

**REPUBLIQUE DU CAMEROUN**  
**PAIX – TRAVAIL - PATRIE**  
-----  
**UNIVERSITE DE DSCHANG**  
-----

**REPUBLIQUE OF CAMEROUN**  
**PEACE – WORK – FATHERLAND**  
**THE UNIVERSITY OF DSCHANG**

**INSTITUT UNIVERSITAIRE DE TECHNOLOGIE**  
**FOTSO VICTOR DE BANDJOUN**  
B.P. 134 Bandjoun (Cameroun)

# **SUPPORT DE COURS,** **TD ET TP**

## **DEVELOPPEMENT DES APPLICATIONS & TECHNOLOGIE WEB**

© M. NOULAMO THIERRY  
Enseignant à l'IUT FOTSO Victor de Bandjoun  
thierry.noulamo@gmail.com

Année académique 2023-2024

---

## SOMMAIRE

### Chapitre 1 La Technologie Web

1 Le WEB (Présentation et Historique).....	6
2 Exemple d'application réparti .....	6
3. L'architecture client-serveur .....	7
3.1 Définitions de bases .....	7
3.2 Client-Serveur sur le réseau d'entreprise .....	8
3.3 Client-serveur sur le WEB .....	9
3.3.1 Principe du C/S HTML/WEB : .....	9
3.3.2 Principe du C/S à composants distribués : .....	10
4. Quelques Technologies du Web dynamique .....	10
4.1 Serveurs d'Application.....	10
4.3 Le navigateur HTML .....	11
4.2.1 Les fonctions de base du navigateur Web .....	11
4.2.2 Un contenu multimédia .....	11
4.2.3 Intégration des protocoles Internet .....	11
4.2.4 Quelques Navigateurs disponibles .....	11
5. URL : Uniform Resource Locator.....	12
6. Structure d'un site .....	12

### Chapitre 2 Le Langage HTML

1. Introduction .....	13
2. La syntaxe HTML : balises .....	13
3. L'interprétation des pages HTML.....	14
4. Structure de base d'un document HTML.....	14
5. Balises de mise en page.....	15
6. Les styles .....	16
7. Les images.....	21
8. Ancres et liens hypertextes.....	21
9. Les tableaux.....	22
10. Scinder une page en blocs .....	23
11. Les formulaires HTML .....	24

### Chapitre 3 Le Langage XML

1- Introduction à XML .....	27
1.1- Présentation de XML .....	27
1.2 Voici les principaux atouts de XML :	
2- Structure d'un document XML.....	28
2.1- Qu'est-ce que le XML? .....	28
2.2- Structure d'un document XML.....	28
2.3- La syntaxe des éléments en XML.....	29
3- Présentation des DTD .....	30
3.1- Le rôle de la DTD .....	30
4- Déclaration d'entités dans les DTD.....	32
4.1- Déclarer des entités .....	32
5- Les espaces de nom XML.....	33
5.1- Introduction aux espaces de noms .....	33
6- Mise en page de XML avec XSL.....	33

---

---

6.1- Introduction à XSL .....	33
6.2- XSLT et XSL/FO .....	34
6.3- Structure d'un document XSL .....	34
7- Transformations de XML avec XSLT .....	35
7.1- XSLT et XSL/FO .....	35
8- Création de liens XML avec XLL .....	36
8.1- Introduction à XLL .....	36
<b>Chapitre 4 JavaScript</b>	
1. Introduction .....	37
2. Le Javascript minimum .....	37
2.1. La balise <SCRIPT> .....	37
2.2. Les commentaires .....	37
2.3. Masquer le script pour les anciens browsers .....	38
2.4 Où inclure le code en Javascript ? .....	38
2.5 Une première instruction Javascript .....	38
2.6 Votre première page Html avec du Javascript .....	39
2.7 Quelques Remarques .....	39
2.8 Versions du langage Javascript .....	39
3. Utiliser des variables .....	40
3.1 La déclaration de variable .....	40
3.2 Les données sous Javascript .....	40
4. Les opérateurs .....	41
4.1 Les opérateurs arithmétiques .....	41
4.2 Les opérateurs de comparaison .....	41
4.3 Les opérateurs associatifs .....	41
4.4 Les opérateurs logiques .....	41
4.5 Les opérateurs d'incrémentation .....	42
4.6 La priorité des opérateurs Javascript .....	42
Opération            Opérateur .....	42
5. Les fonctions .....	42
5.1 Déclaration des fonctions .....	42
5.2 L'appel d'une fonction .....	43
5.3 Les fonctions dans <HEAD>...<HEAD> .....	43
6. Les événements .....	43
6.1 Les gestionnaires d'événements .....	44
6.2 Exemple .....	44
6.3. Gestionnaires d'événement disponibles en Javascript .....	44
7. Les Structures de contrôles .....	45
7.1 les conditions .....	45
7.2 L'expression for .....	45
7.3 While .....	45
8. Les formulaires .....	45
9. Les boites de dialogue ou de message .....	46
9.1 La méthode alert() .....	46
9.2 La méthode prompt() .....	46
9.3 La méthode confirm() .....	47

---

10. L'objet String .....	47
11. L'objet Math .....	47
<b>Chapitre 5 AJAX</b>	
4.1 Introduction .....	48
4.1.1 But d'Ajax.....	48
4.1.2 En quoi consiste Ajax? .....	48
4.2 Ajax et DHTML .....	49
4.2.1 L'objet XMLHttpRequest.....	49
4.2.3 Construire une requête, pas à pas .....	50
4.3 Exemple de programme Ajax.....	50
4.4 Comment Créer un site Ajax? .....	53
4.5 Inconvénients d'Ajax .....	53
4.6 Quelques Outils .....	53
<b>Chapitre 6 Configuration PHP / APACHE / MYSQL</b>	
Configuration d'APACHE :.....	54
LoadModule status_module modules/mod_status.so.....	55
Configuration PHP .....	56
Configuration de MySQL.....	58
<b>Chapitre 7 Le Langage PHP</b>	
1. Introduction .....	59
2. Principe des applications web .....	59
3. Notions de base .....	62
4. Structures de contrôle.....	62
5. Interactions avec l'utilisateur .....	63
6. Formulaire validé par lui-même .....	64
7. Tableaux .....	66
8 Passage et transmission de variables .....	67
8.1-Passage et transmission de variables par formulaire.....	68
9.1- Ouverture / fermeture de fichier .....	70
9.2- Lecture de fichier .....	71
9.3- Fonctions diverses de traitement de fichier : .....	73
10 Variables persistantes: Cookies et Session.....	73
10.1- les Cookies .....	74
10.2- les sessions .....	75
11. PHP et les bases de données.....	77
11.1- Utilisation d'une base de données MySql .....	77
11.2- Concepts fondamentaux : Bases, tables, champs, enregistrements.....	78
11.3- Administration de MYSQL.....	78
11.4- SQL : petit récapitulatif du langage .....	78
11.5- Accéder à MYSQL via PHP .....	79
<b>Chapitre 8 Editeur HTML (Dreamweaver)</b>	
1. Introduction .....	82
2. Présentation des fenêtres et des panneaux.....	82
3. Présentation des menus .....	84
4. Affichage d'un exemple de site .....	85
5. Définition d'un site local .....	85

---

6. Création et enregistrement d'une nouvelle page.....	89
7. Mise en forme avec Dreamwever (TP) .....	92
8. Ajout de styles au texte .....	92
<b>Exercices</b>	
Exercice 20 .....	98

# Chapitre 1 La Technologie Web

## 1 Le WEB (Présentation et Historique)

- Invention récente :
  - 1992
  - au CERN
  - par Tim Bernes Lee
- Au départ :
  - Sites documentaires
  - Liens hypertextes
- Technologie HTTP et HTML

## 2 Exemple d'application réparti

Description d'un processus de preinscription en ligne à l'Université de PARIS 13.

Pour ce préinscrire à l'université de PARIS 13, les phases suivantes sont nécessaires :

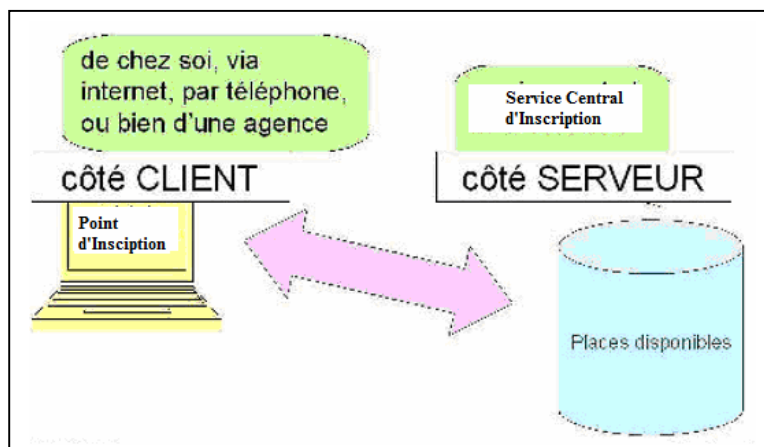
- Accéder au site de l'Université
- Remplir le formulaire d'inscription et le soumettre
- A la réception des informations, une première confirmation est effectuée en temps réel
- Après étude du dossier de candidature, la réponse d'acceptation est renvoyée au candidat
- Le processus d'inscription s'achève

La mise sur pied d'un tel module d'inscription nécessite une étude qui intègre la phase

- D'analyse
- De conception
- Et d'implémentation.

Les connaissances acquises dans les enseignements précédant (SI, UML, BD, GL, ...) couvrent les deux premières phases et restent valables dans ce cours.

Dans le cadre de ce cours il sera question de développer une application tel que celle décrite ci-dessus. La figure ci-dessous donne une idée du déploiement d'une telle application.



Notre futur candidat, a parti de sont ordinateur qui se trouve soit à domicile ou dans un cyber café accède au site de l'Université via le réseau Internet. Ceci n'est possible que si l'application développée est répartie. La mise sur pied de telles applications utilise généralement le modèle client/serveur.

### **3. L'architecture client-serveur**

Depuis le début des années 90, les réseaux LAN (Local Area Network) et WAN (Wide Area Network) sont devenus l'épine dorsale du système informatique des entreprises en s'appuyant sur la technologie Internet.

Depuis le milieu des années 90, cette technologie est mise à la disposition du grand public par le WEB.

Les applications client-serveur sont donc devenu une activité essentielle pour un grand nombre de personnes partout dans le monde.

Les architectures client-serveur sont un modèle d'architecture où les programmes sont répartis entre processus clients et serveurs communiquant par des requêtes avec réponse.

#### **3.1 Définitions de bases**

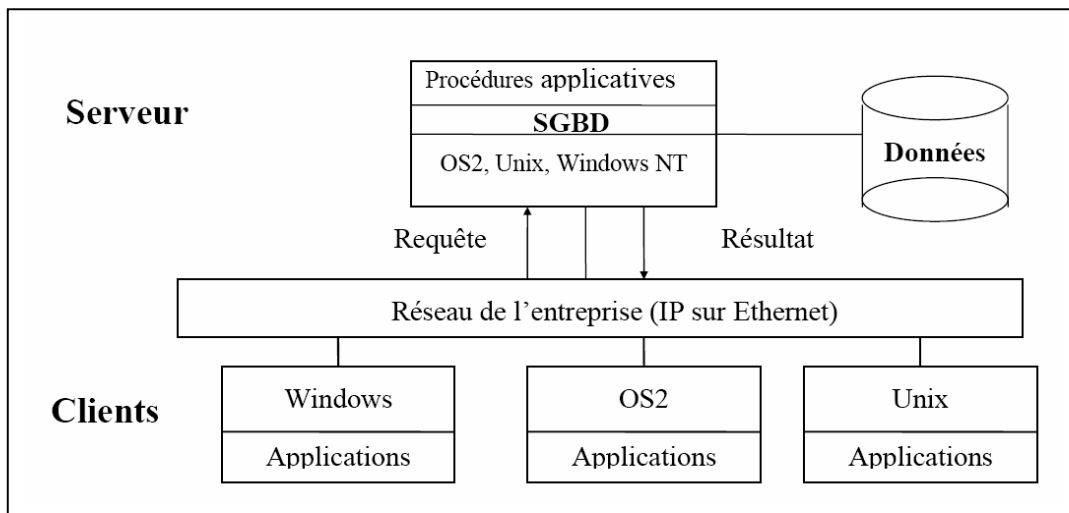
##### **Client**

- Processus demandant l'exécution d'une opération à un autre processus par envoi d'un message (requête) contenant le descriptif de l'opération à exécuter et attendant la réponse à cette opération par un message en retour.
- Il supporte une partie du code de l'application. Le code implémente au minimum les dialogues avec les utilisateurs et l'affichage des résultats (GUI : Graphical User Interface).

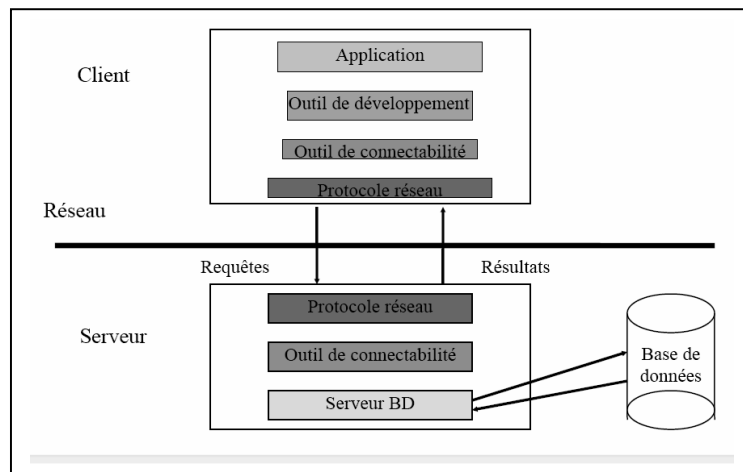
##### **Serveur**

- Processus accomplissant une opération sur demande d'un client et transmettant la réponse à ce client.
- Il assure le stockage, la distribution, la gestion de la disponibilité et de la sécurité des données. Ces fonctionnalités sont offertes par le SGBD résidant sur le serveur ou accessibles sur un autre serveur connecté au réseau. Il supporte une partie du code de l'application : un certain nombre de traitements sur les données peuvent être regroupés et exécutés sur le serveur.

### 3.2 Client-Serveur sur le réseau d'entreprise



- Cette architecture supporte des machines hétérogènes (Windows 3.11, Windows 95 ou 98, Windows NT, Mac OS, Unix, ...)
- Répartition des traitements entre clients et serveurs
- Relative indépendance vis à vis du choix des protocoles réseau et des systèmes d'exploitation sur les serveurs et les clients. Les vendeurs de logiciel doivent aujourd'hui avoir cette flexibilité.



**Outil de connectabilité**, généralement appelé le middleware, est le logiciel qui implémente le protocole qui permet au processus client de dialoguer avec le processus serveur (ODBC, SQL Net, ...).

#### Outils de développement :

- langages de 3ème génération : programmes C, Cobol appelant des bibliothèques du SGBD.



- langages de 4ème génération : ils offrent des composants logiciels qui permettent de développer très rapidement une application (Access, PowerBuilder, Uniface, ...) et intègrent un langage de programmation en général interprété.

La technologie C/S est diffusée depuis 10 ans dans les entreprises. Le retour d'expérience est : l'administration du C/S coûte cher !!

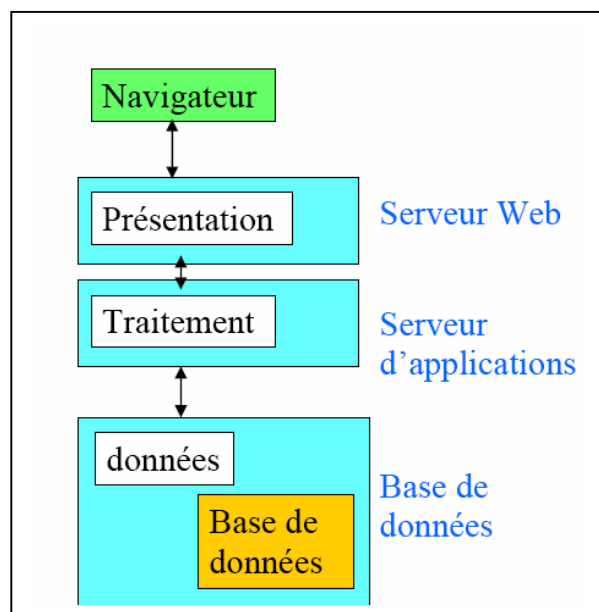
- Il est parfois nécessaire de fabriquer une version d'application par type de plateforme
- En cas de mise à jour, il faut déployer la mise à jour sur tous les clients
- Sur chaque client, lorsque l'on installe un logiciel, il faut installer le ou les middlewares qui lui permet(tent) de communiquer avec des serveurs.
- Peu de développeurs maîtrisent réellement l'ensemble des technologies mise en oeuvre dans une application C/S.

### 3.3 Client-serveur sur le WEB

Internet et Java sont les solutions largement utilisées pour réduire ces coûts .

Le petit client chétif est un PC équipé d'un navigateur Internet. Les applications sont des pages HTML téléchargées sur le PC.

On parle dans ce cas d'architecture three-tiers car le serveur WEB et le serveur d'application jouent le rôle d'intermédiaires entre le client et le serveur de la base de données.



On peut distinguer deux types de client-serveur sur Internet :

- le C/S HTML/WEB (CGI : Common Gateway Interface, servlets Java),
- le C/S à composants distribués (applets Java, Active X).

#### 3.3.1 Principe du C/S HTML/WEB :

L'utilisateur saisit des informations dans un formulaire HTML. Le navigateur envoie une requête HTTP au serveur WEB. Celui-ci contient le nom d'un programme exécutable depuis le serveur ainsi que les informations saisies dans le formulaire. Le programme est exécuté sur le serveur

WEB (ou un autre serveur du réseau): il interroge la base de données et renvoie au navigateur la réponse formatée en HTML.

**Avantages :**

- Une seule version des applications (sur le serveur)
- Mise à jour de l'application uniquement sur le serveur
- Compatible avec les modèles de sécurité (firewall, ...)

**Inconvénients :**

- La gestion du contexte de session est plus lourde à gérer que dans la solution composants distribués.

### **3.3.2 Principe du C/S à composants distribués :**

Le navigateur WEB télécharge un composant logiciel (Active X ou applet Java) stocké sur le serveur WEB. Le composant s'exécute sur le navigateur. On peut à nouveau distinguer deux types d'architectures :

- **Fat client** : l'applet gère l'ensemble des traitements (interface graphique, connexion BD, traitements spécifiques de l'application). Il établit la connexion avec la base de données en utilisant le serveur WEB comme serveur de connexion (JDBC, SQLJ).
- **Thin client** : l'applet gère l'interface graphique. Les traitements sont répartis entre client et serveur. L'application est distribuée sur le réseau, mais le programmeur peut accéder aux objets gérés par le serveur en manipulant des pointeurs comme s'il s'agissait d'objets locaux. Trois protocoles actuellement permettent d'utiliser ce type d'architecture : RMI (Remote method invocation), DCOM (Distributed Component Object Model), et CORBA (Common gObject Request Broker Architecture)

**Avantages :**

- Une seule version des applications (machine virtuelle Java sur chaque plateforme).
- Les mises à jour sont faites uniquement sur le serveur.

**Inconvénients :**

- Incompatibilité avec les modèles de sécurité lors du déploiement
- Installation de la Java Runtime sur les postes clients.
- Java est un langage interprété, problèmes de performance possibles.

Nous traitons dans le cadre de ce cours le client/serveur HTML/WEB.

## **4. Quelques Technologies du Web dynamique**

### **4.1 Serveurs d'Application**

- Microsoft ASP (.NET)
  - Active Server Page
  - Fonctionne uniquement sous Windows

- Java : Servelets/JSP
  - Programmation Objet
  - Richesse des interfaces
- PHPg
  - Communauté libre
  - Très populaire pour les sites Internet
  - Simple à apprendre

### **4.3 Le navigateur HTML**

Le navigateur ou browser est avant tout un client HTTP. C'est grâce à ce logiciel que l'utilisateur visualise les informations qu'il télécharge depuis le Web.

#### **4.2.1 Les fonctions de base du navigateur Web**

Il émet les requêtes (Queries) HTTP à destination des serveurs Web.

Il interprète les documents HTML qui lui sont envoyés en retour.

Il fournit des outils destinés à faciliter la navigation (gestionnaire de Bookmarks, Historique des visites,...).

#### **4.2.2 Un contenu multimédia**

Tous les navigateurs peuvent afficher du texte formaté en HTML et des images au format GIF ou JPG ; La plupart des navigateurs ne se limitent plus à l'interprétation des pages HTML.

Par l'intermédiaire d'extensions (Plug-ins), le navigateur Web peut désormais traiter tout type de documents multimédias : son, images animées...

Bien que non officiellement standardisées, ces extensions acquièrent parfois un statut de standard de fait (plug-in RealAudio pour la diffusion de fichiers sonores).

Le navigateur Web est aujourd'hui perçu comme l'interface universelle et standardisée d'accès aux ressources du réseau.

#### **4.2.3 Intégration des protocoles Internet**

Le navigateur Web n'est plus un simple client HTTP.

La plupart des navigateurs du marché sont aussi des clients d'autres applications Internet : SMTP (Messagerie électronique), FTP (transfert de fichiers), NNTP (news), ...

#### **4.2.4 Quelques Navigateurs disponibles**

- Microsoft Internet Explorer
- Mozilla
- Opera

## 5. URL : Uniform Resource Locator

Utilité

URL signifie : Uniform Resource Locator

Une URL identifie de manière unique une ressource dans un réseau internet ou intranet.

