mysql_num_rows()	renvoie le nombre d'enregistrements résultants d'une requête de type SELECT
mysql_query()	envoie une requête SQL
mysql_select_db()	sélectionne une base de données MySQL

TP 5

Introduction au PHP

Objectifs

- *Etudier le langage PHP.*
- *Se familiariser avec l'environnement du développement web*

Enoncé du TP :

TP 5

Introduction au PHP

EXERCICE 1 :

Objectifs : inclure des balises php dans une page HTML. Mixer le code php et HTML. Utilisation de balises.

- Afficher dans une page la phrase « Ceci est une ligne créée uniquement en PHP ».
- Afficher à la ligne suivante : « Ceci est la 2ème phrase créée avec PHP».
- Créer un lien sur le site de l'ISET (www.iset.org).

EXERCICE 2 :

Objectifs : Déclaration et initialisation de variables. Utilisation du point de concaténation.

Déclarer 2 variables : nom et prénom. Les initialiser avec les valeurs « Zitouni » et « Ali» et les afficher sur la page en utilisant 3 modes syntaxiques différents :

- 2 commandes echo
- 1 commande echo avec 1 seule chaîne de caractère
- 1 commande echo avec le point de concaténation

EXERCICE 3 :

Objectifs : Utilisation de l'instruction IF.

- Affecter respectivement les valeurs 150, 50 et 10 aux variables prix_table, prix_armoire et Nombre.
- Calculer le prix HT total pour les 10 armoires.
- Comparer le prix de l'armoire et de la table et afficher quel est le prix le plus élevé.

EXERCICE 4 :

Objectifs : Utilisation des instructions WHILE et FOR.

Affecter une valeur à la variable nbre et afficher la somme des entiers de 1 à nbre.

NB : on réalisera cet exercice avec l'instruction FOR puis avec l'instruction WHILE.

EXERCICE 5:

Objectifs : Utilisation des instructions WHILE et FOR.

Concevez un programme qui écrit 500 fois «Je dois faire des sauvegardes régulières de mes fichiers.».

EXERCICE 6 :

Objectifs : Utilisation des instructions WHILE et FOR.

Écrire un programme PHP qui affiche tous les nombres impairs entre 0 et 15000, par ordre croissant : «1 3 5 7 ... 14995 14997 14999».

EXERCICE 7 :

Objectifs : Utilisation des tableaux, utilisation de fonctions et utilisation de fichiers : require()

Initialiser un tableau de 4 cases (contenant des nombres) et en faire la somme.

a) sans faire de fonction

b) en créant une fonction somme

c) en créant un fichier spécifique qui contient la fonction somme.

EXERCICE 8 :

Objectifs : Utilisation de formulaires et de contrôles. Bouton submit. Retour page précédente.

- Construire une page qui permette de saisir un nom et un mot de passe.
- Renvoyer l'utilisateur sur une autre page et lui afficher si son mot de passe est correct ou non (NB : le mot de passe valide sera « mot »).
- Sur cette 2ème page : prévoir un bouton retour.

EXERCICE 9 :

Objectifs : Utilisation de formulaires et de contrôles. Liste de choix et liste de valeurs.

Construire une page qui permette d'afficher :

- une liste avec les noms des vendeurs suivants : M. Ali, M. Zied, M. Maher et M. Mahdi (on utilisera une liste non modifiable).
- une liste qui affiche la liste des produits disponibles (la liste des produits est paramétrée dans le fichier produits.php)

Ajouter une zone de texte pour saisir le nombre de produits à commander et renvoyer sur une autre page le récapitulatif de la demande (ex : vous avez commandé 10 armoires auprès de M. Ali).

TP 6

PHP & BD

Objectifs

- *Connaître le principe de manipulation des bases de données par le php.*
- *Se familiariser avec l'environnement du développement web*

Enoncé du TP :

TP 6

PHP & BD

Syntaxe php	
Etablir une connexion à une BDD	$nom_connexion = mysql_connect ("nom_lien_odbc", nom_login", "mot_de_passe");
Exécuter une requête et renvoyer le résultat dans un recordset	$nom_recordset = mysql_query (« texte SQL à exécuter / nom_variable_contenant_texte_SQL »);
Balayer toutes les lignes d'un recordset	while ($nom_variable = mysql_fetch_row($nom_recordset)) {}
Afficher la valeur d'un champ d'un recordset pour la ligne en cours	$nom_variable = mysql_fetch_row($nom_recordset); echo $nom_variable[« num_du_champs »] ;
Fermeture de la connexion	mysql_close($maConnexion);
Ouvrir une session ou accéder à la session	Session_start() ;
Créer une variable session	Session_register("nom_variable") ; NB : le nom de la variable n'est pas précédé du $
Fermer une session	Session_destroy() ;
Pour utiliser une variable session	$_SESSION["nom_variable_session"] ;
Pour détruire une variable session	unset ($_SESSION["nom_variable_session"]) ;
Syntaxe html	
Zone de texte	<INPUT id=id_zone_de_texte name=nom_zone_de_texte>
Bouton submit	<INPUT id=id_bouton type=submit value= « texte à afficher sur le bouton » name=nom_bouton >
Formulaire	<FORM action = "nom_page_à_afficher_sur_clic_de_submit" METHOD = « get/post »> … </FORM>
Récupération de valeurs d'une page à une autre	$nom_variable = $_GET["nom_contrôle_page_précédente"] ou : $nom_variable = $_POST["nom_contrôle_page_précédente"]
Liste de valeurs	<SELECT NAME ="nom_liste"> <OPTION VALUE ="valeur_à_stocker" > Valeur_à_afficher </OPTION> <OPTION VALUE ="valeur_à_stocker" > Valeur_à_afficher </OPTION> … </SELECT>
Balise d'ancre Pour récupérer la variable transmise	<A href="nom_du_fichier?nom_variable = valeur à transmettre"> texte à afficher </A> $nom_nouvelle_variable = $_GET(["nom_variable"]