Différents types d'URL

**Consultation** : <http://www.inapg.inra.fr/etudiants/site/index.php>

**Téléchargement** : <ftp://ftp.mozilla.org>

**Mail** : <mailto:webmaster@inapg.inra.fr>

Forme générale

protocole:numéro\_de\_port//nom\_ou\_adresse\_serveur/emplacement

le protocole indique la syntaxe d'échange entre le client et le serveur ;

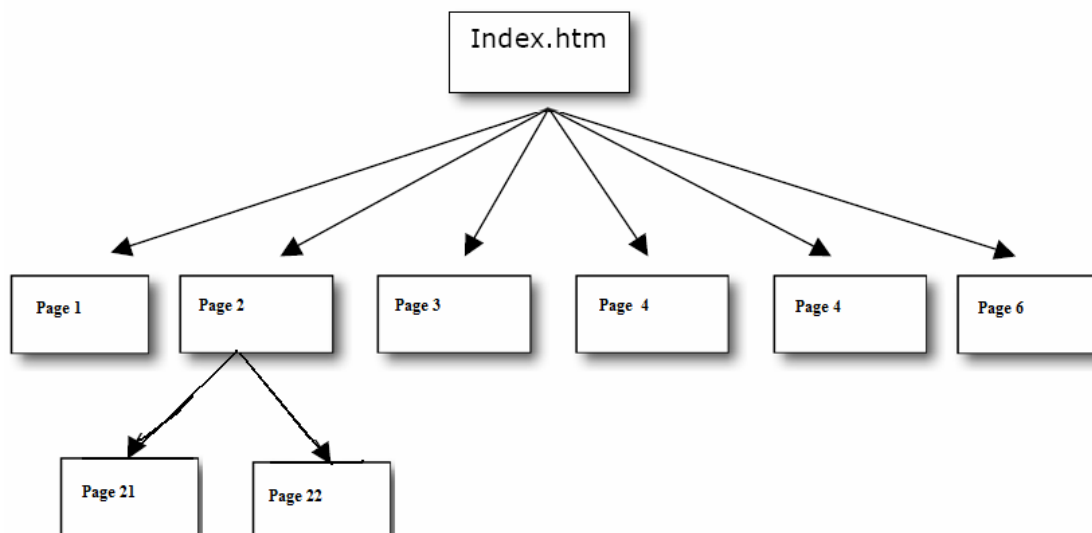
le numéro de port indique un chemin d'écoute particulier sur le serveur (par défaut : 80 pour http, 21 pour ftp, etc.) ;

le nom/l'adresse du serveur indique comment le retrouver sur le réseau l'emplacement indique l'emplacement exact de la ressource demandée sur le serveur.

## 6. Structure d'un site

Tous les fichiers que vous souhaitez utiliser dans votre site WEB doivent être stockés dans le même répertoire. Ce répertoire est appelé répertoire racine du site. C'est lui qui sera indexé par le serveur web. Des sous-répertoires pourront être créés, chaque sous-répertoire contenant un ensemble de fichiers logiquement proches. Par exemple, on regroupera toutes les images dans un même sous-répertoire.

La structure de répertoire permet ainsi de mieux organiser votre site. On obtiendra une structure de site arborescente telle que l'indique la figure ci-dessous.



## Chapitre 2 : HTML5/CSS3

# Introduction

Historique

Organismes de normalisation

Définition du HTML5

Principe du HTML5

Quand choisir HTML5/CSS3

## Historique



- **1962** : US Air Force commande la création d'un réseau de communication militaire pouvant résister à une attaque nucléaire.
- **1964** : [Paul Baran](#) crée un réseau en forme de toile, évitant ainsi un système central vulnérable.
- **1969** : ARPANET est créé pour relier 4 universités, indépendamment d'un objectif militaire. (Présentation public en 1972)
- **1971** : 1er Mail électronique. Le caractère @ est déjà présent. (Amélioration en 1972 avec boîte mail possédant un système d'archivage, de tri et « faire suivre »)
- **1976** : Déploiement du protocole **TCP** (débuté en 1972) permettant la transmission de paquet et la gestion des erreurs. Il sera scindé en 2 protocoles en 1978 : **TCP/IP**
- **1980** : [Tim Berners-lee](#) et [Robert Caillau](#) mettent au point un système de navigation Hypertext. (ancêtre des navigateurs)
- **1984** : Mise en place du système de nommage **DNS**. (Domain Name System-système de noms de domaine) C'est un service permettant de traduire un nom de domaine en informations de plusieurs types qui y sont associées, notamment en adresses IP de la machine portant ce nom.
- **1986** : Création du langage **SGML** (Standard General Markup Language) pour structurer des contenus divers, considéré comme le 1er langage à balise.
- **1991** : [Tim Berners-lee](#) met au point le protocole **HTTP** (Hyper Text Transfer Protocol), ainsi que le langage **HTML** (Hyper Text Markup Language). Naissance du **World Wide Web** (WWW).
- **1996** :
  - 1ère spécification HTML 2.0 par le **W3C** (World Wide Web Consortium). le HTML 1.0 n'a jamais vraiment existé.
  - Apparition du **CSS** (Cascade Style Sheet – feuille de style) créé par [Håkon Wium Lie](#).
- **De 1997 à 1999** :
  - Spécification HTML 3.2, 4.0 et 4.01
  - Spécification CSS 2.0
  - Création du **XML** (eXtensible Markup Language)
- **2000** : Le W3C lance le XHTML 1.0, celui-ci possède exactement le même contenu que le html 4.01 la différence tient sur la syntaxe, le xhtml suivait les règles du xml. Exemple tous les attributs d'une balise ne s'exprime plus qu'en minuscule. (cette sortie coïncida avec l'essor des navigateurs compatible avec CSS).
- **2001** : Le W3C recommande le XHTML 1.1, celui-ci est entièrement en XML, cependant gros problème le navigateur le plus populaire du moment ne peut pas l'interpréter (ie explorer). Le W3C perd pied avec la réalité des publications web. Ceci ne l'empêche pas de se lancer dans le XHTML 2.0 cependant celui-ci possède plusieurs problèmes techniques et ne répond toujours pas aux besoins développeurs du moment.
- **2004** : Scission au sein du W3C. Les représentants d'Opéra, Apple et Mozilla ne sont pas en

accord avec les priorités du W3C. Proposition de reprise du développement HTML pour création d'application web mais celle-ci est rejetée. Les représentants d'Opéra (dont *Ian Hickson*) et autres forment leur propre groupe WHATWG (Web Hypertext Application Technology Working Group)

- **2006** : *Tim Berners-Lee* admet que la migration du HTML vers XML était une erreur. Quelques mois plus tard le W3C planche sur la version HTML 5 (avec espace) et continue cependant sur le XHTML 2.0. De son côté WHATWG travaille sur le HTML5 (sans espace).
- **2007** : Le W3C accepte les propositions de recommandations du WHATWG sur le HTML5 (sans espace).
- **2009** : Le format XHTML 2.0 est définitivement mort
- **2012** : La spécification HTML5 doit devenir « recommandation candidate », c'est-à-dire fin prête dans le discours normatif.

## Organismes de normalisation

### IETF

L'Internet Engineering Task Force, abrégée IETF, littéralement traduit de l'anglais en « Détachement d'ingénierie d'Internet » est un groupe informel, international, ouvert à tout individu, qui participe à l'élaboration de standards Internet. L'IETF produit la plupart des nouveaux standards d'Internet. L'IETF est un groupe informel, sans statut, sans membre, sans adhésion. Le travail technique est accompli dans une centaine de groupes de travail. En fait, un groupe est généralement constitué d'une liste de courrier électronique. L'IETF tient trois réunions par année.

### W3C

Le World Wide Web Consortium, abrégé par le sigle W3C, est un organisme de normalisation à but non-lucratif, fondé en octobre 1994 chargé de promouvoir la compatibilité des technologies du World Wide Web telles que HTML, XHTML, XML, RDF, SPARQL, CSS, PNG, SVG et SOAP. Fonctionnant comme un consortium international, il regroupe en 2012, 378 entreprises partenaires.

Le W3C a été fondé par Tim Berners-Lee après qu'il eut quitté le CERN en octobre 1994. Le W3C a été fondé au MIT/LCS (Massachusetts Institute of Technology / Laboratory for Computer Science) avec le soutien de l'organisme de défense américain DARPA et de la Commission européenne.

### WHATWG

Le Web Hypertext Application Technology Working Group (ou WHATWG) est une collaboration non officielle des différents développeurs de navigateurs web ayant pour but le développement de nouvelles technologies destinées à faciliter l'écriture et le déploiement d'applications à travers le Web. La liste de diffusion du groupe de travail est publique et ouverte à tous. La Mozilla Foundation, Opera Software et Apple, Inc. en sont les premiers contributeurs.

Ce groupe de travail se limite aux technologies qu'il estime imprésentables dans les navigateurs Web sur la base des implémentations actuelles, et particulièrement de celles d'Internet Explorer. Il se présente notamment comme une réponse à la lenteur supposée du développement des standards par le W3C et au caractère supposé trop fermé de son processus interne d'élaboration de spécification. Cependant, de nombreux participants à ce projet sont également des membres actifs du W3C, et le nouveau groupe de travail HTML du W3C a adopté en 2007 les propositions du WHATWG comme base de travail d'un futur HTML5.

## Définition du HTML5

HTML5 est une combinaison de nouvelles balises HTML, de propriété CSS3, de JavaScript et de plusieurs technologies associées mais structurellement séparées de la spécification HTML5.

## Principe du HTML5

- HTML5 devra supporter le contenu déjà existant (pas d'an zéro du HTML5)
- S'il existe une façon répandue d'accomplir une tâche chez les webdesigners (même mauvaise), celle-ci doit être codifiée en HTML5 (« si ce n'est pas cassé, on répare pas »).
- En cas de conflit on privilégie d'abord l'utilisateur, les webdesigners, les implémenteurs (dans les navigateurs), les spécificateurs et enfin la beauté du code.

## Quand choisir HTML5/CSS3

Le W3C recommande dès à présent aux développeurs Web à utiliser HTML 5. De plus la majorité des navigateurs semblent avoir intégré les spécifications CSS3 et celui est le mieux à même à répondre aux nouveaux supports de diffusion du web (tablettes, téléphone...).

La question devrait plutôt être quand ne pas l'utiliser. Il existe encore une majorité d'administration travaillant encore avec des versions d'internet explorer antérieure à la version 9.0 dans ce cas mieux vaut rester dans les recommandations de L'HTML 4.0.1. Pour le reste foncez vers le HTML5.



# Le balisage

Structure d'une page

Contenus & nouvelle balise

Exemple: utilisation des nouveaux éléments

## Structure d'une page

### Les éléments de base

Le langage HTML5 est une amélioration du langage HTML4, avec des simplifications par rapport à la version XHTML. Tout document peut donc débuter par la déclaration du doctype puis est suivi de l'élément racine html qui inclut les éléments head et body.

#### Doctype

La déclaration du doctype est traditionnellement utilisée pour spécifier le type de balisage du document.

Celui de XHTML 1.0 était le suivant :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Celui de HTML 4.01 était :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01/EN" "http://www.w3.org/TR/html4/strict.dtd" >
```

Un des principes d'HTML5 étant la rétrocompatibilité avec les anciennes versions, sa définition se résume à un simple :

```
<!DOCTYPE html>
```

#### Html

L'élément **<html>** est l'élément racine du document. Il est le parent de tous les autres, soit **<head>** et **<body>**. Celui-ci possède des attributs dont le plus utiles est **lang**. Celui-ci indique la langue utilisée par défaut dans la page. Cette valeur sera reconnue par les moteurs de recherche pour leur permettre d'indexer les pages du site en effectuant un tri par langue.

```
<html lang= « fr » >
```

#### Head

Cette section donne quelques informations générales sur la page, comme son titre et l'encodage de la page (pour la gestion des caractères spéciaux). Ces informations sont les 2 seules obligatoires dans cette section.

Si la balise **<title>** n'a pas changé, la déclaration de l'encodage a elle été simplifiée. Si avant nous avions :

```
<meta http-equiv=»Content-Type» content=»text/html; charset=utf-8» />
```

En HTML5 :

```
<meta charset= « utf-8 »>
```

## Minimum Page HTML5

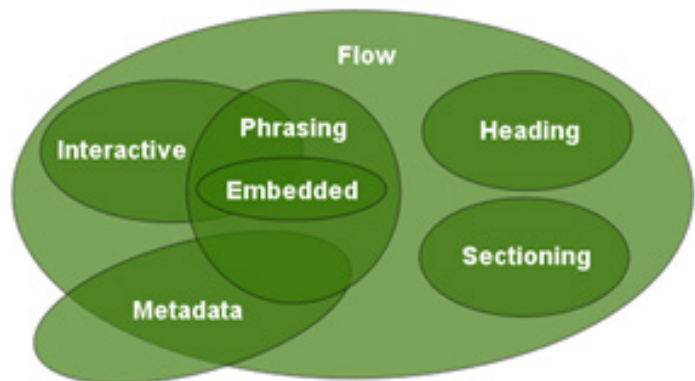
```
<!DOCTYPE html>
<html lang= « fr » >
  <head>
    <meta charset= « utf-8 »>
    <title>page simple HTML5</title>
  </head>
  <body>
    <!--ici contenu de ma page html5 -->
  </body>
</html>
```

## Contenus & nouvelles balises

### Les contenus

Dans les versions précédentes de HTML les contenus étaient divisés en 2 grandes catégories, « en lignes » ou en « bloc », celui-ci est remplacé par un nouveau schéma ; les éléments HTML sont regroupés selon les catégories suivantes :

- Contenu Metadonnées (*Metadata*)
- Contenu de flux (*Flow*)
- Contenu de section (*Sectioning*)
- Contenu d'en-tête (*Heading*)
- Contenu de phrasé (*Phrasing*)
- Contenu embarqué (*Embedded*)
- Contenu interactif (*Interactive*)



Il est à noter que certains éléments peuvent apparaître dans plusieurs catégories.

### Contenu de métadonnées

Les métadonnées déterminent la présentation ou le comportement du contenu de la page. Elles servent aussi à configurer les relations entre ce document et d'autres (exemple : balise **<meta>** contenant des mots clé ou bien un descriptif de la page).

Les modifications apportées par HTML5 concernent les balises **<link>** et **<script>**. Concernant la balise script il n'est plus nécessaire d'indiquer le **type**. Celui-ci sera par principe considéré comme étant du javascript.

```
<script src= « file/js » ></script>
```

Il en va de même pour l'appel des feuilles de styles, l'attribut réellement nécessaire étant le média sur lequel il s'applique :

```
<link rel= « stylesheet » href= « styles.css » media= « screen » />
```

Les différents média : all, print, screen, tv, aural, braille, embossed, handled, projection, tty

## Contenu de flux

Le contenu de flux représente les éléments qui sont considérés comme formant l'ensemble du contenu d'une page web. Un flux est en règle générale un texte ou un fichier embarqué comme une image ou bien une vidéo.

Cette catégorie possède plusieurs nouveaux éléments :

- **header** : spécifie une introduction, ou un groupe d'éléments de navigation pour le document.
- **footer** : définit le pied de page d'un article ou un document. Contient généralement le nom de l'auteur, la date à laquelle le document a été écrit et / ou ses coordonnées.

*Exemple* : voir exemple complet page suivante

- **figure** : définit des images, des diagrammes, des photos, du code, etc.
- **figcaption** : légende pour la balise <figure>.

*Exemple* : Création du document *00-figure.html*

Entre les balises **<body>** insérer le code suivant:

```
<figure >
  
  <figcaption>Photo promotion du film <strong>Les oiseaux</strong></figcaption>
</figure>
```



- **canvas** : utilisé pour afficher des éléments graphiques, il faut utiliser un script pour l'animer.
- Exemple : ces balises feront l'objet d'un chapitre complet

## Contenu de section

Ceci est une nouvelle catégorie. Le W3C décrit cette catégorie comme « définissant la portée des en-têtes et des pieds de pages ». Un contenu de section est un sous-ensemble d'un contenu de flux. Elle contient 4 nouveaux éléments HTML5 :

- **section** : définit les sections dans un document. Tels que les chapitres, en-têtes, pieds de page, ou toutes autres sections du document.
- **article** : partie indépendante du site, comme un commentaire.
- **aside** : associé à la balise qui le précède. Il pourrait être utilisé par exemple pour faire référence à des sujets complémentaires ou pour définir un terme spécifique.
- **nav** : définit une section dans la navigation. Il contient les liens utiles à la navigation

Exemple : voir exemple global

## Contenu d'en-tête

Contient tous les éléments d'en-tête standard comme **<h1>**, **<h2>** et ainsi de suite. En plus HTML5 comprend l'élément :

- **hgroup** : permet de définir des groupes de titres. Celui-ci ne peut contenir que les éléments de **<h1>** à **<h6>**. Il sert essentiellement à définir un plan, une table des matières.

*Exemple* : Création du document *01-hgroup.html*

Entre les balises **<body>** insérer le code suivant:

```
<hgroup>
  <h1>La vie des castors</h1>
  <h2>...ou la passionnante aventure d'une espèce en danger</h2>
</hgroup>
```

## La vie des castors

...ou la passionnante aventure d'une espèce en danger

### Contenu de phrasé

Ce contenu est le texte du document, y compris les mises en évidence de passage à l'intérieur d'un paragraphe. Nouvelles balise HTML5 :

- **mark** : définit un texte marqué.

*Exemple* : Création du document [02-mark.html](#).

Entre les balises **<body>** insérer le code suivant:

```
<p>L'avantage d'avoir de <strong>bonnes</strong> chaussures de marche de randonnée devient
<em>extrêmement</em> clair <mark>après 3 jours de marche</mark>.</p>
```

L'avantage d'avoir de **bonnes** chaussures de marche de randonnée devient *extrêmement* clair après 3 jours de marche

### Contenu embarqué

Un contenu embarqué importe une autre source dans la page, comme une image, une musique ou bien une vidéo. Ce contenu contient de nouvelles balises HTML5 permettant une alternative à FLASH :

- **audio** : pour définir un son, comme la musique ou les autres flux audio (streaming).
- **video** : Insérer un contenu vidéo en streaming.
- **track** : Insérer un sous-titre (au format WebVTT) à une vidéo affichée avec la balise video .
- **embed** : définit un contenu incorporé, comme un plug in.

*Exemple* : Ces contenus font l'objet du chapitre 7

### Contenu interactif

L'une des balises les plus basiques **<a>**, est considérée comme un élément interactif ainsi que les éléments de formulaire tel **<textarea>** ou encore **<button>**.

La balise **<a>** subit un profond changement. En effet si dans les versions précédentes cette balise est considérée inline :

Bienvenue !

```
<h2><a href= http://www.afci.fr> AFCl NEWSOFT </a></h2>
```

```
<p><a href= http://www.afci.fr> formation en micro-informatique diplomate 3d web pao bureau-
tique infographie video centre de formation informatique diplomante internet web </a></p>
```

En HTML5, il est possible d'envelopper plusieurs éléments dans un seul élément **<a>**

*Exemple* : Création du document [03-a\\_custom.html](#).

Entre les balises **<body>** insérer le code suivant:

```
<a href= http://www.afci.fr>
  <h2> APCI NEWSOFT </h2>
  <p>formation en micro-informatique diplomate 3d web pao bureautique infographie
video centre de formation informatique diplomante internet web.</p>
</a>
```

## Exemple: utilisation des nouveaux éléments

### Création page modèle

Nous allons dans cette partie créer une page modèle dont nous nous resservirons dans les prochains exercices. Celle-ci contiendra un simple en tête incluant une image ainsi qu'un titre et un sous-titre.



Création du fichier html de base : *modele\_page.html* (encodage UTF-8 (sans BOM))

```
<!DOCTYPE html>
<html lang= « fr » >
  <head>
    <meta charset= « utf-8 »>
    <title>Modèle page HTML5</title>
  </head>
  <body>
  </body>
</html>
```

Création de la feuille de style associé : *assets/css/base.css*

```
@charset «utf-8»;
/* CSS Document */
body {
  font: normal 90% «Trebuchet MS»,Verdana,»Lucida Grande»,Tahoma,Arial,Helvetica,Sans-Serif;
  color: #444;
```

```
background:#f4f4f4;
padding:0 0 2em 0;
margin:0;
}
```

Nous devons à présent attacher cette feuille de style à notre page html. Dans l'entête de notre page inclure le lien suivant:

```
<link rel="stylesheet" href="assets/css/base.css" media="screen" />
```

Enregistrer et visualiser le résultat.

La balise **<header>** :

La balise **< header>** (ainsi que **<footer>**) ont été créées afin de donner au créateur et au développeur des options de balisage sémantique plus précis inhérent à la structure d'un document. Ces balises donnent une indication plus précise du contenu d'une section (auparavant celle-ci était contenu dans un **<div>** dont l'id permettait de connaître son contenu ex : **<div id="header">**).

Dans le corps du document HTML :

```
<body>
  <header>
    
    <h1>HTML5</h1>
    <h2>Page modèle exercice</h2>
  </header>
</body>
```



Dans la feuille de style :

```
header{
  color: #ee662a;
  background:white;
  border-bottom:2px dotted #ccc;
  margin:0;
  padding:0.8em;
}

header h1{
  font-size: 4em;
  font-weight: normal;
  margin:0;
}

header h2{
  font-size: 1.5em;
  font-weight:bold;
  margin: 0 0 2.5em 10em;
  color: #e74b25;
}
```

```
header img{
  float:left;
  margin-right:3em;
}
```

Si l'on enregistre le document et qu'on le visualise dans les navigateurs les plus récents aucun problème (opéra, safari, chrome et ie9). Cependant si l'on essaie de le visualiser dans un navigateur antérieur celui-ci ne reconnaît pas la balise header (voir application **ietester**).

#### Correctif IE

Pour pallier cette erreur nous allons procéder en 2 temps :

*Création de la feuille de style associé : [assets/css/correctif\\_balise.css](#)*

```
@charset «utf-8»;
/* CSS Document */
header, section, aside, nav, footer, figure, figcaption{
  display:block;
}
```

*Puis création d'un document javascript : [assets/js/correctif.js](#)*

```
// JavaScript Document
document.createElement('header');
document.createElement('footer');
document.createElement('nav');
document.createElement('aside');
document.createElement('section');
document.createElement('figure');
document.createElement('figcaption');
```

Enfin il ne nous reste plus qu'à inclure ces 2 nouveaux fichiers au sein de l'en-tête de notre document :

```
<link rel="stylesheet" href="assets/css/correctif_balise.css" media="screen" />
<script src="assets/js/correctif.js"></script>
```

*Enregistrer et visualiser le résultat.*

A présent nous avons notre document de référence pour l'ensemble de nos exercices.

#### Exemple architecture de site:

Nous allons à présent dans l'exemple qui suit créer une page HTML contenant l'ensemble des nouvelles balises.

*Création du fichier html: [04-architectureSite.html](#) à partir de la page [modele\\_page.html](#)*


*Création de la feuille de style associé : [assets/css/styles\\_architecture.css](#)*

Puis modifier le titre de la page HTML («architecture site») et inclure la nouvelle feuille css:

```
<link rel="stylesheet" href="assets/css/styles_architecture.css" media="screen" />
```



Le contenu de la page sera défini de la manière suivante :



# HTML5

Page modèle exercice

header

ACCUEIL

A PROPOS

DIAPORAMA

CONTACT

nav

## Recommendations

aside

Donec libero eros, elementum vitae lacinia bibendum, malesuada et est. Pellentesque elit odio, lobortis non commodo quis, varius sed justo. Morbi aliquam purus eu leo bibendum et aliquet eros rhoncus. Ut feugiat porttitor magna, in mollis turpis ultricies non. Maecenas at tortor eros, et feugiat dolor. Pellentesque tincidunt dignissim commodo. Proin mi tortor, tincidunt sit amet mollis non, rhoncus ut est. Aliquam augue odio, lacinia sit amet hendrerit vel, bibendum eu arcu. Etiam dui leo, feugiat a molestie vel, tristique a ante.

Quisque fermentum scelerisque lacus eget sollicitudin. Nunc feugiat molestie odio vel accumsan. Etiam non est tortor. Morbi arcu odio, lacinia vitae aliquet in, vulputate vel ante. Curabitur mollis sapien non nunc semper sit amet ultricies nibh placerat. Cras facilis varius consectetur. Phasellus nibh odio, luctus auctor mattis vitae, aliquam et arcu.

## Mission du HTML5

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque vehicula, enim non mattis lobortis, libero est hendrerit erat, a tincidunt lacus sapien sit amet mi. Nulla fringilla lobortis porttitor. Mauris rutrum neque ac leo vehicula vel fringilla nisi consectetur. Suspendisse potenti. In libero metus, scelerisque et sodales et, convallis nec elit. Aenean est lectus, adipiscing nec feugiat sed, consequat ut nunc. Ut id felis vel purus venenatis bibendum.

### What's new?

Par David, le 05 septembre 2012

time

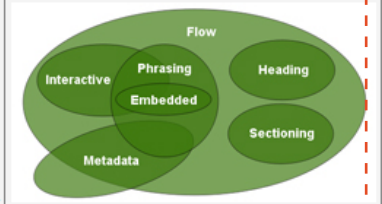
Donec libero eros, elementum vitae lacinia bibendum, malesuada et est. Pellentesque elit odio, lobortis non commodo quis, varius sed justo. Morbi aliquam purus eu leo bibendum et aliquet eros rhoncus. Ut feugiat porttitor magna, in mollis turpis ultricies non. Maecenas at tortor eros, et feugiat dolor. Pellentesque tincidunt dignissim commodo. Proin mi tortor, tincidunt sit amet mollis non, rhoncus ut est. Aliquam augue odio, lacinia sit amet hendrerit vel, bibendum eu arcu. Etiam dui leo, feugiat a molestie vel, tristique a ante.

#### Plus d'info:

aside

Les dernières recommandations du W3C.

lien



Les différents contenus HTML5

figure + figcaption

Fusce fermentum congue velit ac lobortis. Donec id erat a mauris sagittis dignissim id ut lorem. Suspendisse dignissim, massa sit amet vehicula mattis, nisi mauris rhoncus felis, ut sodales odio turpis a metus. Maecenas elementum semper enim. Quisque volutpat lectus at turpis porta eget volutpat arcu auctor. Sed egestas vehicula ipsum, quis tristique nibh et orci libero, a scelerisque augue. Suspendisse vestibulum cursus malesuada. Vivamus varius mauris vitae lorem auctor placerat. Vivamus dignissim suscipit pharetra. Proin et nulla elit.

mark

Copyright 2012 APCI NEWSOFT

footer

Copyright APCI NEWSOFT 2012

En vous inscrivant ou en utilisant ce site, vous acceptez notre contrat utilisateur et notre charte sur le Respect de la vie privée

footer

La balise <nav> :

ACCUEIL

A PROPOS

DIAPORAMA

CONTACT

Nous allons à présent créer notre menu de navigation. Sous la balise </header> inclure le code suivant:

17

```
<nav>
  <ul>
    <li><a href="#">ACCUEIL</a></li>
    <li><a href="#">A PROPOS</a></li>
    <li><a href="#">DIAPORAMA</a></li>
    <li><a href="#">CONTACT</a></li>
  </ul>
</nav>
```

Nous définissons à présent dans la feuille de style associée : *styles\_architecture.css*

```
/* navigation */
nav {
  background-color:#f2662a;
  height:35px;
}

nav li{
  float:left;
  width:140px;
  height:35px;
  background-color:black;
  text-align: center;
  border-left: 1px white solid;
  border-right: 1px white solid;
  line-height:35px;
  list-style:none;
}

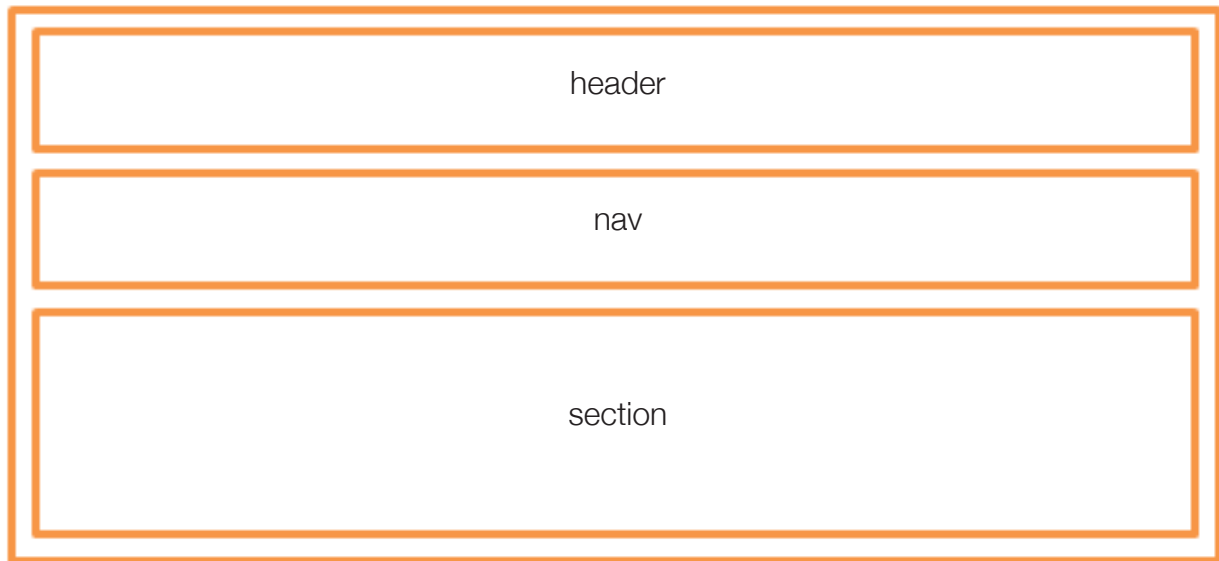
nav li a{
  color:white;
  text-decoration:none;
  display:block;
}

nav li a:hover{
  background-color:#f0462b;
  color:#ebebea;
}
```

*Enregistrer et visualiser le résultat.*

Il existe une alternative à la balise **<nav>** qui est la balise **<menu>**. Cependant celle-ci avait été déclarée obsolète pour le XHTML. Celle-ci est censée permettre la création rapide de menu déroulant, cependant les navigateurs gèrent très peu cette balise, surtout qu'il est possible de réaliser le même effet avec liste à puces et feuilles de style.

La balise <section> :



Une section est définie comme étant un regroupement thématique de contenu. Dans notre exemple celui-ci sera notre conteneur principal.

Ajoutons le contenu de notre section dans la page architecture (aidez-vous des textes lorem ipsum pour remplir celle-ci). A la suite de la balise **</nav>**

```
<section>
  <h2>Mission du HTML5</h2>
  <p>Lorem ipsum ... bibendum.</p>
  <p>Quisque ... arcu.</p>
  <h2>What's new?</h2>
  <p>Par David, posté le 05 Septembre 2012</p>
  
  <p>Les différents conenus HTML5</p>
  <p>Donec ... ante.</p>
  <p>Fusce ... elit.</p>
  <p>Quisque ... purus.</p>
</section>
```

Puis définition dans CSS associé

```
/* section */
section {
  width: 600px;
  float: left;
}

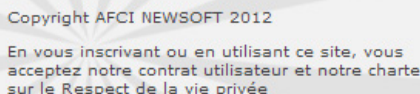
section p, h1, h2 {
  margin-left: 20px;
  margin-right: 20px;
}
```

```
section h2 {  
    margin-top:15px;  
}
```

*Enregistrer et visualiser le résultat.*

On peut remarquer simplement que section se comporte de la même manière qu'un **<div>**.

La balise **<footer>** :

A screenshot of a website footer. It features a light gray background with a subtle sunburst pattern. The text is in a small, sans-serif font. It reads: "Copyright AFCI NEWSOFT 2012" followed by "En vous inscrivant ou en utilisant ce site, vous acceptez notre contrat utilisateur et notre charte sur le Respect de la vie privée".

Copyright AFCI NEWSOFT 2012  
En vous inscrivant ou en utilisant ce site, vous acceptez notre contrat utilisateur et notre charte sur le Respect de la vie privée

Après la balise section nous ajoutons un pied de page sur l'ensemble du document.

```
<footer>  
    <p>Copyright AFCI NEWSOFT 2012</p>  
    <p>En vous enregistrant ou en utilisant ce site, vous acceptez notre contrat utilisateur et notre  
charte de Respect de la vie privée.</p>  
</footer>
```

Le CSS associé :

```
footer{  
    clear: both;  
    width: 400px;  
    margin-left: 20px;  
    font-size: 10px;  
    background: #f2f2f2 url(file:../images/footer_background.jpg) no-repeat left;  
    height: 118px;  
    padding-top: 10px;  
    border-top:#e2e2e2 solid 2px;  
    border-bottom:#e2e2e2 solid 2px;  
}
```

*Enregistrer et visualiser le résultat.*

La balise **<article>** :

La définition du W3C indique qu'il s'agit d'une composition autonome, indépendante du reste, dans un document, une application ou d'un site, et qu'elle est par principe distribuable ou réutilisable en soi. Il peut par exemple s'agir d'un post dans un forum, d'un article de presse, d'une entrée, d'un blog, d'un commentaire soumis par un utilisateur ou tout autre contenu autonome.

Le concept clé est qu'un article est un contenu *en soi*. Celui-ci peut être facilement republié dans différents contextes. Il pourra être multi-diffusé dans divers formats, tel qu'un flux RSS, dans un email, via des applications mobiles.

Repérer au milieu de votre **<section>** le 2ème **<h2>** et entourer ce contenu par la balise article.

```
<article>
  <h2>What's new?</h2>
  ...
  <p><small>© copyright 2012 AFCI NEWSOFT.</small></p>
</article>
```

En HTML5 un article peut posséder son propre en-tête et pied de page

```
<article>
  <header>
    <h2>What's new?</h2>
    <p>Par David, posté le 05 Septembre 2012</p>
  </header>
  ...
  <footer>
    <p>© copyright 2012 AFCI NEWSOFT</p>
  </footer>
</article>
```

Et dans le fichier CSS

```
/* article */
article header{
  background:none;
  height:auto;
  border-bottom:none;
}
article header h2{
  color:black;
  margin:0;
}

article header p{
  color: gray;
  margin: 0;
}
article footer{
  background:none;
  height:auto;
  border:none;
}
article footer p{
  font-size:10px;
}
```

Enregistrer et visualiser le résultat.

### La balise **<aside>** :

Cette balise peut être utilisée de 2 manières. L'une correspond au placement classique d'une barre latérale dans une page web. La seconde concerne une zone de contenu complémentaire, non indépendant, à l'intérieur d'une **<section>**.

1er cas :



Dans votre document html après la balise **</nav>**

```
<aside id=»main_aside»>
  <h3>Dernières recommandations</h3>
  <p>Nullam ... egestas.</p>
  <p>Quisque ..libero.</p>
</aside>
```

Dans le CSS :

```
/* aside */
#main_aside{
  margin-top:15px;
  float: left;
  width: 300px;
  background-color: #378059;
  padding:7px;
  color:white;
}
```

*Enregistrer et visualiser le résultat.*

2ème cas :

Dans la section **<article>** juste avant le 2ème gros paragraphe ajouter le code suivant

```
<aside id=»article_aside»>
  <h3>Plus d'infos ?</h3>
  <p>Lisez les recommandations du W3C. <a href=»#»>Lien.</a></p>
</aside>
```

Dans le CSS :

```
#article_aside{
  float: left;
  width: 12em; /* 1 em est la hauteur de caractère de la police utilisée, unité relative */
  background: #F9A890;
  padding: 10px;
  margin: 0 10px;
}
```

```
#article_aside h3{
    color:#378059;
    margin:5px 0 5px 0;
}
#article_aside p{
    margin: 0;
}
```

Enregistrer et visualiser le résultat.

#### La balise **<figure>**:

Parmi les nouveaux éléments certains donnent davantage de sens à nos pages web. Par exemple **<figure>** et **<figcaption>** permettent d'identifier des images et des légendes associées à l'intérieur de notre contenu.

Dans la partie **<article>** localiser la balise image et modifier le contenu pour obtenir.

```
<figure>
    
    <figcaption>Les différents contenus HTML5</figcaption>
</figure>
```

Dans le CSS :

```
/*figure */
figure {
    float: right;
    border: 1px solid gray;
    padding: 0.25em;
    margin: 0 1.5em 1.5em 0;
}

figcaption{
    text-align:center;
    font: italic 0.9em Georgia, «Times New Roman», Times, serif;
}
```

Enregistrer et visualiser le résultat.

#### La balise **<time>** :

Au début de **<article>** nous pouvons trouver une date de publication. Le problème avec ce format est que l'on ne l'indique pas dans un format lisible par un système informatique. HTML5 ajoute l'élément **<time>**. L'attribut `datetime` formaté selon l'année, le mois et le jour. S'il n'est pas obligatoire celui-ci sera reconnu par un système informatique. De plus si l'on publie le contenu, il est également possible d'ajouter l'attribut `pubdate`.

```
<p>Par David, posté le <time datetime="2012-09-05" pubdate>05 Septembre 2012</time></p>
```

# Récapitulatif balise

Nouveaux éléments

Nouveaux attributs

Changement balises et attributs

Obsolescence d'attributs



## Nouveaux éléments

- **section** : définit les sections dans un document. Tels que les chapitres, en-têtes, pieds de page, ou toutes autres sections du document.
- **article** : partie indépendante du site, comme un commentaire.
- **aside** : associé à la balise qui le précède.
- **header** : spécifie une introduction, ou un groupe d'éléments de navigation pour le document.
- **footer** : définit le pied de page d'un article ou un document. Contient généralement le nom de l'auteur, la date à laquelle le document a été écrit et / ou ses coordonnées.
- **nav** : définit une section dans la navigation.
- **figure** : définit des images, des diagrammes, des photos, du code, etc.
- **figcaption** : légende pour la balise <figure>.
- **audio** : pour définir un son, comme la musique ou les autres flux audio (streaming).
- **video** : Insérer un contenu video en streaming.
- **track** : Insérer un sous-titre (au format WebVTT) à une vidéo affichée avec la balise video .
- **embed** : définit un contenu incorporé, comme un plug in.
- **mark** : définit un texte marqué.
- **meter** : Permet d'utiliser les mesures avec un minimum et maximum connus, pour afficher une jauge.
- **progress** : définit une barre de progression sur le travail en cours d'exécution.
- **time** : définit une date ou une heure, ou les deux. Cette balise a été abandonnée en octobre 2011 en faveur de la balise data 2 avant d'être réintroduite<sup>3</sup>.
- **canvas** : utilisé pour afficher des éléments graphiques, il faut utiliser un script pour l'animer.
- **command** : définit un bouton.
- **details** : précise les détails supplémentaires qui peuvent être masqués ou affichés sur demande.
- **keygen** : permet de générer une clé (sécurisé).
- **output** : représente le résultat d'un calcul.
- **ruby, rt et rp** : annotations Ruby.

## Nouveaux attributs

Pour la balise <a>:

- **media** : permet de spécifier pour quel media ou device il est optimisé.
- **type** : définit le MIME de la cible URL.

Pour la balise <area> :

- **hreflang** : spécifie le langage de l'url.

- **media** : permet de spécifier pour quel media ou appareil il est optimisé.
- **rel** : Indique la relation entre le document courant et l'URL cible.
- **type** : définit le MIME de la cible URL.

Pour la balise `<button>` :

- **autofocus** : Indique que le bouton doit avoir le focus pendant le chargement de la page.
- **form** : spécifie à quel formulaire le bouton appartient.
- **formaction** : Spécifie où envoyer le form-data quand un formulaire est soumis. Remplace l'attribut action du formulaire.
- **formenctype** : Indique comment le form-data doit être encodé avant d'être envoyé à un serveur. Remplace l'attribut enctype du formulaire.
- **formmethod** : définit comment il faut envoyer le form-data.
- **formnovalidate** : si présent, indique que le formulaire ne doit pas être validé quand il est envoyé.
- **formtarget** : spécifie où ouvrir/exécuter l'action.

Pour la balise `<fieldset>` :

- **name** : définit le nom du fieldset.
- **disabled** : désactive le fieldset.
- **form** : définit le formulaire du fieldset.

Pour la balise `<form>` :

- **autocomplete** : auto complétion.
- **novalidate** : si présent le formulaire n'est pas validé lorsqu'il est soumis.

Pour la balise `<html>` :

- **manifest** : URL de déclaration (manifest) des fichiers pour un usage hors ligne.

Pour la balise `<iframe>` :

- **sandbox** : Spécifie des restrictions sur le contenu de l'iframe
- **seamless** : Indique que l'iframe doit être parfaitement intégrée dans le document.
- **srcdoc** : le code HTML du document affiché dans l'iframe.

Pour la balise `<input>` :

- **autocomplete** : auto completion.
- **autofocus** : définit le focus lors du chargement de la page.
- **form** : spécifie à quel formulaire le champ appartient.
- **formaction** : Remplace l'attribut "action" du formulaire. Indique l'URL à laquelle envoyer les données du formulaire.
- **formenctype** : Remplace l'attribut "enctype" du formulaire. Indique comment la forme-données doit être encodé avant d'être envoyé au serveur.

- **formmethod** : Remplace l'attribut "method" du formulaire. Définit la méthode HTTP d'envoi des données à l'URL.
- **formnovalidate** : Remplace l'attribut "novalidate" du formulaire. S'il est présent le champ de saisie ne devrait pas être validé lors de son envoi.
- **formtarget** : Remplace l'attribut "target" du formulaire. Indique la fenêtre cible utilisée lorsque le formulaire est soumis.
- **height** : Définit la hauteur.
- **list** : Désigne un "datalist" contenant des options prédéfinies pour le champ de saisie.
- **max** : Indique la valeur maximale du champ d'entrée.
- **min** : Indique la valeur minimale du champ d'entrée.
- **multiple** : Si présent, l'utilisateur peut entrer plusieurs valeurs.
- **pattern** : Définit un motif.
- **placeholder** : Un conseil pour aider les utilisateurs à remplir le champ de saisie.
- **required** : Indique que la valeur du champ de saisie est nécessaire pour soumettre le formulaire.
- **step** : Indique l'intervalle entre les valeurs.

nouveaux types :

- **datetime**
- **datetime-local**
- **date**
- **month**
- **week**
- **time**
- **number**
- **range**
- **email**
- **url**
- **search**
- **color**

Pour la balise **<link>** :

- **sizes** : Définit la taille, hauteur et largeur.

Pour la balise **<menu>** :

- **label** : Label visible du menu.
- **type** : Définit le type de menu à afficher. La valeur par défaut est « list ».

Pour la balise **<meta>** :

- **charset** : Définit la table de caractères pour l'encodage de la page.

Pour la balise **<ol>** :

- **reversed** : si présent, change l'ordre d'affichage.

Pour la balise **<script>** :

- **async** : définit si le script doit être exécuté de manière asynchrone ou pas.

Pour la balise **<select>** :

- **autofocus** : active le focus sur cet élément.
- **form** : définit un ou plusieurs formulaires pour le "select".

Pour la balise **<style>** :

- **scoped** : Si présent, le style est appliqué uniquement sur le parent et les fils.

Pour la balise **<textarea>** :

- **autofocus** : focus l'élément textarea.
- **dirname** : Indique le nom du textarea.
- **form** : définit une ou plusieurs formulaires pour le textarea.
- **maxlength** : nombre maximum de caractères.
- **placeholder** : définit une astuce pour aider l'utilisateur.
- **required** : Indique que la valeur du champ de saisie est nécessaire.
- **wrap** : définit comment le texte est affiché dans le textarea.

Ainsi que les attributs globaux qui s'appliquent à toutes les balises :

- **contenteditable**
- **contextmenu**
- **data-\***
- **draggable**
- **hidden**
- **on\*** (gestionnaires d'événements)
- **spellcheck**

## Changements dans les balises et attributs

Cette section est vide, insuffisamment détaillée ou incomplète. Votre aide est la bienvenue !

Les balises suivantes sont supprimées car leurs effets étaient purement représentatifs, ce qui est le rôle de CSS.

- *basefont*,
- *big*,
- *center*,

- *font*,
- *strike*,
- *tt*,
- *u*

Les balises *frame*, *frameset* et *noframes* ont été supprimées elles aussi ; elles étaient déjà désuètes car elles créaient des problèmes d'accessibilité et d'utilisation pour l'utilisateur final.

Les balises suivantes sont elles aussi supprimées :

- *acronym* n'est plus incluse car elle créait beaucoup de confusions;
- *applet* est remplacé par *object*;
- *isindex*, car elle peut être remplacée par l'utilisation des contrôleurs de formes;
- *dir* est obsolète en faveur de *ul*

Enfin, *noscript* n'est fourni que dans la version HTML, il n'est pas inclus dans la version XML.

## Obsolescence d'attributs

- *charset*
- *coords*
- *rev*
- *shape*

*Sur la balise area*

- *nohref*

*Sur la balise head*

- *nohref*

*Sur la balise html*

- *version*

*Sur la balise iframe*

- *longdesc*

*Sur la balise img*

- *longdesc*

- *name* (préférer l'attribut *id*)

*Sur la balise link*

- *charset*
- *rev*
- *target*

*Sur la balise meta*

- scheme

*Sur la balise object*

- archive
- classid
- codebase
- declare
- standby

*Sur la balise param*

- type
- valuetype

*Sur la balise td*

- axis
- scope

*Sur la balise th*

# Les formulaires

Composant d'un formulaire

Place holder, required, pattern & validation

Les nouveaux types `<input>`

Les formulaires comptent parmi les plus anciens et les plus familiers exemples d'interactivité sur le web. L'élément FORM a été introduit dans le langage HTML en 1993 et les contrôles associés font partie de l'environnement quotidien de l'utilisateur.

Les nouveaux composants HTML5 ajoutent des fonctionnalités que les concepteurs web incorporaient traditionnellement par d'autres moyens, comme javascript ou flash. Il est courant dans un formulaire de rendre tel ou tel champ obligatoire, d'en vérifier le contenu avant de soumettre un formulaire. Ces fonctionnalités sont maintenant intégrées à HTML5.

## Composant d'un formulaire

Nous allons créer un simple formulaire permettant de recevoir un nom et un prénom :

A partir du modèle de page créé précédemment, insérer le formulaire suivant

*Création du fichier html: [chap\\_4/05-formulaire\\_simple.php](#) (attention à l'extension) à partir de la page [modele\\_page.html](#)*

Insérer après la balise `</header>`