Formulaires avec PHP

Énoncés :

1. Écrire un **formulaire** qui demande le nom et l'âge de l'utilisateur. Le bouton *submit* de ce formulaire provoquera l'affichage d'une page qui saluera l'utilisateur avec cette phrase : «Bonjour *machin*, vous avez *xx* ans...» (avec les bonnes valeurs, bien entendu).
2. Deux vacanciers ont laissé à Tunis leur bébé de 9 mois, qui n'avait pas été sage. Quelle ne fut pas leur surprise quand 6 mois plus tard, rentrés chez eux à Sousse, ils ont vu arriver leur enfant qui avait fait à quatre pattes le trajet Tunis-Sousse par l'autoroute.

   Écrire un **formulaire** qui permet de saisir la distance parcourue par le bébé, le nombre d'heures où il marchait par jour, et le nombre de jours qu'il a passés sur la route. Le formulaire affichera alors la vitesse du bébé.
3. Écrire un **formulaire** qui demande le nom et le sexe de l'utilisateur (M ou Mme). Ce formulaire appelle une page qui affichera «Bonjour monsieur Truc» ou «Bonjour madame Bidule» suivant le cas (avec le vrai nom de la personne, bien entendu!).
4. Écrire un **formulaire** «calculatrice» : 2 cases pour la saisie des opérandes, un groupe de 4 cases à cocher (ou une liste déroulante) pour le choix de l'opération, et affichage du résultat de l'opération:
5. Un permis de chasse à points remplace désormais le permis de chasse traditionnel. Chaque chasseur possède au départ un capital de 100 points. S'il tue une poule il perd 1 point, 3 points pour un chien, 5 points pour une vache et 10 points s'il tue son meilleur ami. Le permis coûte 200 euros. Écrire un **formulaire** qui permet de saisir la liste des victimes du chasseur et calcule le prix à payer pour les permis en surplus.

BD avec PHP

Préambule :

Citoyen (NCIN, Nom, Prenom)
Domicile (Numdomicile, Adresse, Ville, Description, NCIN)
Maladie (Nummaladie, Nom, Description, Date_ Contamination, Date_ Guérison, Degré_ gravité, NCIN)

Énoncés :

6. Afficher le contenu des tables citoyen, domicile, maladie.
7. Sélectionner un citoyen dans une liste pour afficher ses différents domiciles, les plus récents en premier.
8. Choisir une ville, puis un citoyen de cette ville, puis sélectionner une de ses maladies pour afficher les dates de contamination et de guérison ainsi que le degré de gravité de la maladie.
9. On souhaite gérer une société qui vend des produits de bureautique. La société dispose d'une base de données **Commandes** dont voici le descriptif succinct des six tables qui la composent :

- une table ***Articles*** qui recense les références des différents produits que vend la société. Les attributs sont : **`Reference`** (clé), **`Categorie`**, **`Nom`**, **`Fournisseur`**, **`PrixHT`**, **`TauxTVA`**, **`QteStock`**, **`SeuilCritique`** (du stock).
- une table ***Clients*** recense les clients avec les attributs : **`NimClient`** (clé), **`Societe`**, **`Contact`**, **`Adresse`**, **`CodePostal`**, **`Ville`**, **`Tel`**, **`Fax`**, **`Confirmation`**, **`Paiement`**, **`Informations`**, **`EncoursMax`** (encours maximum toléré chez le client).
- une table ***Encours*** qui recense les encours des clients (c'est-à-dire le total des montants non encore réglé par chaque client). Elle contient les attributs : **`NumClient`** et **`Encours`** (nombre réel), **`DateCmd`** (date de la dernière commande).
- une table ***Fournisseurs*** recense les fournisseurs. Un fournisseur est reconnu par son : **`Numero`**, **`Societe`**, **`Contact`**, **`Adresse`**, **`CodePostal`**, **`Ville`**, **`Tel`**, **`Fax`**.
- une table ***Factures*** contenant des informations générales sur les factures émises (**`NumFacture`**, **`NumClient`**, **`DateFac`** (date de la facture), **`MontantHT`**, **`MontantTVA`**, **`MontantTTC`**, **`DateReg`** (date de règlement de la facture), **`MontantReg`** (montant réglé)).
- une table ***LignesFac*** (lignes de factures) contenant les quatre informations suivantes : **`NumLigne`** (numéro de la ligne ; clé permettant d'identifier un enregistrement de la table), **NumFacture** (numéro de la facture), **`RefArticle`** (référence de l'article commandé), **`Quantite`** (quantité des articles commandés dans la facture).