```
<div id="wrap">
  <form id="mon_formulaire" action="" method="post">
    <label for="nom">Nom : </label><input type="text" name="nom" id="nom" /> <br/>
    <label for="prenom">Prénom : </label><input type="text" name="prenom"
id="prenom" />
    <br/>
    <input type="submit" value="Valider" />
  </form>
</div>
```

Dans le fichier de base CSS : [assets/css/base.css](#) ajoutez

```
/* formulaire */
#wrap {
  padding: 0 3em 3em 3em;
  margin:auto;
}
```

La balise formulaire contient 3 attributs :

- **id** : permet d'associer un style CSS ou bien servira d'identifiant pour un traitement (js ou autre)
- **action** : spécifie l'agent traitement le formulaire. Cette valeur est l'URL qui pointe vers un script serveur, celui-ci récupère les valeurs et effectue un traitement
- **method** : (post/get) précise le mode d'envoi des données, ici POST.

Nous avons ensuite deux éléments `<input>` de type **text**, ceux-ci représentent les champs dans lesquels seront saisies les valeurs de internautes. Ils possèdent 2 attributs supplémentaires :

- **id**
- **name**

Ces 2 attributs devront toujours être présents afin de faciliter le traitement des informations que cela soit en javascript ou en PHP. (On peut remarquer qu'il possède la même valeur)



Et enfin il reste un dernier élément **<input>** celui-ci étant de type **submit**. C'est lui qui donnera l'ordre d'envoi du formulaire au serveur lorsque l'on cliquera dessus.

*Nous avons à présent le minimum requis pour avoir un formulaire.*

Nous allons tout de même un peu customiser notre formulaire en ajoutant l'élément **<fieldset>** permettant de regrouper des éléments de formulaire et y ajouter une légende.

```
<fieldset>
  <legend> Informations personnelles </legend>
  <label for=»nom«>Nom : </label><input type=»text« name=»nom« id=»nom« />
  <br/>
  <label for=»prenom«>Prénom : </label><input type=»text« name=»prenom« id=»prenom« />
</fieldset>
```

Puis dans le css:

```
#mon_formulaire {
  margin-top: 5px;
  padding: 10px;
  width: 400px;
}

#mon_formulaire label{
  float:left;
  font-size:13px;
  width:110px;
}
```

Il est à noter que la balise `<form>` s'enrichit de 2 nouveaux attribut :

- **autocomplete** : (on/off) permet de lancer la saisie semi-automatique
- **novalidate** : (on/off) Si présent le formulaire n'est pas validé lorsqu'il est soumis

## Placeholder, required, pattern et validation

HTML5 introduit de nombreuses nouveautés pour les formulaires pour améliorer l'aide à la saisie et les contrôles disponibles pour l'utilisateur. Plusieurs attributs simples à mettre en place améliorent la prise en charge des formulaires, tout en se passant de JavaScript.

### Placeholder

**placeholder** est un attribut qui permet de renseigner un texte indicatif par défaut dans un champ de formulaire.

*Modifier la page : 05-formulaire\_simple.php*

Ajouter l'attribut placeholder sur chaque champ de saisie

```
<label for=»nom«>Nom : </label><input type=»text« name=»nom« id=»nom« placeholder=»votre nom« /><br/>
```

```
<label for="prenom">Prénom : </label><input type="text" name="prenom" id="prenom"
placeholder="votre prénom" /><br/>
```

### Enregistrer et visualiser le résultat

L'attribut placeholder peut être placé sur les éléments :

- **<input>** : de type text, search, password, url, tel, email
- **<textarea>**

Compatibilité :

Firefox 4.0+, Opera 11.01+, Chrome 3.0+, Safari 3.0+, ie 10+

Il est bien sûr possible de styler cet élément néanmoins le flou actuel nous oblige à passer par des pseudo class :

```
-webkit-input-placeholder { //safari
-moz-placeholder { //firefox
```

A l'heure actuelle, les styles applicables au placeholder sont très limités, ils se concentrent majoritairement sur des styles de texte :

- Text-decoration
- Font-style
- Color

Dans le fichier CSS :

```
/* firefox */
#mon_formulaire input:-moz-placeholder {
    color: red;
}
/*safari */
#mon_formulaire input::-webkit-input-placeholder {
    color: red;
}
```

### Alternatives

Tous les navigateurs ne se comportent pas de la même manière : compréhension de l'attribut, possibilité de modification des styles via CSS, etc.

- **jQuery Placeholder** : Permet l'attribution du comportement du placeholder ainsi que d'une classe CSS permettant d'uniformiser la méthode de sélection.
- <https://github.com/NV/placeholder.js> : propose une alternative compatible jQuery ou en JavaScript pur.

### Les champs requis

L'attribut required permet de rendre obligatoire le remplissage d'un champ et bloquer la validation du formulaire si l'un des champs (concernés par cet attribut) n'a pas été renseigné.

Modifier la page : [chap\\_4/05-formulaire\\_simple.php](#)

Ajouter l'attribut **required** sur chaque champ de saisie

```
<label for="nom">Nom : </label><input type="text" name="nom" id="nom" placeholder="votre nom" required="required" /><br/>
<label for="prenom">Prénom : </label><input type="text" name="prenom" id="prenom" placeholder="votre prénom" required="required" /><br/>
```

L'attribut placeholder peut être placé sur les éléments :

- **<input>** : de type text, search, password, url, tel, email, date, datetime, datetime-local, months, week, time, number, checkbox, radio, file

- **<textarea>**

#### Compatibilité :

Firefox 4.0+, Opera 9.5+, Chrome 3.0+, ie 10+

Nous allons pouvoir appliquer un CSS sur le champ de saisie afin d'indiquer que celui-ci est obligatoire

```
[required] {
    border: 1px solid orange;
}
```

#### *Enregistrer et visualiser le résultat*

Il est possible aussi de modifier l'affichage du message obligatoire, l'exemple ci-dessous modifiera celui du navigateur Chrome :

```
[type="text"]::-webkit-validation-bubble-arrow {
    background-color: #f4f4f4;
    border: 1px solid #D8000C;
    border-width: 1px 0 0 1px;
    box-shadow: none;
}
[type="text"]::-webkit-validation-bubble-icon {
    display:none;
}
[type="text"]::-webkit-validation-bubble-message {
    background: #f4f4f4;
    border: 1px solid #D8000C;
    color: #D8000C;
    font-size: 100%;
    box-shadow: 0 1px 5px #bbb;
    text-shadow: 0 1px 0 #fff;
}
[type="text"]::-webkit-validation-bubble-message::before {
    content: «Téléphone : »;
```

## Les patterns et la validation à la volée

Les champs requis non complétés ou ne respectant pas une certaine règle de syntaxe définie par le type du formulaire, ou par l'attribut **pattern** bloquent le processus d'envoi des données d'un formulaire.

Cette validation à la volée permet à l'utilisateur d'être informé très rapidement de ses erreurs et de les corriger étape par étape grâce aux indications fournies par les infobulles situées sous chacun des champs invalides.

Certains patterns par défaut existent pour certains types de champ, comme les champs de type **email** ou **url** par exemple. Nous aborderons ces nouveaux types dans le prochain chapitre.

Il est cependant possible de créer ces propres patterns grâce aux **expressions régulières**.

Exemple : `<input type="text" pattern="[A-F][0-9]{5}" />`

Ce champ attend une valeur numérique d'au moins 5 chiffres précédés d'une lettre majuscule comprise entre A et F. Si le format n'est pas respecté, le navigateur en informe l'utilisateur.

Dans notre fichier nous allons ajouter un nouveau champ permettant la saisie d'un code. Celui-ci devra répondre au critère du précédent exemple :

```
<label for="code">Code : </label><input type="text" name="code" id="code" pattern="[A-F][0-9]{5}" />
```

L'attribut pattern peut être placé sur les éléments :

- **<input>** : de type text, search, password, url, tel, email

CSS3 prévoit deux pseudo-classes que sont :valid et :invalid afin de marquer clairement les champs... valides et invalides. Il est alors possible d'écraser les styles par défaut appliqués à ces éléments par les navigateurs, qui diffèrent :

- Firefox attribue la propriété box-shadow (rouge) au champ.
- Internet Explorer utilise la propriété border-color (rouge).
- Chrome et Opera n'ajoutent aucun style, mais placent le focus sur le premier champ erroné

*Modifier le CSS et ajouter*

```
:valid {  
    box-shadow: 0 0 2px 1px green;  
}
```

*Enregistrer et tester sur Firefox*

### Compatibilité :

Firefox 4.0+, Opera 9.5+, Chrome 3.0+, ie 10+

### Les bibliothèques JavaScript comme complément

Pour combler le retard des anciens navigateurs quant à la compatibilité avec les nouveautés introduites par HTML5 pour les formulaires, quelques solutions construites sur JavaScript existent. Il s'agit de charger une bibliothèque qui va détecter si le navigateur ne reconnaît pas les attributs évoqués, ou les nouveaux types de champs, et tenter de les simuler dynamiquement avec JavaScript le cas échéant.

La prise en charge par les navigateurs web est très inégale et variée. Étant donné qu'il s'agit de fonctionnalités disparates, il se peut très bien qu'une version relativement passée prenne en charge un attribut mais pas l'autre et inversement selon le moteur de rendu, voire qu'un type spécifique pour `<input>` soit passé sous silence. Il n'est donc pas possible de déclarer - dans l'état actuel du parc de navigateurs - que l'ensemble des apports des formulaires HTML5 est supporté par une version précise, bien que de nombreux progrès eussent été accomplis.

### Alternatives pour l'émulation des fonctionnalités de formulaires HTML5 avec JavaScript

Ces bibliothèques sont couramment nommées polyfills en version originale, et prennent parfois en compte de bien anciens navigateurs (jusqu'à Internet Explorer 6). Dans les cas d'utilisation les plus simples, il suffira d'inclure le fichier JavaScript dans la page courante grâce à la balise `<script>`.

- H5F : propose un support des nouveautés apportées par HTML5 Forms pour les vieux navigateurs. (<http://www.thecssninja.com/javascript/H5F>)

- H5Validate : propose un support des validations. (<http://ericleads.com/h5validate>)

- Webforms 2 : comprend une émulation de certains attributs (pattern, required, autofocus), des nouveaux types pour `<input>`, et des méthodes JavaScript pour connaître l'état de la validation.

<https://github.com/westonruter/webforms2>

- html5Widgets : embarque une panoplie de contrôles supplémentaires (datepicker/calendrier pour les champs de dates), une gestion des attributs required, pattern, autofocus, placeholder, etc et des éléments range, output. <https://github.com/zoltan-dulac/html5Widgets>

### Les nouveaux types `<input>`

HTML5 apporte plus d'une douzaine de nouveaux types, dont nous allons découvrir ou redécouvrir la liste et le descriptif pour chacun d'eux.

Les tests suivant seront principalement effectués sur Opéra, safari, chrome

#### Le type tel

Il s'agit d'une déclinaison d'un champ de type text. Aussi, aucun format spécifique n'est attendu par le navigateur. Il n'y a donc pas de pattern précis qui vérifierait si au moins un chiffre est renseigné.

Cependant, sur certains navigateurs de SmartPhone, comme sur Safari pour iOS par exemple, l'entrée du numéro de téléphone est facilitée par le basculement à un clavier de type numérique.

Créer la page : [chap\\_4/06-formulaire\\_typeTel.php](#)

Le navigateur ne proposant pas cette vérification, un contrôle plus fin grâce à l'attribut pattern ainsi qu'un contrôle côté serveur sera nécessaire.

Pour effectuer ce type de contrôle, vous aurez besoin d'une expression régulière pour détecter un

numéro de téléphone local ou international.

```
pattern="^(\\+\\d{1,3}(-|)?\\(\\d{1,5}\\)|\\(\\d{2,6}\\)?)(-|)?\\d{3,4}(-|)?\\d{4}(((x| ext)\\d{1,5}){0,1})$"
```

```
<form id="mon_formulaire" action="" method="post">
  <label for="champ1">Tel.</label>
  <input type="tel" id="champ1" required> <em>(sans pattern)</em><br />
  <label for="champ2">Tel.</label>
  <input type="tel" id="champ2" pattern="^(\\+\\d{1,3}(-|)?\\(\\d{1,5}\\)|\\(\\d{2,6}\\)?)(-|)?\\d{3,4}(-|)?\\d{4}(((x| ext)\\d{1,5}){0,1})$" required> <em>(avec pattern)</em><br />
  <input type="submit" value="Valider" />
</form>
```

### Compatibilité :

**Nav.** : Firefox 4.0+, Opera 9.5+, Chrome 3.0+, ie 10+

**Tel.** : Firefox, Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

### Le type url

En apparence ce champ ressemble à celui de type **text**. Cependant, le navigateur attend cette fois un format bien spécifique devant respecter un pattern de type url.

Tous les types d'URL sont admis (ftp://, mailto:, http://, etc.). Par défaut, sur Opera, lorsque vous ne spécifiez pas le type de protocole, le navigateur ajoute automatiquement le type http://, ce qui valide forcément le format attendu pour ce champ.

*Créer la page : [chap\\_4/07-formulaire\\_typeURL.php](#)*

```
<form id="mon_formulaire" action="" method="post">
  <label for="champ1">Votre site web</label>
  <input type="url" id="champ1" required value="http://">
  <br/>
  <input type="submit" value="Valider" />
</form>
```

### Compatibilité :

**Nav.** : Firefox 4.0+, Opera 9.5+, Chrome 3.0+, ie 10+

**Tel.** : Firefox, Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

### Le type email

Toujours très proche du type text, ce champ est équivalent au type url, seul le format attendu change.

Ce champ attend au minimum un caractère (caractère non accentué comprenant les séparateurs tirets ou underscore) suivi d'un @ suivi à son tour d'un caractère.

*Exemple d'entrée invalide* : é@c

*Exemples d'entrées valides* : -@\_ , f@r

Une nouvelle fois, le clavier du SmartPhone compatible avec ce type de champ de formulaire vous présentera un clavier adapté incluant le symbole arobase.

Créer la page : [chap\\_4/08-formulaire\\_typeEmail.php](#)

```
<form id="mon_formulaire" action=" method="post">
  <label for="champ1">Votre e-mail</label>
  <input type="email" id="champ1" required>
  <br />
  <input type="submit" value="Valider" />
</form>
```

### Compatibilité :

**Nav.** : Firefox 4.0+, Opera 9.0+, Chrome 6.0+, ie 10+, Safari 5+

**Tel.** : Firefox, Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

### Le type date,time et datetime

#### Champ de type date

Ce champ visuellement proche de celui de type text vous permet d'activer une aide au remplissage (type *datepicker*) présente uniquement sur quelques navigateurs et différente de l'un à l'autre.

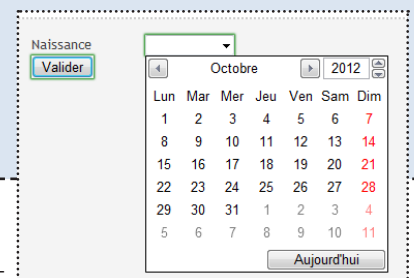
Le contenu attendu est une date du calendrier Grégorien au **format RFC3339** sans précision de la *timezone*, mais le champ accepte un contenu vide ou mal formaté sans retourner d'erreur.

La liste des attributs compatible est la suivante :

name, disabled, form, type, autocomplete, autofocus, list, min, max, step (chiffre entier), readonly, required, value, pattern ainsi que les attributs classiques, d'événements et xml.

Créer la page : [chap\\_4/09-formulaire\\_typeDate.php](#)

```
<form id="mon_formulaire" action="09-formulaire_typeDate.php" method="post">
  <label for="champ1">Votre date</label>
  <input type="date" id="champ1" required>
  <br />
  <input type="submit" value="Valider" />
</form>
```



### Compatibilité :

**Nav.** : Firefox 4.0+, Opera 9.0+, Chrome 6.0+, ie 10+, Safari 5+

**Tel.** : Firefox, Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

#### Champ de type time

Ce type de champ permet de renseigner une heure, avec plus ou moins de précision. Le format attendu est le même que pour le champ de type date, sans localisation (*timezone*).

La liste des attributs compatibles est très proche, seul l'attribut step autorise ici un nombre à virgule (*float*).

À l'instar du champ de type date, ce type de champ déclenche l'affichage d'une sorte de sélecteur

permettant d'incrémenter ou décrétement la valeur du champ minute par minute.

On peut rencontrer ce comportement sur Safari et Opera sous la forme suivante.

Créer la page : [chap\\_4/10-formulaire\\_typeTime.php](#)

```
<form id="mon_formulaire" action="10-formulaire_typeTime.php" method="post">
  <label for="champ1">Heure de r&eacute;veil</label>
  <input type="time" id="champ1" name="time">
  <br />
  <input type="submit" value="Valider" />
</form>
```

### Compatibilité :

**Nav.** : Firefox 4.0+, Opera 10.6+, Chrome 16, ie 10+,Safari 5+

**Tel.** : Firefox,Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

### champ de type datetime

Ce type de champ est une combinaison des deux types **date** et **time**. Il permet donc de renseigner une date et une heure précise dans un même champ (visuel), dans le cas de Safari, et dans deux champs séparés dans le cas d'Opera.

datetime propose un timezone qui est précisé en dehors des champs sous Opera, et directement intégré au champ dans Safari ("Z" équivaut ici à "UTC").

Créer la page : [chap\\_4/11-formulaire\\_typeDateTime.php](#)

```
<form id="mon_formulaire" action="11-formulaire_typeDateTime.php" method="post">
  <label for="champ1">Prochain RDV</label>
  <input type="datetime" name="datetime" id="champ1"><br />
  <input type="submit" value="Valider" />
</form>
```

### Compatibilité :

**Nav.** : Firefox 4.0+, Opera 10.6+, Chrome 16, ie 10+,Safari 5+

**Tel.** : Firefox,Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

### Champ de type datetime-local

Ce type de champ est très proche du type **datetime**, la seule différence étant l'absence de précision du fuseau horaire.

Les informations visuelles fournies par les champs de formulaire sont semblables à ceux de datetime, "UTC" et "Z" en moins pour Opera et Safari, respectivement.

Créer la page : [chap\\_4/12-formulaire\\_typeDateTime-Local.php](#)

```
<form id="mon_formulaire" action="12-formulaire_typeDateTime-Local.php" method="post">
  <label for="champ1">Prochain RDV</label>
  <input type="datetime" name="datetime" id="champ1"><br />
  <input type="submit" value="Valider" />
```

Valeur renvoyée par le champ :

Prochain RDV 2012-10-24 00:08 UTC

Valider



&lt;/form&gt;

**Compatibilité :****Nav.** : Firefox 4.0+, Opera 10.6+, Chrome 16, ie 10+,Safari 5+**Tel.** : Firefox,Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1**Champ de type week**

Le type de champ **week** permet de renseigner une semaine dans une année. Il attend donc un format de type 2012W05 qui correspond à l'année et au numéro de semaine dans celle-ci.

La liste des attributs compatibles reste la même que pour le type date.

Créer la page : [chap\\_4/13-formulaire\\_typeWeek.php](#)

```
<form id="mon_formulaire" action="13-formulaire_typeWeek.php" method="post">
  <label for="champ1">Semaine </label>
  <input type="week" id="champ1" name="week">
  <br />
  <input type="submit" value="Valider" />
</form>
```

Valeur renvoyée par le champ :

Semaine

Valider

Semaine	Lun	Mar	Mer	Jeu	Ven	Sam	Dim
40	1	2	3	4	5	6	7
41	8	9	10	11	12	13	14
42	15	16	17	18	19	20	21
43	22	23	24	25	26	27	28
44	29	30	31	1	2	3	4
45	5	6	7	8	9	10	11

Aujourd'hui

**Compatibilité :****Nav.** : Firefox 4.0+, Opera 10.6+, Chrome 16, ie 10+,Safari 5+**Tel.** : Firefox,Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1**Champ de type number**

Le type de champ **number** permet de renseigner une valeur numérique. Le champ de formulaire est alors transformé en une sorte de boîte permettant l'incrémentement et la décrémentement d'une valeur numérique initiale (0 par défaut), lorsque la prise en charge par le navigateur est complète.

Créer la page : [chap\\_4/14-formulaire\\_typeNumber.php](#)

```
<form id="mon_formulaire" action="14-formulaire_typeNumber.php" method="post">
  <label for="champ1">Code postal </label>
  <input type="Number" id="champ1" name="cp">
  <br />
  <input type="submit" value="Valider" />
</form>
```

Valeur renvoyée par le champ :

nombre

Valider

Il est possible d'ajouter un attribut step pour spécifier un pas d'incrémentement. Mais aussi de fixer un minimum et un maximum.

Créer la page : [chap\\_4/14-formulaire\\_typeNumber\\_Bis.php](#)

```
<form id="mon_formulaire" action="14-formulaire_typeNumber_Bis.php" method="post">
  <label for="champ1">Nb bits </label>
  <input type="number" id="champ1" name="number" step="8" min="0" max="64">
  <br />
  <input type="submit" value="Valider" />
</form>
```

**Compatibilité :**

**Nav.** : Firefox 4.0+, Opera 10.6+, Chrome 16, ie 10+,Safari 5+

**Tel.** : Firefox,Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

**Champ de type range**

Ce champ propose un contenu évalué approximatif.

Bien que l'on puisse convertir la position du curseur numériquement (par défaut la valeur la plus basse est 0, la plus haute 100), l'utilisateur n'a pas de repère numérique, seule la position du curseur est un indice. La valeur la plus haute se trouve à gauche (il fallait le deviner ?) pour un sens de lecture *ltr* (left to right) bien entendu !

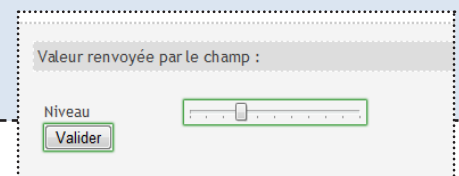
Si vous avez l'habitude de passer d'une lecture *rtl* à *ltr* de par votre polyglottisme, cet affichage approximatif pourrait vous poser des soucis (cela reste une supposition), d'autant plus que le curseur se place par défaut au milieu.

Créer la page : [chap\\_4/15-formulaire\\_typeRange.php](#)

```
<form id="mon_formulaire" action="15-formulaire_typeRange.php" method="post">
  <label for="champ1">Niveau </label>
  <input type="range" id="champ1" name="range"><br />
  <input type="submit" value="Valider" />
</form>
```

Possibilité d'une valeur de départ, un max un min et un pas d'incrément

```
<form id="mon_formulaire" action="15-formulaire_typeRange.php" method="post">
  <label for="champ1">Niveau </label>
  <input type="range" id="champ1" name="range" value="15" max="50" min="0"
step="5"><br />
  <input type="submit" value="Valider" />
</form>
```

**Compatibilité :**

**Nav.** : Firefox 4.0+, Opera 11+, Chrome 10, ie 10+,Safari 5+

**Tel.** : Firefox,Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

**Champ de type color**

Ce type permet de transformer le champ en une palette de couleurs.

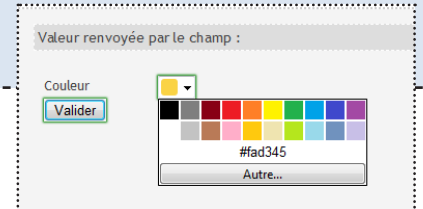
La valeur attendue est une couleur au format hexadécimal (un dièse suivi de 6 caractères alpha-numériques compris entre A et F, et 0 et 9).

Actuellement seul Opera 11+ permet l'affichage de cette palette, avec un nombre de couleurs limité, proposant, par l'intermédiaire d'un bouton "Autre...", d'étendre la palette et un colorpicker vous permettant de composer une palette de couleurs personnalisées temporaire.

Les autres navigateurs affichent un simple champ de type texte, il n'est pas possible de prévoir un *pattern* contrôlant la valeur renseignée par l'utilisateur, puisque cet attribut n'est pas compatible avec ce type.

Créer la page : [chap\\_4/16-formulaire\\_typeColor.php](#)

```
<form id="mon_formulaire" action="16-formulaire_typeColor.php" method="post">
  <label for="champ1">Couleur </label>
  <input type="color" id="champ1" name="color" value="#fad345"> <input type="submit"
value="Valider" />
</form>
```



### Compatibilité :

**Nav.** : Firefox 4.0+, Opera 11+, Chrome 10, ie 10+, Safari 5+

**Tel.** : Firefox, Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

### L'élément output

Ce nouvel élément représente la somme d'un calcul.

Il accepte comme attribut : for, name, et form.

La valeur de l'attribut for de l'élément <output> reprend les valeurs des id de chaque champ <input> nécessaires au calcul.

L'attribut form de l'<output> reprend la valeur de l'id du formulaire qui comprend les différents champs servant au calcul.

Ce dernier point est important car l'élément output n'est pas forcément interne au formulaire.

Créer la page : [chap\\_4/17-formulaire\\_typeOutput.php](#)

```

<form action="17-formulaire_typeOutput.php" method="post" id="tva_form" onsubmit="ttc.value
= ht.value * (1 + tva.value/100); return false;">
  <p>
    <label for="t_ht">Tarif HT</label><input type="number" name="ht" id="t_ht"> &euro;
  </p>
  <p>
    <label for="t_tva">TVA</label><input type="number" name="tva" id="t_tva" va-
    lue="19.6"> %
  </p>
  <p>
    Prix TTC :
    <output for="t_ht t_tva" name="ttc" form="tva_form"></output> &euro;
  </p>
  <button>valider</button>
</form>

```

### Compatibilité :

**Nav.** : Firefox 4.0+, Opera 11+, Chrome 10, ie 10+, Safari 5+

**Tel.** : Firefox, Chrome, Opera (Android 4), Safari (iOS 3.1), Android Browser 3.1

### L'élément Keygen

Cet élément nécessite un certain bagage technique pour en comprendre et maîtriser les usages.

Essayons d'en résumer les grandes lignes.

**<keygen>** permet de générer un jeu de clefs pour le cryptage et le décryptage d'informations (enregistrées en base de données ou transmises d'un serveur à l'autre par exemple).

Le jeu équivaut à une paire de clefs, l'une dite publique, l'autre dite privée. La clef privée est stockée localement, tandis que la clef publique est envoyée sur le serveur. Différents niveaux de cryptage existent et correspondent à un niveau de sécurité plus ou moins élevé.

### L'attribut keytype

Lié à l'élément keygen cet attribut permet de renseigner le type de clef attendue. Pour le moment la seule existante est celle de type RSA (Rivest, Shamir et Adleman, les inventeurs de la méthode de cryptage).

**<keygen keytype="rsa" name="key">**

### L'attribut challenge

Cet attribut ajoute une chaîne de caractère envoyée avec la clef publique.

**<keygen keytype="rsa" challenge="sel\_de\_mer" name="key">**

Créer la page : [chap\\_4/18-formulaire\\_typeKeygen.php](chap_4/18-formulaire_typeKeygen.php)

```

<p class=»hint»>Génération d'une clef. Le niveau peut-être augmenté grâce à l'attribut
<code>challenge</code>.</p>
<form action=»element-keygen.php» method=»post»>
  <p>
    <label for=»e-mail»>E-mail personnel</label>
    <input type=»email» name=»email» id=»e-mail»>
  </p>
  <p>
    <label for=»secu»>Sécurisation</label>
    <keygen type=»secu» name=»secu» id=»secu»>
  </p>
  <br /><br />
  <button>Tayst</button>
</form>

<script src=»https://ajax.googleapis.com/ajax/libs/jquery/1.6.1/jquery.min.js»></script>

<script type=»text/javascript»>
  $(function()
  {
    $(«form p+p»).after('<p><label for=»plusplus»>Ajouter un «challenge» ?</label><input
type=»checkbox» name=»plusplus» id=»plusplus» /></p>');
    $(«#plusplus»).live('change', function()
    {
      if($(this).filter(':checked').length==1) {
        var date = new Date();
        $(«#secu»).attr('challenge', date.getTime());
        $(«#secu»).after('<span class=»moreplus»> ++</span>');
      }else {
        $(«#secu»).removeAttr('challenge');
        $(«.moreplus»).remove();
      }
    });
  });
</script>

```

# CSS 3

Préfixe Navigateur

Les bordures

Les ombrages

Transparance & opacité

Les arrières-plans multiples

Les dégradés

Les transformations

Les transitions

Les animations

## Préfixe de navigateur

Nous sommes aujourd'hui dans un entre-deux; toutes les nouveautés CSS3 ont été définies par le W3C cependant celles-ci n'ont pas encore été implémentées directement dans les navigateurs. Cependant certains navigateurs avaient développé leurs propres règles CSS3; celles-ci ont d'ailleurs servi aux recommandations du W3C.

Pour cette raison il sera nécessaire de donner une définition spécifique CSS pour chaque navigateur et que la page HTML devra toujours être testée sur le plus grand nombre de navigateur.



Safari

-webkit-



Microsoft

-ms-



Mozilla

-moz-



Chrome

-chrome-



## Les bordures

Dans la version CSS 2.1 il est possible de définir des styles de bordures mais ceci ne répondait pas à toutes les attentes des designers. Très souvent ceux-ci avaient à faire des bords arrondie ou bien des blocs avec des bordures très spécifiques et la seule solution était de placer une image en fond. Dans ce cas le contenu devait s'adapter au contenant (problème si celui-ci est changeant).

Avec CSS3 il possible de définir ces types de bordure.

Créer le document : [chap\\_5/01-bordure.html](#) à partir de la la page [modele\\_page.html](#)

Inclure le code suivant après la balise header:

```
<aside>
  <div id="introduction-content">
    <h2>Le CSS3 du jour</h2>
    <p>Lorem .. bibendum.</p>
  </div>
</aside>
```

Création de la feuille de style associé :  
[assets/css/styles\\_bordure.css](#)

```
#introduction-content{
  width:400px;
  background:#F9A890;
  margin:20px 0 0 20px;
  border: 2px #f2662a solid;
  padding:10px;
}
```

### Le CSS3 du jour

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque vehicula, enim non mattis lobortis, libero est hendrerit erat, a tincidunt lacus sapien sit amet mi. Nulla fringilla lobortis porttitor. Mauris rutrum neque ac leo vehicula vel fringilla nisi consectetur. Suspendisse potenti. In libero metus, scelerisque et sodales et, convallis nec elit. Aenean est lectus, adipiscing nec feugiat sed, consequat ut nunc. Ut id felis vel purus venenatis bibendum.

## Border-radius

Ajoutons des bord arrondies dans le CSS :

```
#introduction-content{
    ...
    border-radius:24px;
    ...
}
```

Enregistrer et visualiser le résultat

### Le CSS3 du jour

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque vehicula, enim non mattis lobortis, libero est hendrerit erat, a tincidunt lacus sapien sit amet mi. Nulla fringilla lobortis porttitor. Mauris rutrum neque ac leo vehicula vel fringilla nisi consectetur. Suspendisse potenti. In libero metus, scelerisque et sodales et, convallis nec elit. Aenean est lectus, adipiscing nec feugiat sed, consequat ut nunc. Ut id felis vel purus venenatis bibendum.

Cette propriété est utilisée sur la plupart des nouveaux navigateurs. Cependant sous d'anciennes versions cette propriété n'est pas appliquée. Sur les anciens navigateurs nous devons ajouter 2 propriétés supplémentaires adressant ce que l'on appelle généralement des préfixes vendeurs.

```
#introduction-content{
    ...
    -webkit-border-radius:24px;
    -moz-border-radius:24px;
    -o-border-radius:24px;
    -ms-border-radius:24px;
    ...
}
```

Dans le cas précédant le bord arrondi a été ajouté à tous les coins mais il est possible de s'adresser à des bords spécifiques :

Dans le fichier CSS modifié le border-radius :

```
#introduction-content{
    ...
    border-radius: 24px 48px 24px 48px
    ...
}
```

Enregistrer et visualiser le résultat

### Le CSS3 du jour

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque vehicula, enim non mattis lobortis, libero est hendrerit erat, a tincidunt lacus sapien sit amet mi. Nulla fringilla lobortis porttitor. Mauris rutrum neque ac leo vehicula vel fringilla nisi consectetur. Suspendisse potenti. In libero metus, scelerisque et sodales et, convallis nec elit. Aenean est lectus, adipiscing nec feugiat sed, consequat ut nunc. Ut id felis vel purus venenatis bibendum.

Nous avons jusqu'ici créé des bords bien arrondis , mais nous pouvons modifier le rayon x et y

Dans le CSS

```
#introduction-content{
    ...
    border-radius: 48px/24px; /* x / y */
    -webkit-border-radius:48px/24px;
    -moz-border-radius:48px/24px;
    -o-border-radius:48px/24px;
    -ms-border-radius:48px/24px;
    ...
}
```

### Le CSS3 du jour

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque vehicula, enim non mattis lobortis, libero est hendrerit erat, a tincidunt lacus sapien sit amet mi. Nulla fringilla lobortis porttitor. Mauris rutrum neque ac leo vehicula vel fringilla nisi consectetur. Suspendisse potenti. In libero metus, scelerisque et sodales et, convallis nec elit. Aenean est lectus, adipiscing nec feugiat sed, consequat ut nunc. Ut id felis vel purus venenatis bibendum.



## Border-image

Afin de pouvoir utiliser la propriété Border-image nous mettrons en commentaire les border-radius et nous allons grossir la taille de la bordure.

Dans le fichier CSS retiré le border-radius & ajout du border-image:

```
#introduction-content{
  border: 20px #f2662a solid;
  border-image: url(..images/border-bg.png) 33%;
  /* cas particulier navigateur */
  -webkit-border-image: url(..images/border-bg.png) 33%;
  -moz-border-image: url(..images/border-bg.png) 33%;
  -o-border-image: url(..images/border-bg.png) 33%;
  -ms-border-image: url(..images/border-bg.png) 33%;
}
```

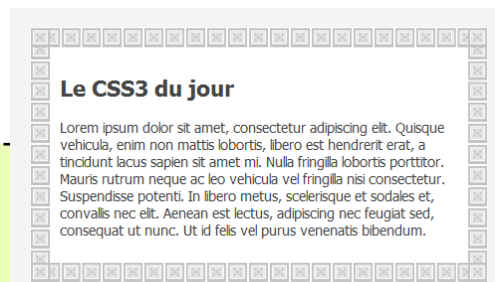
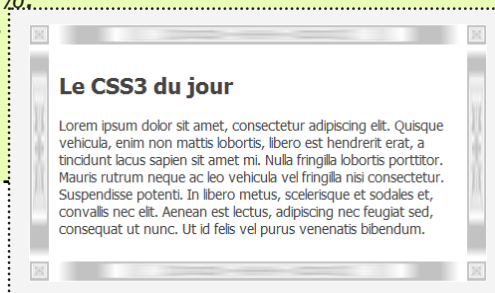


La valeur 33% indique la position où l'image devra être coupée relativement à sa hauteur et à sa largeur.

Enregistrer et visualiser.

On peut remarquer que les bords centraux sont étirés. Afin que ce motif se répète ajouter dans le css :

```
#introduction-content{
  ....
  border-image: url(..images/border-bg.png) 33% repeat;
  /* cas particulier navigateur */
  -webkit-border-image: url(..images/border-bg.png) 33% repeat;
  -moz-border-image: url(..images/border-bg.png) 33% repeat;
  -o-border-image: url(..images/border-bg.png) 33% repeat;
  -ms-border-image: url(..images/border-bg.png) 33% repeat;
  ...
}
```



## Les ombrages

Nouvelle propriété CSS3: l'ombrage sur une boîte et sur du texte. Pour cela rien de plus simple.

Créer le document : [chap\\_5/02-ombrage.html](#) à partir de la page [01-ombrage.html](#)

Puis modifier le ccs pour obtenir [assets/css/styles\\_ombrage.css](#) :

```
#introduction-content{
  width:400px;
  background:#F9A890;
  margin:20px 0 0 20px;
  border:2px #f2662A solid;
```

```
padding:10px;
```

```
box-shadow:15px 10px;
-webkit-box-shadow:15px 10px;
-moz-box-shadow:15px 10px;
-o-box-shadow:15px 10px;
-ms-box-shadow:15px 10px;
}
```

### Le CSS3 du jour

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque vehicula, enim non mattis lobortis, libero est hendrerit erat, a tincidunt lacus sapien sit amet mi. Nulla fringilla lobortis porttitor. Mauris rutrum neque ac leo vehicula vel fringilla nisi consectetur. Suspendisse potenti. In libero metus, scelerisque et sodales et, convallis nec elit. Aenean est lectus, adipiscing nec feugiat sed, consequat ut nunc. Ut id felis vel purus venenatis bibendum.

Les 2 paramètres sont obligatoires ils indiquent le décalage vers la droite puis le décalage vers le bas. Il est possible d'ajouter un troisième paramètre afin de changer le flou (x et y) du dégradé de l'ombrage.

```
box-shadow : 15px 10px 20px;
```

Il existe enfin un 4ème paramètre pour définir la couleur de l'ombre.

```
box-shadow : 15px 10px 20px #ccc;
```

Afin étendre cette propriété au plus grand nombre de navigateur nous devons nous appuyer sur les moteurs Webkit et Moz.

Pour s'adapter à internet explorer ie8

```
<!--[if lte IE 8]>
  <style type="text/css">
    #introduction-content
    {
      zoom: 1;
      filter: progid:DXImageTransform.Microsoft.Shadow(color='#aaaaaa', Direction=135, Strength=6);
    }
  </style>
<![endif]-->
```

Pour ajouter une ombre sur un texte utiliser la propriété text-shadow.

*Modification de la feuille de style associé : [assets/css/base.css](#) pour ajouter une ombre au titre.*

```
header h1{
  ...
  text-shadow: 1px 1px 2px #999;
  -webkit-text-shadow: 1px 1px 2px #999;
  -moz-text-shadow: 1px 1px 2px #999;
  -o-text-shadow: 1px 1px 2px #999;
  -ms-text-shadow: 1px 1px 2px #999;
}
```

# HTML5

## Transparence et opacité

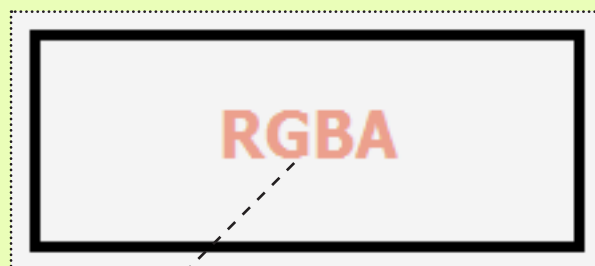
Il existe plusieurs possibilités de créer une opacité. La première est la définition d'une couleur via **rgba**.

Dans un nouveau document [chap\\_5/03-transparence.html](#) ( page [modele\\_page.html](#) )

```
<div class="rgba">
  <h1>RGBA</h1>
</div>
```

```
.rgba {
  width: 200px;
  height: 75px;
  border: #000 4px solid;
  margin: 20px;
}

.rgba h1 {
  text-align: center;
  font-weight: bold;
  font-size: 24px;
}
```



Nous ajoutons au titre **h1**

```
color: rgba(228,76,39,0.5);
```

*Enregistrer et visualiser le résultat.*

Nous allons à présent utiliser la propriété **opacity**. Dans la page HTML ajouter le code suivant :

```
<div class="opacity">
  <h1>OPACITY</h1>
</div>
```

Puis dans la feuille de style

```
.opacity {
  width: 200px;
  height: 75px;
  border: #000 4px solid;
  margin: 20px;
  background-color: #FFF;
  box-shadow: 8px 8px 8px;
  -webkit-box-shadow: 8px 8px 8px;
  -moz-box-shadow: 8px 8px 8px;
  -o-box-shadow: 8px 8px 8px;
  -ms-box-shadow: 8px 8px 8px;
}
```

```
.opacity h1 {
  text-align:center;
  font-weight:bold;
  font-size:24px;
  color:rgb(228,76,39);
}
```



Ajoutons au **div opacity** la propriété suivante :

```
opacity: 0.5; - - - - -
```

*Enregistrer et visualiser le résultat.*

Nous allons à présent utiliser les propriétés **rgba** et **opacité** conjointement avec la propriété drop-shadow.

*Dans le corps de la page HTML:*

```
<div class=»dropshadow»>
  <h1> DROPSHADOW </h1>
</div>
```

*Puis dans la feuille de style*

```
dropshadow {
  width: 200px;
  height: 75px;
  border: #000 4px solid;
  margin:20px;
  background-color:#FFF;
  box-shadow: 8px 8px 8px rgba(0,0,0,0.5); - - - - -
  -webkit-box-shadow: 8px 8px 8px rgba(0,0,0,0.5);
  -moz-box-shadow: 8px 8px 8px rgba(0,0,0,0.5);
  -o-box-shadow: 8px 8px 8px rgba(0,0,0,0.5);
  -ms-box-shadow: 8px 8px 8px rgba(0,0,0,0.5);
}
```