- Ecrire un script qui présente les articles de la base sous la forme d'un tableau HTML. Les informations à afficher sont : le nom de l'article, sa catégorie et son prix HT.
- Faire de même mais en présentant en rouge les articles dont le seuil critique du stock est atteint ou dépassé.
- Faire de même mais en présentant dans la même page un tableau par catégorie.
- Afficher les informations sur les articles avec le nom de la société qui les fournit. Le nom de l'article est un lien hypertexte qui, une fois cliqué, donne le détail du fournisseur correspondant.
- Ecrire une page de recherche d'articles. On donne une catégorie d'articles et une quantité et le script affiche les articles appartenant à cette catégorie ayant en stock au minimum la quantité demandée.

Problème traité via PHP

Énoncés :

Première partie : Système d'inventaire de services d'un club de sport

Il s'agit dans cette partie de créer dans l'arrière boutique un système d'inventaire de services qui permet à un gérant de répertorier toutes les disciplines sportives proposées dans son club. Les informations saisies à partir d'un formulaire HTML sont transmises et stockés dans une base de données.

Étape 1 :

Créez un formulaire HTML nommé **inventaireDisciplines.html** qui permet au gérant d'un club de sport d'indiquer dans des *champs* appropriés

1) la référence,
2) le nom du service (sport),
3) la catégorie du service (natation, sport d'hiver, exercices cardio-vasculaires, etc…)
4) les frais mensuels
5) les frais annuels
6) la durée d'enseignement de la discipline (3 , 6, 9 mois)

Ce formulaire doit être agrémenté d'un bouton «*Envoyer*!» qui initiera une insertion ou mise à jour d'une table appelée «**Inventaire**» de votre base de données.

Étape 2 :

a) Dans votre base, créez la table «**Inventaire**» dont les colonnes correspondent aux champs de votre formulaire HTML ci-dessus. De plus,

b) Concevez un *script PHP* pour stocker un à un dans la table **Inventaire**, 5 sports de 3 catégories différentes de votre choix.

c) Ensuite, pour vérifier que les produits sont bien stockés, prévoyez l'affichage d'un **message de confirmation d'ajout** et l'apparition d'une fenêtre dressant la **liste de toutes les disciplines existantes** dans la BD.

Deuxième partie : Réalisation d'un panier à provision (caddie)

Étape 1 :

Le but de cette étape est de vous amener *pas à pas* à l'implémentation d'un panier à provision dans un site de commerce électronique « C.E. » avec la technologie PHP version MySQL. Pendant la navigation d'un tel site, un sportif (visiteur du site de c.e.) peut à volonté **ajouter** des sports à son panier. À n'importe quel moment, ce sportif peut demander **l'affichage** du contenu courant de son panier et décider d'**enlever** des abonnements. L'utilisateur peut enfin **confirmer** l'abonnement aux sports qui existent dans son panier et terminer la session d'achat.

Après la construction d'une table d'inventaire (*Inventaire*) des sports, la prochaine étape consiste à créer une table «**Catalogue** » qui est utilisée pour conserver l'information temporaire (le temps d'une session) dans le panier à provision. Pour une grande partie, l'information dans la table **Catalogue** est un reflet de celle présente dans la table *Inventaire* (Première partie). Ces deux tables diffèrent dans leur nature et l'usage des informations d'arrière plan qu'elles contiennent.

Néanmoins, une information comme les frais d'abonnement est conservée seulement dans la table *Inventaire* et référencée par un script PHP si nécessaire. Cette approche est justifiée par le fait que les frais peuvent changer de saison en saison et la table *Catalogue* (panier) ne peut accommoder ces changements. De plus, la table *Catalogue* conserve l'identifiant unique du panier dans chaque enregistrement.

Étape 2 :

Il s'agit de créer une simple page Web qui affiche un tableau de sports agrémenté de boutons pour l'**ajout,** l'**affichage,** le **retrait** et la **validation** (ou confirmation) dans l'achat opéré par un client sportif.

Multimédia et Développement Internet (Niveau 4)
Département Informatique

☆☆☆☆☆

Supports et Références Bibliographiques

Langage de Script par Pierre MAURICE Directeur de recherche INRIAIRIT, université Paul-Sabatier, Toulouse.

Cours de KEBIR Mohamed Ines

Support de cours – Réseaux & Télécommunications de Eric BRASSART, Institut Universitaire de technologie d'Amiens

Média graphie

Sites Internet

[1]	**Titre**	Comment ça marche ? : Encyclopédie informatique libre
	URL	http://www.commentcamarche.net/
	Langue	Française

[2]	**Titre**	
	URL	php.developpez.com/cours
	Langue	Française

[3]	**Titre**	Le site du PHP Group
	URL	http://www.php.net
	Langue	Française

[4]	**Titre**	PHPDébutant
	URL	http://www.phpdebutant.com
	Langue	Française