```
.dropshadow h1 {
  text-align:center;
  font-weight:bold;
  font-size:24px;
  color:rgb(228,76,39);
}
```



*Enregistrer et visualiser le résultat.*

## Les arrières plans multiples

Background-image est un outil essentiel en CSS2.1, mais celui-ci était limité à une seule image. CSS3 étend cette capacité à plusieurs images.

Nouveau document : [chap\\_5/04-bg\\_multiple.html](#) & nouvelle feuille de style : [assets/css/styles\\_bg\\_multiples.css](#).

On ajoute un fond dégradé via une image :

```
body{
  background-image: url(../images/bg1.png); - - - - -
```

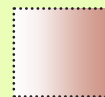


Donnant le résultat suivant :



Sans options celle-ci se répète de manière verticale et horizontale. Maintenant ajoutons une 2ème image au background de la manière suivante :

```
body{
  background-image: url(../images/bg1.png),url(../images/bg2.png) ; - - - -
```



On peut remarquer que les 2 fonds se chevauchent indépendamment et qu'ils ont leur propre vie. Ajoutons une 3ème image :

```
body{
  background-image: url(../images/bg1.png),url(../images/bg2.png), url(../images/bg3.png) ;
```



Et ainsi de suite...

Il est important de noter que si les images se trouvent toutes dans le background celles-ci ne sont pas toutes sur le même plan. La 1ère image mentionnée est celle qui est la plus éloignée et celle indiqué en dernier est celle qui se trouve la plus proche de nous.

Il est possible aussi de définir la position d'une des images du background. Dans l'exemple qui suit nous placerons un dégradé de chaque côté:

```
body{
  background-image: url(../images/bg4.png),url(../images/bg5.png);
  background-repeat: repeat-y;
  background-position: top left, top right;
```



## Les dégradés

Les propriétés précédentes n'offrent que la possibilité de créer une couleur de fond unie. Si l'on voulait intégrer un dégradé au sein de notre site web il fallait recourir au background-image auquel on associait une image élaboré dans un logiciel tel que photoshop. Avec CSS3 et la propriété background nous allons pouvoir créer nos dégradés directement dans nos pages en évitant le chargement d'une image par le réseau.

Nous pouvons obtenir deux types de dégradés: linéaire, horizontal ou vertical, centré ou non, avec deux ou plusieurs couleurs.

*Nouveau document : [chap\\_5/05-degrade.html](#) & nouvelle feuille de style : [assets/css/styles\\_degrade.css](#).*

Création des conteneurs de dégradé dans la page HTML:

```
<div id=»simple»>
  Dégradé simple
</div>
<div id=»angulaire»>
  Dégradé angulaire
</div>
<div id=»circulaire»>
  Dégradé circulaire
</div>
<div id=»repetitif»>
  Dégradé répétitif
</div>
<div id=»rgba»>
  <h1>Dégradé RGBA</h1>
</div>
```

```
div{
  height:75px;
  width:400px;
  margin:20px;
  text-align:center;
  border: 2px #f2662a solid;
  font-size:20px;
  font-weight:bold;
}
```

**Dégradé simple**

**Dégradé angulaire**

**Dégradé circulaire**

**Dégradé répétitif**

**Dégradé RGBA**

### Dégradé simple

Pour obtenir des dégradés linéaires, la syntaxe à utiliser est la suivante:

**background : linear-gradient(top|left, couleur 1 X%, couleur 2 Y%, ... couleur N z%);**

ou bien sur la propriété *background-image*. Dans le fichier CSS

```
#simple{
  background-image:linear-gradient(top, white,black);
  background-image:-moz-linear-gradient(top,white,black);
  background-image:-o-linear-gradient(top,white,black);
  background-image:-ms-linear-gradient(top,white,black);
  background-image:-webkit-linear-gradient(top,white,black);
}
```

Dégradé simple

Nous pouvons remarquer que le seul navigateur sur lequel ne s'applique pas le dégradé est IE. Cependant n'oubliez pas d'indiquer quand même sa valeur dans le cas où cette propriété finirait par être implémentée.

Si nous avons voulu que le dégradé linéaire soit appliqué de façon horizontale, il suffirait de remplacer la valeur **top** par **left**.

### Dégradé angulaire

Le dégradé ne se limite pas à une application vertical ou horizontal d'une couleur vers une autre.

Dégradé angulaire

```
#angulaire{
  background-image:linear-gradient(45deg, white, green, black);
  background-image: -moz-linear-gradient(45deg, white, green, black);
  background-image: -o-linear-gradient(45deg, white, green, black);
  background-image: -ms-linear-gradient(45deg, white, green, black);
  background-image: -webkit-linear-gradient(45deg, white, green, black);
}
```

### Dégradé radial

Pour créer un fond en dégradé radial nous disposons encore de propriétés *background* ou *background-image* :

**background : radial-gradient(départ, forme, coul1 X%, coul2 Y%);**

Valeur pour départ:

*top, middle, bottom left, center, right* - exemple: *top right* (départ du cercle en haut à droite)

Il est aussi possible d'indiquer directement une position précise en indiquant un pourcentage 60% 40% (à 60% de hauteur et 40% de largeur).

Valeur pour la forme:

Il est d'abord composé des mots-clés *circle* ou *ellipse* pour la forme elle-même et des mots clés suivant pour son extension:

*cover* : occupe toute la surface du conteneur.

*closest-side* : associé à *circle*, le dégradé est circulaire et s'arrête sur le côté le plus proche de son centre. Avec *ellipse* il s'étend sur toute la surface.

*closest-corner* : le dégradé s'étend jusqu'au coin le plus proche de son conteneur.



*farthest-side* : le dégradé s'étend jusqu'au côté le plus éloigné de son conteneur.

*farthest-corner* : le dégradé s'étend jusqu'au coin le plus éloigné de son conteneur.

*contain* : la forme circulaire ou elliptique est entièrement contenue dans le conteneur.

Dans le fichier CSS définir le dégradé radial suivant:

Dégradé circulaire

```
#circulaire{
  background-image: radial-gradient(center top, circle cover, white, #111);
  background-image: -moz-radial-gradient(center top, circle cover, white, #111);
  background-image: -o-radial-gradient(center top, circle cover, white, #111);
  background-image: -ms-radial-gradient(center top, circle cover, white, #111);
  background-image: -webkit-radial-gradient(center top, circle cover, white, #111);
}
```

### Dégradé répétitifs

Pour répéter un effet de dégradé il suffit de définir la valeur suivante à la propriété background ( ou background-image) : X% définissant la zone restante

**background : repeating-linear-gradient(top|left, couleur 1 X%, couleur 2 Y%, ... );**

**background : repeating-radial-gradient(départ, forme, coul1 X%, coul2 Y%);**

```
#repetitif{
  background-image: repeating-linear-gradient(left, white 80%, black, white);
  background-image: -moz-repeating-linear-gradient(left, white 80%, black, white);
  background-image: -o-repeating-linear-gradient(left, white 80%, black, white);
  background-image: -ms-repeating-linear-gradient(left, white 80%, black, white);
  background-image: -webkit-repeating-linear-gradient(left, white 80%, black, white);
}
```

Dégradé répétitif

### Dégradé RGBA

Nous pouvons ajouter des couleurs RGBA afin d'obtenir un effet de transparence dans les dégradés, ce qui leurs donnent une certaine profondeur.

Dans l'exemple qui suit le dégradé transparent est appliqué au style <h1>, ce qui permet à l'image d'arrière plan du conteneur d'être visible:

```
#rgba h1{
  height:75px;
  margin:0;

  background-image:linear-gradient(top, rgba(147,185,196,0.9),rgba(110,124,140,0.9));
  background-image:-webkit-linear-gradient(top,rgba(147,185,196,0.9),rgba(110,124,140,0.9));
  background-image:-moz-linear-gradient(top,rgba(147,185,196,0.9),rgba(110,124,140,0.9));
  background-image:-ms-linear-gradient(top,rgba(147,185,196,0.9),rgba(110,124,140,0.9));
  background-image:-o-linear-gradient(top,rgba(147,185,196,0.9),rgba(110,124,140,0.9));
}
```

Dégradé RGBA



## Les transformations

Les transformations CSS vont nous permettre de faire tourner, de changer l'échelle ou encore de tordre un élément de notre page. De même il est possible de définir une perspective afin de simuler une position ou une animation dans l'espace 3D.

### Transformation 2D

Nouveau document : [chap\\_5/06-transformation.html](#) & nouvelle feuille de style : [assets/css/transformation.css](#).

```
<div id="wrap">
  <div class="transformation">
    
  </div>
</div>
```

```
#wrap{
  width:80px;
  margin-right:10px;
}
#transformation{
  float:right;
  margin:10px;
  border: 20px solid #fff;
  border-radius: 10px;
  transform: rotate(5deg);
  -webkit-transform: rotate(5deg);
  -moz-transform: rotate(5deg);
  -o-transform: rotate(5deg);
  -ms-transform: rotate(5deg);
}
```



Si l'on avait voulu modifier les proportions du div on aurait appliqué :

transform: scale(1.5) -> 1.5 représente la valeur 150%, à taille normal un objet vaut 1 (100%)

### Transformation 3D

#### Les rotations

rotate3d(x,y,z,angle) : spécifie une rotation 3D autour de l'axe défini par le vecteur (x,y,z).

rotateX(angle) : spécifie une rotation autour de l'axe X.

rotateY(angle) : spécifie une rotation autour de l'axe Y.

rotateZ(angle) : spécifie une rotation autour de l'axe Z. Équivalent à rotate(angle) en 2D.

#### Les translations

translate3d(x,y,z) : spécifie une translation en 3D.

translateZ(z) : spécifie une translation sur l'axe Z.

#### Les changements d'échelle

scale3d(x,y,z) : spécifie un changement d'échelle en 3D.

scaleZ(z) : spécifie un changement d'échelle sur l'axe Z.

#### Matrice

matrix3d(une matrice 4x4) : spécifie une matrice de transformation. Par défaut, toutes les fonctions de transformations sont converties en utilisant une matrice, mais cela est transparent pour nous.

#### Perspective

perspective(nombre) : spécifie la profondeur de perspective.

Il est possible d'appliquer plusieurs transformations différentes sur un même élément HTML. Pour cela, séparez les différentes fonctions par un espace. Attention toutefois, les fonctions sont appliquées dans l'ordre d'écriture, et donc l'effet final peut varier (par exemple, une rotationX de 45deg + une rotationY de 45deg n'est pas équivalente à une rotationY de 45deg + une rotationX de 45deg).

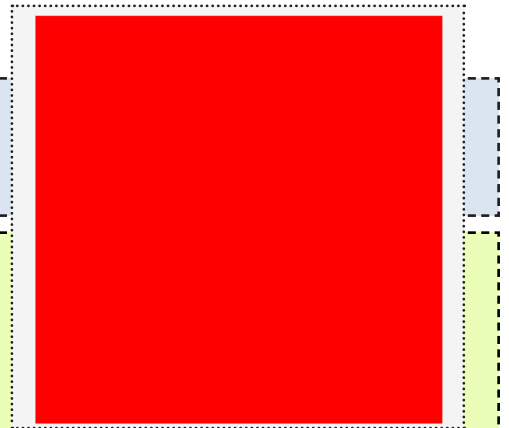
*Nouveau document : [chap\\_5/06-transformation-bis.html](#) & nouvelle feuille de style : [assets/css/transformation.css](#).*

Nous créons une boîte rouge contenue dans un autre div

```
<div id="scene3D">
  <div id="red"></div>
</div>
```

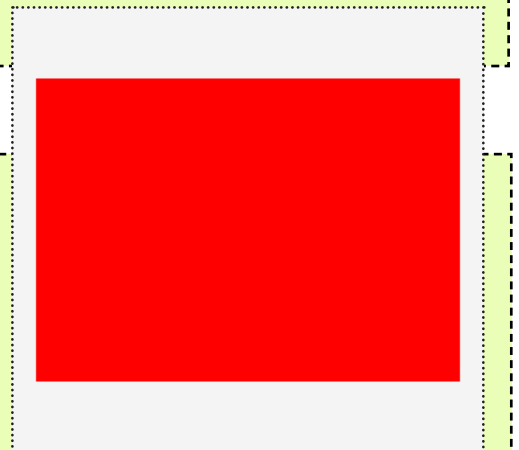
```
#red{
  width:300px;
  height:300px;
  background:red;
}
```

```
#scene3D{
  margin:auto;
  width:300px;
  height:300px;
}
```



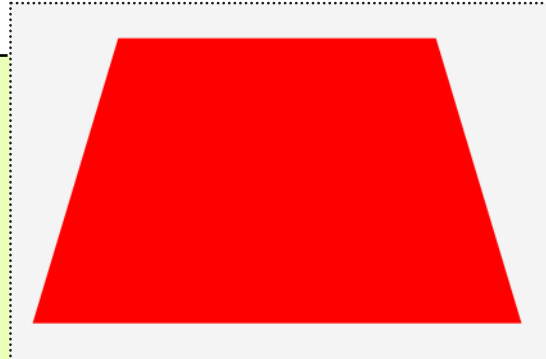
J'applique une rotation de 45° sur l'axe des X (axe horizontal)

```
#red{
  ...
  transform: rotateX(45deg);
  -webkit-transform: rotateX(45deg);
  -moz-transform: rotateX(45deg);
  -o-transform: rotateX(45deg);
  -ms-transform: rotateX(45deg);
}
```



La transformation provoque un aplatissement de la forme, ce qui est tout à fait normal. Pour obtenir une impression de 3D (de profondeur), il faut spécifier la perspective. Mais celle-ci devra être spécifiée sur le conteneur de notre boîte rouge.

```
#scene3D{
    ...
    perspective:500px;
    -webkit-perspective:500px;
    -moz-perspective:500px;
    -o-perspective:500px;
    -ms-perspective:500px;
}
```



Enregistrer et visualiser le résultat. Essayer d'ajouter du texte.

## Les transitions

Les transformations que nous venons de réaliser sont opérées de manière immédiate dans les navigateurs. Les nouveautés CSS3 permettent d'effectuer une transition dans les transformations, ce qui donne un effet visuel intéressant. Elle se définit de la manière suivante:

**transition-property: prop1, prop2 ... propN** : Liste des propriétés sur lesquels on applique la transition.

**transition-duration : Ns** : durée de la transition en seconde.

**transition-timing-function : value** : type de fonction de transition. Les valeurs peuvent être:

- *linear* : vitesse constante du début à la fin
- *ease-in* : la vitesse de transition augmente
- *ease-out* : la vitesse de transition diminue
- *ease-in-out* : la vitesse de transition est lente au début et à la fin

**transition-delay : Ns** : temps d'attente avant déclenchement de la transition

Il est possible de définir toutes ces informations sur une même ligne définie de la manière suivante:

**transition : prop || durée || fonction || délai**

Pour notre 1er exemple nous allons appliquer notre transition sur un élément possédant une transformation précise à savoir un lien : a vers a:hover. Et pour ce faire nous l'appliquerons sur le menu navigation de notre exemple:

Ouvrir le CSS [assets/css/styles\\_structure.css](#) et ajouter la transition suivante :

```
nav li a{
    ...
    transition-property:background;
    transition-duration:1s;
    transition-timing-function:ease-out;
    transition-delay:0s;
```

```

-webkit-transition-property:background;
-webkit-transition-duration:1s;
-webkit-transition-timing-function:ease-out;
-webkit-transition-delay:0s;

-o-transition-property:background;
-o-transition-duration:1s;
-o-transition-timing-function:ease-out;
-o-transition-delay:0s;

-moz-transition-property:background;
-moz-transition-duration:1s;
-moz-transition-timing-function:ease-out;
-moz-transition-delay:0s;
}

```

#### Enregistrer et visualiser le résultat.

Nous pouvons remarquer qu'au survol des menus la couleur passe du noir au orange de manière progressive. Attention la definition de la transition dans ce cas se place dans la propriété lien et pas lien survolé.

Dans l'exemple qui suit nous allons définir un diaporama et au survol de chaque miniatures nous effectuerons une transition vers une rotation et une nouvelle echelle.

Créer un nouveau document [chap5/07-transition.html](#) ainsi que la feuille de style [assets/css/styles\\_diaporama.css](#).

Dans la page html insertion du diaporama :

```

<div id="diaporama">
  <ul class="galerie">
    <li><a href="#"></a></li>
    <li><a href="#"></a></li>
    <li><a href="#"></a></li>
    <li><a href="#"></a></li>
    <li><a href="#"></a></li>
    <li><a href="#"></a></li>
    <li><a href="#"></a></li>
    <li><a href="#"></a></li>
    <li><a href="#"></a></li>
  </ul>
</div>

```

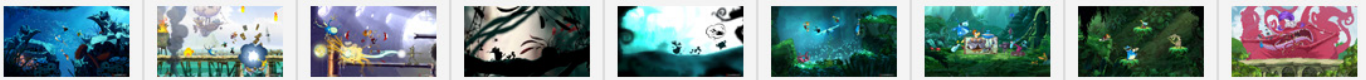
```

#diaporama{
  float:left;
  padding:20px;
}

```

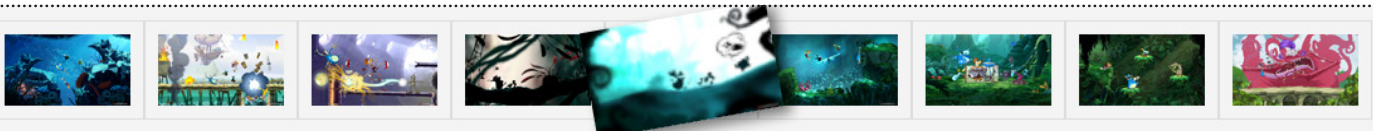
```
ul.galerie li{
  float:left;
  margin:0, 10px;
  padding: 10px;
  border: 1px solid #ddd;
  list-style:none;
}
ul.galerie li a img{
  float:left;
  width:100px
}
```

Enregistrer et visualiser le résultat.



Définition d'une image survolé

```
ul.galerie li a: hover img{
  /* grossissement */
  -webkit-transform: scale(1.5) rotate(-10deg);
  -moz-transform: scale(1.5) rotate(-10deg);
  -o-transform: scale(1.5) rotate(-10deg);
  -ms-transform: scale(1.5) rotate(-10deg);
  transform: scale(1.5) rotate(-10deg);
  /* ombre */
  -webkit-box-shadow: 4px 4px 10px rgba(0,0,0,0.5);
  -moz-box-shadow: 4px 4px 10px rgba(0,0,0,0.5);
  -o-box-shadow: 4px 4px 10px rgba(0,0,0,0.5);
  -ms-box-shadow: 4px 4px 10px rgba(0,0,0,0.5);
  box-shadow: 4px 4px 10px rgba(0,0,0,0.5);
}
```



Application de la transition

```
ul.galerie li a img{
  /* transition */
  -webkit-transition: -webkit-transform 0.2s ease-in-out;
  -moz-transition: -moz-transform 0.2s ease-in-out;
  -o-transition: -o-transform 0.2s ease-in-out;
  transition: transform 0.2s ease-in-out;
}
```

## Les animations

La différence entre animations et transitions CSS n'est pas immédiatement apparente, car les deux propriétés créent une illusion de mouvement. Dans le chapitre précédent, la transition est provoquée par le survol d'un menu ou bien d'une image, donc réalisé par un évènement utilisateur. En aucun cas on ne peut spécifier un nombre de déclenchement; exemple : faire clignoter un lien trois fois lors du survol, ici ce n'est pas une transition que nous utiliserons mais une animation.

Pour réaliser une animation nous aurons besoin de placer dans les propriétés de l'objet animé:

**animation-name** : nom de l'animation

**animation-duration** : durée de l'animation

**animation-iteration-count** : nombre de fois ou l'animation est réalisée

**animation-timing-function** : type de fonction d'animation

- *linear* : vitesse constante du début à la fin
- *ease-in* : la vitesse de transition augmente
- *ease-out* : la vitesse de transition diminue
- *ease-in-out* : la vitesse de transition est lente au début et à la fin

**animation-delay** : délai d'attente avant exécution de l'animation.

Une fois ces paramètres réglés nous devons définir l'animation en elle même

**@-nom\_animation{**

**from {**

définition de la position de départ

**}**

**X% {**

définition de la position à X% de l'animation. La position X% n'est pas obligatoire mais est très pratique lorsque l'on veut faire une animation en boucle ou bien la rotation d'un objet pour lui donner un sens.

**}**

**to {**

définition de la position d'arrivée

**}**

**}**

### L'effet BOUNCE

Dans l'exemple qui suit nous allons animer une icône en boucle avec un effet «Bounce» (effet de rebond).

Créer un nouveau document [chap5/08-animation.html](#) ainsi que la feuille de style [assets/css/styles\\_animation.css](#).

```

<div id="icoList">
  <div id="icoChrome"></div>
  <div id="icoMoz"></div>
  <div id="icoIE"></div>
  <div id="icoOpera"></div>
  <div id="icoSafari"></div>
</div>

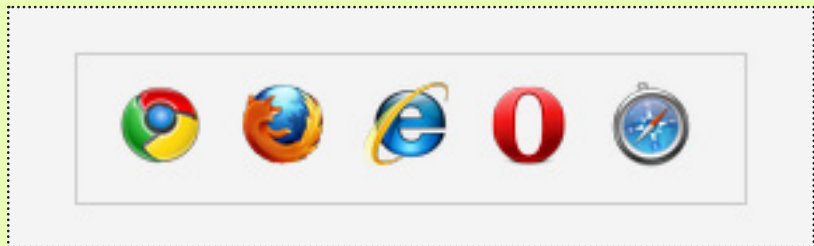
```

Puis appliquer le css suivant :

```

#icoList{
  margin: 20px auto 0 auto;
  padding: 10px;
  border: 1px solid #CCC;
  width: 230px;
  height: 40px;
}
#icoList div{
  position: relative;
  width: 32px;
  height: 32px;
  border: 0;
  float: left;
  margin: 7px;
}
#icoChrome{
  background: url('../images/chrome1.png');
}
#icoMoz{
  background: url('../images/firefox1.png');
}
#icoIE{
  background: url('../images/ie1.png');
}
#icoOpera{
  background: url('../images/opera1.png');
}
#icoSafari{
  background: url('../images/safari1.png');
}

```



Nous allons à présent définir l'effet de rebond sur l'icone Opéra. D'abord appel de l'animation

```

#icoOpera{
  ...
  /* appel animation */
  animation: bounce 0.7s ease infinite;
}

```

```

-webkit-animation: bounce 0.7s ease infinite;
-o-animation: bounce 0.7s ease infinite;
-moz-animation: bounce 0.7s ease infinite;
-ms-animation: bounce 0.7s ease infinite;
}
@keyframes bounce{
  from {top: 0px;}
  50% {top: -10px;}
  to {top: 0px;}
}
@-webkit-keyframes bounce{
  from {top: 0px;}
  50% {top: -10px;}
  to {top: 0px;}
}
@-o-keyframes bounce{
  from {top: 0px;}
  50% {top: -10px;}
  to {top: 0px;}
}
@-moz-keyframes bounce{
  from {top: 0px;}
  50% {top: -10px;}
  to {top: 0px;}
}
@-ms-keyframes bounce{
  from {top: 0px;}
  50% {top: -10px;}
  to {top: 0px;}
}

```

Enregistrer et visualiser le résultat.

Notre animation est ajoutée: elle dure 0.7secondes, avec une méthode d'accélération **ease** et de manière **infinite**. Il est possible d'ajouter cette animation sur chaque icone avec un délai de lancement : **exemple pour ico-moz : animation: bounce 0.7s ease-in-out infinite 0.1s;**

### Slider «photos en boucles»

Pour la réalisation du slider nous allons utiliser une **<section>** de type conteneur qui ne laissera entrevoir que certaines images. Les images seront placées au coeur d'une liste que nous déplacerons à tempo régulier.

Afin de donner l'effet «photos en boucles» nous allons lister 6 images puis recopier les 3 premières images à la fin du slideshow à déplacer (soit 9 images au total).

Créer un nouveau document <chap5/08-animation-slider.html> ainsi que la feuille de style [assets/css/styles\\_animation\\_slider.css](assets/css/styles_animation_slider.css).

Dans la page HTML ajouter le code suivant



```

<section id="galerie">
  <ul class="slider">
    <li></li>
    <li></li>
    <li></li>
    <li></li>
    <li></li>
    <li></li>
    <li></li>
    <li></li>
    <li></li>
  </ul>
</section>

```

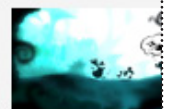
Puis appliquer le css suivant :

```

#galerie{
  width:200px;
  height:100px;
  border:1px solid #CCC;
  margin:20px auto 0px auto;
}
#galerie .slider{
  display: table;
  width:1260px;
  padding-left:25%;
}
#galerie .slider li{
  display: table-cell;
  width:140px;
  height:70px;
  list-style:none;
}
#galerie .slider li img{
  margin-top:7px;
  display: inline-block;
}

```

Enregistrer et visualiser le résultat.



Pour cacher le slider en dehors de la section: ajouter au CSS:

```
#galerie{  
  ...  
  overflow:hidden;  
}
```

Puis nous allons définir l'animation de la manière suivante

```
@keyframes slideAnim{  
  from,11%,to {margin-left:-140px;}  
  15%,26%{margin-left:-280px;}  
  30%,41%{margin-left:-420px;}  
  45%,56%{margin-left:-560px;}  
  65%,76%{margin-left:-700px;}  
  80%,91%{margin-left:-840px;}  
  99.99%{margin-left:-980px;}  
}
```

Au démarrage de l'animation (**from**) nous décalons le slide de 140px (ce qui correspond à la largeur d'un **<ul>** contenant l'image, ce qui permet d'afficher en premier l'image n°2. A 11% de l'animation la position du slider est toujours de -140px, ce qui donne l'impression d'une image fixe pendant un peu plus de 2 secondes, puis entre 11% et 15% le slider effectue un glissement de -140px à -280px puis garde cette position jusqu'à 26% de l'animation et ainsi de suite.

A 99.99% la position -980px correspond visuellement à la même position que -140px. Juste avant la fin de l'animation, on déplace tout le bloc en position de départ très rapidement (en 0,01% dans ce cas). Cela provoque une superposition des images invisible à l'oeil nu. L'animation reprend donc depuis le début, mais on a l'impression qu'elle continue...

Il ne nous reste plus qu'à placer l'animation dans le slider:

```
#galerie .slider{  
  ...  
  animation: slideAnim 20s ease 0s infinite;  
  -webkit-animation: slideAnim 20s ease-out 0s infinite;  
  -moz-animation: slideAnim 20s ease-out 0s infinite;  
  -ms-animation: slideAnim 20s ease-out 0s infinite;  
  -o-animation: slideAnim 20s ease-out 0s infinite;  
}
```

Il ne reste plus qu'à définir nos keyframes pour l'ensemble des navigateurs :

```
@-webkit-keyframes slideAnim{  
  from,11%,to {margin-left:-140px;}  
  15%,26%{margin-left:-280px;}  
  30%,41%{margin-left:-420px;}  
  45%,56%{margin-left:-560px;}  
  65%,76%{margin-left:-700px;}
```

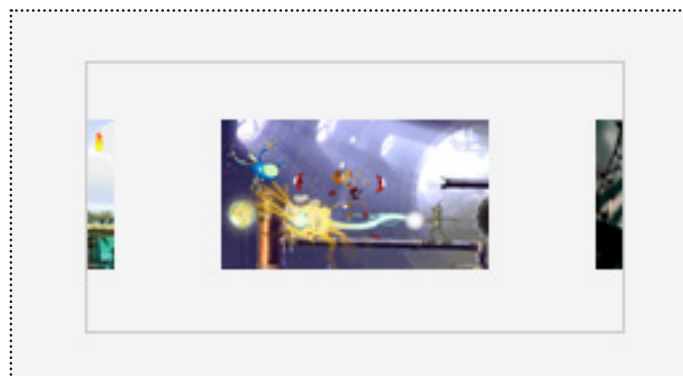
```
80%,91%{margin-left:-840px;}
99.99%{margin-left:-980px;}
}

@-moz-keyframes slideAnim{
  from,11%,to {margin-left:-140px;}
  15%,26%{margin-left:-280px;}
  30%,41%{margin-left:-420px;}
  45%,56%{margin-left:-560px;}
  65%,76%{margin-left:-700px;}
  80%,91%{margin-left:-840px;}
  99.99%{margin-left:-980px;}
}

@-ms-keyframes slideAnim{
  from,11%,to {margin-left:-140px;}
  15%,26%{margin-left:-280px;}
  30%,41%{margin-left:-420px;}
  45%,56%{margin-left:-560px;}
  65%,76%{margin-left:-700px;}
  80%,91%{margin-left:-840px;}
  99.99%{margin-left:-980px;}
}

@-o-keyframes slideAnim{
  from,11%,to {margin-left:-140px;}
  15%,26%{margin-left:-280px;}
  30%,41%{margin-left:-420px;}
  45%,56%{margin-left:-560px;}
  65%,76%{margin-left:-700px;}
  80%,91%{margin-left:-840px;}
  99.99%{margin-left:-980px;}
}
```

Enregistrer et visualiser le résultat.



# Multimédia

Audio

Vidéo

Depuis l'arrivée de Youtube et Dailymotion, il est devenu courant aujourd'hui de regarder des vidéos sur des sites web. Il faut dire que l'arrivée du haut débit a aidé à démocratiser les vidéos sur le Web.

Cependant, aucune balise HTML ne permettait jusqu'ici de gérer la vidéo. Il fallait à la place utiliser un plugin, comme Flash. Encore aujourd'hui, Flash reste de loin le moyen le plus utilisé pour regarder des vidéos sur Youtube, Dailymotion, Vimeo et ailleurs. Mais utiliser un plugin a de nombreux défauts : on dépend de ceux qui gèrent le plugin (en l'occurrence, l'entreprise Adobe, qui possède Flash), on ne peut pas toujours contrôler son fonctionnement, il y a parfois des failles de sécurité. . . Au final, c'est assez lourd.

C'est pour cela que deux nouvelles balises standard ont été créées en HTML5 : `<video>` et `<audio>`

## La lecture audio

La balise **<audio>** que nous allons découvrir est reconnue par tous les navigateurs récents, y compris Internet Explorer à partir de la version 9 (IE9).

En théorie, il suffit d'une simple balise pour jouer un son sur notre page.

*Créer un nouveau document [chap6/01-fichier-audio.html](#).*

```
<audio src="test.mp3"></audio>
```

*Enregistrer et visualiser le résultat.*

En théorie il ne se passe rien. En effet, le navigateur va seulement télécharger les informations générales sur le fichier (on parle de **métadonnées**) mais il ne se passera rien de particulier.

Pour que l'on puisse entendre le fichier audio nous allons ajouter un attribut supplémentaire à la balise:

**autoplay** : la musique sera jouée dès le chargement de la page. Évitez d'en abuser, c'est en général irritant d'arriver sur un site qui joue de la musique tout seul !

```
<audio src="test.mp3" autoplay="autoplay"></audio>
```

*Enregistrer et visualiser le résultat.*

La musique se lance mais l'internaute n'a aucun pouvoir sur celle-ci, et la musique peut devenir très vite perturbante pour lui. Il risque de changer tout simplement de site.

Nous allons indiquer au navigateur que l'on veut afficher les controls du player audio:

**controls** : pour ajouter les boutons «Lecture», «Pause» et la barre de défilement. Cela peut sembler indispensable, et vous vous demandez peut-être pourquoi cela n'y figure pas par défaut, mais certains sites web préfèrent créer eux-mêmes leurs propres boutons et commander la lecture avec du JavaScript.

```
<audio src="test.mp3" autoplay="autoplay" controls="controls"></audio>
```

*Enregistrer et visualiser le résultat.*

Les boutons par défaut sont définis par les navigateurs ce qui donne un aspect différent selon les navigateurs. Pour une harmonie entre les navigateurs vous pourrez avoir envie de définir vos propres

boutons.

```
<audio id="player" src="test.mp3"></audio>
<div>
  <button onclick="document.getElementById('player').play();">Play</button>
  <button onclick="document.getElementById('player').pause();">Pause</button>
  <button onclick="document.getElementById('player').volume+=0.1;">Volume up</button>
  <button onclick="document.getElementById('player').volume-=0.1;">Volume Down</button>
</div>
```

*Enregistrer et visualiser le résultat.*

Il est aussi possible que pour une musique d'ambiance de notre site on veuille écouter notre fichier audio en bloc :

**loop** : la musique sera jouée en boucle.

```
<audio src="test.mp3" autoplay="autoplay" controls="controls" loop="loop"></audio>
```

*Enregistrer et visualiser le résultat.*

Au chargement de la page il se peut que nous souhaitons ne pas lancer la musique mais que celle-ci soit chargée en tâche de fond sans que l'internaute puisse s'en rendre compte. Ceci est le rôle de l'attribut **preload** :

**preload** : indique si la musique peut être préchargée dès le chargement de la page ou non. Cet attribut peut prendre les valeurs :

- **auto (par défaut)** : le navigateur décide s'il doit précharger toute la musique, uniquement les métadonnées ou rien du tout.
- **metadata** : charge uniquement les métadonnées (durée, etc.).
- **none** : pas de préchargement. Utile si vous ne voulez pas gaspiller de bande passante sur votre site.

*On ne peut pas forcer le préchargement de la musique, c'est toujours le navigateur qui décide. Les navigateurs mobiles, par exemple, ne préchargent jamais la musique pour économiser la bande passante (le temps de chargement étant long sur un portable).*

Il reste maintenant à corriger les différentes erreurs engendrées par les nombreux navigateurs. Dans un premier temps nous remarquons que le format **.mp3** n'est pas reconnu par Firefox, mais qui reconnaît le format **.ogg**.

```
<audio controls="controls" >
  <source src="BigBuck.ogg" type="audio/ogg" />
  <source src="BigBuck.mp3" type="audio/mpeg" />
</audio>
```

*Enregistrer et visualiser le résultat.*

Bien sûr il reste toujours les anciennes versions d'IE qui ne reconnaissent pas le player audio. La parade sera d'insérer un Flash et en dernier recours un lien sur le téléchargement du morceau.

```
<audio controls=»controls» >
  <source src=»BigBuck.ogg» type=»audio/ogg» />
  <source src=»BigBuck.mp3» type=»audio/mpeg» />
  <object type=»application/x-shockwave-flash» data=»player.swf?soundFile=BigBuck.mp3»
    <param name=»movie» value=»player.swf?soundFile=BigBuck.mp3» >
    <a href=»BigBuck.mp3»>Télécharger la chanson</a>
  </object>
</audio>
```

*Enregistrer et visualiser le résultat.*

## La lecture vidéo

La balise `<video>` que nous allons découvrir est reconnue par tous les navigateurs récents, y compris Internet Explorer à partir de la version 9 (IE9).

Il suffit d'une simple balise `<video>` pour insérer une vidéo dans la page :

*Créer un nouveau document [chap6/02-fichier-vidéo.html](#).*

```
<video src=»BigBuck.mp4» controls width=»360» height=»240»></video>
```

*Enregistrer et visualiser le résultat.*

De manière général la balise **<video>** se rapproche de la balise `<audio>` et possède les mêmes attributs et les mêmes problématiques liés au format.

**controls** : pour ajouter les boutons Lecture , Pause et la barre de défilement. Cela peut sembler indispensable, mais certains sites web préfèrent créer eux-mêmes leurs propres boutons et commander la lecture avec du JavaScript. En ce qui nous concerne, ce sera largement suffisant !

**width** : pour modifier la largeur de la vidéo.

**height** : pour modifier la hauteur de la vidéo.

**loop** : la vidéo sera jouée en boucle.

**autoplay** : la vidéo sera jouée dès le chargement de la page. Là encore, évitez d'en abuser, c'est en général irritant d'arriver sur un site qui lance quelque chose tout seul !

**preload** : indique si la vidéo peut être préchargée dès le chargement de la page ou non. Cet attribut peut prendre les valeurs :

**auto** (par défaut) : le navigateur décide s'il doit précharger toute la vidéo, uniquement les métadonnées ou rien du tout.

**metadata** : charge uniquement les métadonnées (durée, dimensions, etc.).

**none** : pas de préchargement. Utile si vous souhaitez éviter le gaspillage de bande passante sur votre site.

Nous pouvons ajouter un dernier attribut:

**poster** : image à afficher à la place de la vidéo tant que celle-ci n'est pas lancée. Par défaut, le navigateur prend la première image de la vidéo mais, comme il s'agit souvent d'une image noire ou d'une image peu représentative de la vidéo, je vous conseille d'en créer une !

```
<video controls width=»360» height=»240» poster=»poster320.jpg»>
  <source src=»BigBuck.ogg» type=»video/ogg» />
  <source src=»BigBuck.mp4» type=»video/mp4» />
  <h1><a href=»BigBuck.mp4»>Charger la vidéo</h1>
</video>
```

### Enregistrer et visualiser le résultat.

Une des principales force de pouvoir manier directement notre vidéo au sein de notre page HTML c'est la possibilité de lui appliquer un CSS.

Dans notre fichier HTML placer une balise <div> autour de la balise <video>.

```
<div id=»wrap» >
  <video controls width=»360» height=»240» poster=»poster320.jpg»>
    ...
  </video>
</div>
```

### Créer un nouveau document CSS `assets/css/ styles_video.css`

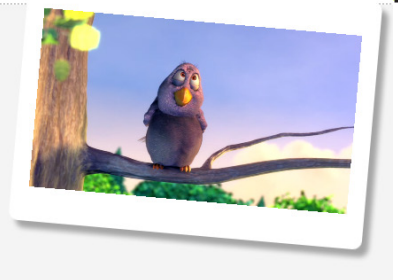
```
#wrap{
  float:right;
  width:360px;
  height:240px;
  padding:20px;
  background:#fff;
  margin:-20px 30px;

  box-shadow:8px 8px 8px rgba(0,0,0,0.33);
  border-radius:10px;

  -webkit-transform: rotate(5deg);
  -moz-transform: rotate(5deg);
  -o-transform: rotate(5deg);
  -ms-transform: rotate(5deg);
  transform: rotate(5deg);
}
```



**HTML5**  
Page Video





# Chapitre 3 : Langage JavaScript

## 1. Introduction

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. Javascript n'est donc pas propre aux navigateurs de Netscape (bien que cette firme en soit un fervent défenseur). Microsoft l'a d'ailleurs aussi adopté à partir de son Internet Explorer 3. On le retrouve, de façon améliorée, dans Explorer 4.

Les versions de Javascript se sont succédées avec les différentes versions de Netscape : Javascript pour Netscape 2, Javascript 1.1 pour Netscape 3 et Javascript 1.2 pour Netscape 4. Ce qui n'est pas sans poser certains problèmes de compatibilité, selon le browser utilisé, des pages comportant du code Javascript. Mais consolons nous en constatant qu'avec MSIE 3.0 ou 4.0 et la famille Netscape, une très large majorité d'internautes pourra lire les pages comprenant du Javascript.

L'avenir de Javascript est entre les mains des deux grands navigateurs du Web et en partie lié à la guerre que se livrent Microsoft et Netscape. On s'accorde à prédire un avenir prometteur à ce langage surtout de par son indépendance vis à vis des ressources du serveur.

## 2. Le Javascript minimum

### 2.1. La balise <SCRIPT>

De ce qui précède, vous savez déjà que votre script vient s'ajouter à votre page Web. Le langage Html utilise des tags ou balises pour "dire" au browser d'afficher une portion de texte en gras, en italique, etc.

Dans la logique du langage Html, il faut donc signaler au browser par une balise, que ce qui suit est un script et que c'est du Javascript (et non du VBScript). C'est la balise <SCRIPT LANGUAGE="Javascript">.

De même, il faudra informer le browser de la fin du script. C'est la balise </SCRIPT>.

### 2.2. Les commentaires

Il vous sera peut-être utile d'inclure des commentaires personnels dans vos codes Javascript. C'est même vivement recommandé comme pour tous les langages de programmation (mais qui le fait vraiment ?).

Javascript utilise les conventions utilisées en C et C++ soit

// commentaire

Tout ce qui est écrit entre le // et la fin de la ligne sera ignoré.

Il sera aussi possible d'inclure des commentaires sur plusieurs lignes avec le code

/\* commentaire sur

plusieurs lignes \*/

Ne confondez pas les commentaires Javascript et les commentaires Html (pour rappel <!-- ...-->).

### **2.3. Masquer le script pour les anciens browsers**

Les browsers qui ne comprennent pas le Javascript (et il y en a encore) ignorent la balise <script> et vont essayer d'afficher le code du script sans pouvoir l'exécuter. Pour éviter l'affichage peu esthétique de ses inscriptions cabalistiques, on utilisera les balises de commentaire du langage Html <!-- ... -->.

Votre premier Javascript ressemblera à ceci :

```
<SCRIPT LANGUAGE="javascript">
<!-- Masquer le script pour les anciens browsers
...
programme Javascript
...
// Cesser de masquer le script -->
</SCRIPT>
```

### **2.4 Où inclure le code en Javascript ?**

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.

### **2.5 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 boîte 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.

## **2.6 Votre première page Html avec du Javascript**

```
<HTML>
  <HEAD>
    <TITLE>Mon premier Javascript</TITLE>
  </HEAD>
  <BODY>
    Bla-bla en Html
    <SCRIPT LANGUAGE="Javascript">
      <!--
        alert("votre texte");
      //-->
    </SCRIPT>
    Suite bla-bla en Html
  </BODY>
</HTML>
Html normal
```

## **2.7 Quelques 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 souhaitez utiliser des guillemets dans vos chaînes de caractères, tapez \" ou \' pour les différencier vis à vis du compilateur.

## **2.8 Versions du langage Javascript**

Avec les différentes versions déjà existantes (Javascript 1.0, Javascript 1.1 et Javascript 1.2), on peut imaginer des scripts adaptés aux différentes versions mais surtout aux différents navigateurs ;

```
<SCRIPT LANGUAGE="Javascript">
// programme pour Netscape 2 et Explorer 3
var version="1.0";
</SCRIPT>
<SCRIPT LANGUAGE="Javascript1.1">
// programme pour Netscape 3 et Explorer 4
var version=1.1;
</SCRIPT>
<SCRIPT LANGUAGE="Javascript1.2">
// programme pour Netscape 4
```

```
var version=1.2;  
</SCRIPT>  
<SCRIPT LANGUAGE="Javascript">  
document.write('Votre browser supporte le Javascript ' + version);  
</SCRIPT>
```

### **3. Utiliser des variables**

Les variables contiennent des données qui peuvent être modifiées lors de l'exécution d'un programme. On y fait référence par le nom de cette variable.

Un nom de variable doit commencer par une lettre (alphabet ASCII) ou le signe\_ et se composer de lettres, de chiffres et des caractères \_ et \$ (à l'exclusion du blanc). Le nombre de caractères n'est pas précisé. Pour rappel Javascript est sensible à la case. Attention donc aux majuscules et minuscules!

#### **3.1 La déclaration de variable**

Les variables peuvent se déclarer de deux façons : soit de façon explicite. On dit à Javascript que ceci est une variable.

La commande qui permet de déclarer une variable est le mot var. Par exemple : var Numero = 1  
var Prenom = "Luc" soit de façon implicite. On écrit directement le nom de la variable suivi de la valeur que l'on lui attribue et Javascript s'en accommode. Par exemple : Numero = 1  
Prenom = "Luc"

Attention! Malgré cette apparente facilité, la façon dont on déclare la variable aura une grande importance pour la "visibilité" de la variable dans le programme Javascript. Voir à ce sujet, la distinction entre variable locale et variable globale dans le Javascript avancé de ce chapitre. Pour la clarté de votre script et votre facilité, on ne peut que conseiller d'utiliser à chaque fois le mot var pour déclarer une variable.

#### **3.2 Les données sous Javascript**

Javascript utilise 4 types de données :

Type Description

Des nombres Tout nombre entier ou avec virgule tel que 22 ou 3.1416

Des chaînes de caractères

Toute suite de caractères comprise entre guillemets telle que "suite de caractères"

Des booléens

Les mots true pour vrai et false pour faux

Le mot null Mot spécial qui représente pas de valeur Notons aussi que contrairement au langage C ou C++, Il ne faut pas déclarer le type de données d'une variable.

On n'a donc pas besoin de int, float, double, char et autres long en Javascript.

## 4. Les opérateurs

### 4.1 Les opérateurs arithmétiques

Signe	Nom	Signification	Exemple	Résultat
+	plus	addition	$x + 3$	14
-	moins	soustraction	$x - 3$	8
*	multiplié par	multiplication	$x * 2$	22
/	divisé	par division	$x / 2$	5.5
%	modulo	reste de la division par	$x \% 5$	1
=	a la valeur	affectation	$x = 5$	5

### 4.2 Les opérateurs de comparaison

Signe	Nom	Exemple	Résultat
==	égal	$x == 11$	true
<	inférieur	$x < 11$	false
<=	inférieur ou égal	$x <= 11$	true
>	supérieur	$x > 11$	false
>=	supérieur ou égal	$x >= 11$	true
!=	différent	$x != 11$	false

### 4.3 Les opérateurs associatifs

Signe	Description	Exemple	Signification	Résultat
+=	plus égal	$x += y$	$x = x + y$	16
-=	moins égal	$x -= y$	$x = x - y$	6
*=	multiplié égal	$x *= y$	$x = x * y$	55
/=	divisé égal	$x /= y$	$x = x / y$	2.2

### 4.4 Les opérateurs logiques

Signe	Nom	Exemple	Signification
&&	et	$(condition1) \&\& (condition2)$	condition1 et condition2
	ou	$(condition1)    (condition2)$	condition1 ou condition2

#### **4.5 Les opérateurs d'incrémentation**

Signe	Description	Exemple	Signification	Résultat
x++	incrémentation (x++ est le même que x=x+1)	y = x++	3 puis plus 1	4
x--	décrémentation (x-- est le même que x=x-1)	y= x--	3 puis moins 1	2

#### **4.6 La priorité des opérateurs Javascript**

Les opérateurs s'effectuent dans l'ordre suivant de priorité (du degré de priorité le plus faible ou degré de priorité le plus élevé).

Dans le cas d'opérateurs de priorité égale, de gauche à droite.

Opération	Opérateur
,	virgule ou séparateur de liste
= += -= *= /= %=	affectation
? :	opérateur conditionnel
	ou logique
&&	et logique
== !=	égalité
< <= >= >	relationnel
+ -	addition soustraction
* /	multiplier diviser
! - ++ --	unaire
()	parenthèses

### **5. Les fonctions**

#### **5.1 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.

### **5.2 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.

### **5.3 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>.

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. Les événements**

Passons en revue différents événements implémentés en Javascript.

Description	Evénement
Lorsque l'utilisateur clique sur un bouton, un lien ou tout autre élément.	Clik
Lorsque la page est chargée par le browser ou le navigateur.	Load
Lorsque l'utilisateur quitte la page.	Unload
Lorsque l'utilisateur place le pointeur de la souris sur un lien ou tout autre élément.	MouseOver
Lorsque le pointeur de la souris quitte un lien ou tout autre élément.	MouseOut
Attention : Javascript 1.1 (donc pas sous MSIE 3.0 et Netscape 2).	
Lorsque un élément de formulaire a le focus c-à-d devient la zone d'entrée active.	Focus
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
Lorsque la valeur d'un champ de formulaire est modifiée.	Change
Lorsque l'utilisateur sélectionne un champ dans un élément de formulaire.	Select
Lorsque l'utilisateur clique sur le bouton Submit pour envoyer un formulaire.	Submit

### 6.1 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 boîte d'alerte avec le message indiqué.

### 6.2 Exemple

```
<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>
```

### 6.3. Gestionnaires d'événement disponibles en Javascript

Il nous semble utile dans cette partie "avancée" de présenter la liste des objets auxquels correspondent des gestionnaires d'événement bien déterminés.



Objets	Gestionnaires d'événement disponibles
Fenêtre	onLoad, onUnload
Lien hypertexte	onClick, onMouseOver, onMouseOut
Élément de texte	onBlur, onChange, onFocus, onSelect
Élément de zone de texte	onBlur, onChange, onFocus, onSelect
Élément bouton	onClick
Case à cocher	onClick
Bouton Radio	onClick
Liste de sélection	Blur, onChange, onFocus
Bouton Submit	onClick
Bouton Reset	onClick

## **7. Les Structures de contrôles**

### **7.1 les conditions**

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

### **7.2 L'expression for**

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

### **7.3 While**

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

## **8. Les formulaires**

Le code ci-dessous illustre la manipulation d'un formulaire avec javascript.

```
<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>
```

Lorsqu'on clique le bouton "contrôler", Javascript appelle la fonction controle() à laquelle on passe le formulaire dont le nom est form1 comme argument.

Cette fonction controle() définie dans les balises <HEAD> prend sous la variable test, la valeur de la zone de texte. Pour accéder à cette valeur, on note le chemin complet de celle-ci (voir le chapitre "Un peu de théorie objet"). Soit dans le document présent, il y a l'objet formulaire appelé form1 qui contient le contrôle de texte nommé input et qui a comme propriété l'élément de valeur value. Ce qui donne document.form1.input.value.

## **9. Les boîtes de dialogue ou de message**

Javascript met à votre disposition 3 boîtes de message :

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

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

### **9.1 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 boîte de dialogue dans laquelle est reproduite la valeur (variable et/ou chaîne de caractères) de l'argument qui lui a été fourni. Cette boîte 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.

### **9.2 La méthode prompt()**

Dans le même style que la méthode alert(), Javascript vous propose une autre boîte de dialogue, dans le cas présent appelée boîte 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 boîte 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.

### 9.3 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 ?")

## 10. 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.

On signale dans la littérature une limitation de la longueur des strings à 50/80 caractères. Cette limitation du compilateur Javascript peut toujours être contournée par l'emploi de signes + et la concaténation.

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 principale 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 la position x et la position y-1.
toLowerCase()	Transforme toutes les lettres en minuscules.
toUpperCase()	Transforme toutes les lettres en Majuscules.

## 11. L'objet Math

Méthode	Résultat
x=Math.abs(y);	renvoie la valeur absolue (valeur positive) de y.
x=Math.ceil(y);	renvoie l'entier supérieur ou égal à y.
x=Math.floor(y);	renvoie l'entier inférieur ou égal à y.
x=Math.round(y);	arrondit le nombre à l'entier le plus proche.
x=Math.max(y,z);	renvoie le plus grand des 2 nombres y et z.
x=Math.min(y,z);	renvoie le plus petit des 2 nombres y et z.
x=Math.pow(y,z);	calcule la valeur d'un nombre y à la puissance z.
x=Math.random();	renvoie la valeur d'un nombre aléatoire choisi entre 0 et 1.
x=Math.sqrt(y);	renvoie la racine carrée de y.
x=eval(variable);	évalue une chaîne de caractère sous forme de valeur numérique.

# Chapitre 4: Installation et Configuration de l'Environnement de Développement TP

## Configuration d'APACHE :

La configuration de Apache se fait de manière simple et ce via un seul et unique fichier de configuration. Le fichier de configuration du serveur web se nomme **httpd.conf** ( un fichier texte qui sera édité avec le bloc-notes) et est situé dans le sous-répertoire **conf** d'Apache.

Ce fichier contient les principaux éléments pour faire en sorte que votre serveur web tourne sans encombre. Une modification dans ce fichier peut rendre indisponible Apache. Voyons quelques paramètres paramétrables sans trop de difficulté. Tout d'abord, une chose bien utile si vous ne souhaitez pas utiliser le répertoire de base de Apache pour vos documents web. Par défaut, le sous-répertoire qui contient les pages web se nomme **htdocs**, si vous souhaitez modifier cela, repérez le paramètre « **DocumentRoot** » puis modifier comme ceci :

**DocumentRoot** "e:/projet/www"

On aura pris soit de créer le sous-répertoire **www** dans **e:/projet** avant même d'avoir fait la modification dans le fichier de configuration, sinon cela aurait pour effet de générer une erreur lors du lancement d'Apache.

Si pour une raison ou pour un autre, on souhaite modifier l'adresse e-mail de l'administrateur du serveur, on repère le paramètre **ServerAdmin** puis on lui indique en valeur une adresse e-mail (de préférence valide).

**ServerAdmin** [toto@nomdedomaine.com](mailto:toto@nomdedomaine.com)

Si l'on souhaite indiquer les fichiers qui seront traités comme des fichiers de base du serveur web, c'est-à-dire la page par défaut d'un répertoire web, nous pouvons modifier pour cela le paramètre **DirectoryIndex**.

**DirectoryIndex** index.htm index.html index.php index.php5

Ici, toutes les pages qui se nomment index.html, index.html, index.php ou index.php5 seront prises en compte par le serveur web comme page par défaut d'un site web. Pour faire en sorte que le visiteur est un minimum d'information concernant votre serveur lorsque une page d'erreur type 404 s'affiche, nous pouvons modifier la valeur du paramètre **ServerTokens**.

**ServerTokens** Prod

En donnant la valeur Prod cela permet de ne fournir que le nom du serveur, soit dans le cas présent Apache, il n'y aura aucune information concernant la version utilisée ni d'autres informations qui pourraient renseigner une personne mal intentionnée.

Par ailleurs, je veille à fournir une adresse e-mail qui pourrait permettre au visiteur de m'informer d'un éventuel problème sur le serveur. Pour ce faire je modifie la valeur du paramètre Server Signature. Comme ceci :

### **ServerSignature** Email

Ce qui au final lorsqu'un message d'erreur est affiché permet à tout visiteur de pouvoir prévenir l'administrateur du serveur.

Une option qui est très utile est l'utilisation du module status, un module est une fonction qui permet d'ajouter des fonctions à votre serveur web. Le module status permet dans le cas présent d'obtenir des informations en quasi temps réel sur l'état du serveur. Pour ce faire dans le fichier de configuration, je vais rechercher la ligne suivante :

**# LoadModule status\_module modules/mod\_status.so**

Dans le cas présent, la ligne est actuellement en commentaire puisque ayant un # en son début de ligne. Donc, on décommente tout d'abord la ligne :

**LoadModule status\_module modules/mod\_status.so**

Puis l'on recherche les quelques lignes ci-dessous (lignes qui dans leur version d'origine sont là aussi commentés #) :

```
<Location /server-status>  
SetHandler server-status  
Order deny,allow  
Deny from all  
Allow from 127.0.0.1  
</Location>
```

Ces quelques lignes permettent de rendre ou non disponible l'état du serveur. Le Deny from all permet tout d'abord interdit l'accès à tout le monde puis avec l'option Allow from **127.0.0.1** de l'autoriser uniquement à **127.0.0.1** (la consultation sera donc possible que depuis le serveur et non depuis une machine dans le réseau local par exemple).

Une fois la modification, on enregistre le fichier de configuration puis l'on ouvre son navigateur favori (Opéra par exemple). Dans la barre d'adresse : <http://localhost/server-status>

Ceci ayant un résultat comparable à cela :

Server Version: Apache  
Server Built: Oct 9 2005 19:16:56

```
Current Time: Sunday, 30-Oct-2005 21:55:41 Paris, Madrid
Restart Time: Sunday, 30-Oct-2005 21:55:37 Paris, Madrid
Parent Server Generation: 0
Server uptime: 4 seconds
1 requests currently being processed, 249 idle workers
```

III

Scoreboard Key:  
 \* " Waiting for Connection "s" Starting up "p" Reading Request

## Configuration PHP

Ceci fait, voyons les deux installations possible de php pour le faire travailler avec Apache.

- Soit installer PHP en tant que module d'Apache
- Soit installer en tant que programme CGI

Dans le cas présent, je vais procéder à l'installer en tant que module de mon serveur apache. Si vous avez jeter un oeil dans le fichier install.txt qui se trouve en racine de notre répertoire php5, vous aurez vu que cela n'est pas très différent au niveau de la modification du fichier de configuration apache (httpd.conf).

Pour ce faire, je vais devoir configurer Apache en conséquence. Ceci étant, je dois éditer le fichier de configuration d'Apache qui se situe dans e:\projet\apache2\conf\ et qui se nomme httpd.conf. A celui-ci, je dois lui ajouter les lignes suivantes :

**LoadModule php5\_module "e:/projet/php5/php5apache2.dll"**

**AddType application/x-httpd-php .php**

La première ligne (**LoadModule php5\_module "e:/projet/php5/php5apache2.dll"**) s'ajoute à la suite des autres **LoadModule** que vous trouverez dans le fichier.

Pour ce qui concerne la deuxième ligne (**AddType application/x-httpd-php .php**), elle s'ajoute à la suite des autres **AddType**.

Ceci fait, vous enregistrez le fichier de configuration ainsi modifié.

Nous allons à présent passer à la configuration du **php.ini** qui à l'origine se trouve dans le répertoire d'installation sous le nom de **php.ini-dist**, première chose, en faire une copie et renommez la dite copie en tant que **php.ini**. On édite le fichier **php.ini** pour modifier les lignes suivantes :

**extension\_dir = "./"** en **extension\_dir = "e:\projet\php5\ext"**

**;upload\_tmp\_dir =** en **upload\_tmp\_dir = e:\projet\php5\uploadtemp**

**;session.save\_path = "/tmp"** en **;session.save\_path = "e:\projet\php5\sessionssave"**

A noter que les deux dernières lignes ne sont nullement obligatoires, si l'on ne pense pas utiliser les fonctions qui font appel à ces valeurs. Dans un autre cas, on aura pris soin de créer le sous-répertoire **uploadtemp** et **sessionssave**.

Ceci étant nous pouvons à présent, faire un premier test pour savoir si notre serveur Apache à bien pris en compte le support PHP. Pour ce faire, nous allons créer un fichier **info.php** que nous placerons à la racine de notre serveur web.

Dans le cas présent, notre racine se situe dans le répertoire suivant **e:\projet\www** (répertoire que l'on a spécifié lors de la configuration d'Apache). Le fichier info.php contiendra la ligne suivante :

**<? phpinfo(); ?>**

La ligne ajoutée et le fichier modifié enregistré, ouvrez votre navigateur web favori et rendez-vous à l'url suivante :

**http://127.0.0.1/info.php** ou **http://localhost/info.php**

PHP Version 5.1.1



System	
Build Date	Nov 27 2005 21:34:13
Configure Command	cs script /nologo & configure.js "--enable-snapshot-build" "--with-gd=shared"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	
PHP API	20041225
PHP Extension	20050822
Zend Extension	220051025
Debug Build	no
Thread Safety	enabled
Zend Memory Manager	enabled
IPv6 Support	enabled
Registered PHP Streams	php, file, http, ftp, compress.zlib
Registered Stream Socket Transports	tcp, udp
Registered Stream Filters	conv.*:conv.*, string.intl3, string.toupper, string.tolower, string.strip_tags, convert.*, zlib.*

This program makes use of the Zend Scripting Language Engine:  
Zend Engine v2.1.0, Copyright (c) 1998-2005 Zend Technologies

Powered By



Cette page confirme la bonne prise en compte de PHP par Apache.

## Configuration de MySQL

Si vous pensez utiliser le serveur de données MySQL, il vous sera nécessaire de décommenter la ligne suivante :

```
;extension=php_mysql.dll en extension=php_mysql.dll
```

Par ailleurs, vous aurez pris soin de copier le fichier libmysql.dll dans le répertoire système soit c:\windows\system32 ou c:\winnt\system32 selon le système d'exploitation utilisé.

NB : Par défaut, MySQL n'est plus activé dans PHP5 ce qui explique la manipulation vue ci-dessus. Si vous obtenez un message similaire à celui-ci :

```
"Unable to load dynamic library './php_mysql.dll'"
```

C'est tout simplement parce que le fichier libmysql.dll n'a pu être trouvé par le système.



# Chapitre 5 Le Langage PHP

## 1. Introduction

Les services Web sont une nouvelle technologie permettant aux pages Web d'accéder à des applications distribuées. En offrant à la fois l'accès à des informations et à des fonctionnalités applicatives sous la forme d'un service, les services Web sont fournis et vendus en tant que flux de services auxquels il est possible d'accéder à partir de n'importe quelle plate-forme. La page Web qui se connecte au service Web s'appelle généralement un consommateur tandis que le service même s'appelle un fournisseur. La création de consommateurs de services Web se fait à l'aide des types de documents ColdFusion, ASP.NET et JSP (Java Server Pages), PHP, etc. Nous Etudierons dans le cadre de ce cours PHP.

### Un peu d'histoire

1994 : création de PHP/FI (Personal Home Page Tools/Form Interpreter) par Rasmus Lerdof (Canada) pour évaluer la consultation de son CV en ligne.

1995 : mise à disposition du code pour tout le monde.

1997 : redéveloppement du coeur par 2 étudiants (Andi Gutmans et Zeev Zuraski, Israël) pour donner PHP (PHP: Hypertext Processor) 3.0.

2002 : 8 millions de sites basés sur PHP.

2004 : 15 millions de sites basés sur PHP.

### Evolution du langage

Au départ basé sur le langage Perl, PHP a été réécrit ensuite en langage C.

Le moteur a été réécrit plusieurs fois.

D'abords procédural, le langage a évolué à partir de la version 4.0 pour intégrer les technologies objet.

### Principales caractéristiques

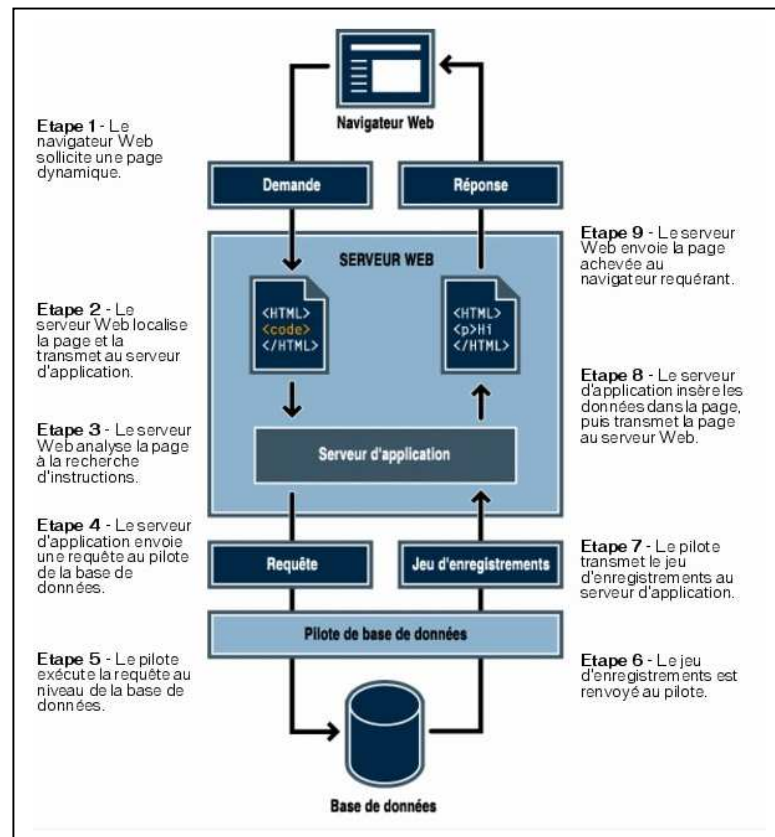
Le moteur PHP s'intègre au serveur web Apache : il prend alors en charge toutes les pages avec une extension .php.

PHP est un langage interprété : il n'y a pas de phase de compilation et les erreurs sont découvertes au moment de l'exécution.

La syntaxe est très inspirée du langage C, tout en apportant beaucoup de simplification, notamment dans la gestion des variables.

**Remarque :** dans le présent support, on n'aborde pas les caractéristiques objet de PHP ; le code décrit est compatible PHP 3.0 et versions ultérieures.

## 2. Principe des applications web



Les applications web fonctionnent grâce à des échanges réseau entre un logiciel client (le navigateur web) et un logiciel serveur (le serveur web).

Ces échanges sont basés sur le protocole HTTP : le navigateur envoie une requête au serveur web et reçoit du contenu à afficher.

La navigation sur un site est constituée d'un ensemble de requêtes/réponses qui s'enchaînent.

Il y a 2 types de requêtes : les requête de type GET (barre d'URL et clic sur un lien) et les requêtes de type POST (validation de formulaire).

### Apport de PHP

Les pages HTML sont des pages de contenu statique stockées dans le système de fichier du serveur.

Les pages PHP contiennent du code HTML, mais aussi du code PHP qui est exécuté sur le serveur au moment de la requête et produit du code HTML ; la page est dite dynamique, car le contenu final HTML renvoyé au navigateur peut changer selon le contexte.

PHP nécessite l'ajout et la configuration d'un module spécifique dans le serveur web Apache.

### Exemple simple

```
<?php echo "<?xml version=\"1.0\" encoding=\"ISO-8859-15\"?>" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr">
```

```
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-
    15" />
  <meta name="keywords" content="agronomie,INA-PG">
  <meta name="revisit-after" content="30 days" />
  <title>Ma première page PHP</title>
</head>
<body>
  <div>
    Bonjour nous sommes le :
    <?php
      $date = date("d-m-Y");
      echo "$date"; // Affichage de la date
    ?>
  </div>
</body>
</html>
```



Code HTML généré

Si on demande l'affichage du code source dans le navigateur :

```
<?xml version="1.0" encoding="ISO-8859-15"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-
      15" />
    <meta name="keywords" content="agronomie,INA-PG">
    <meta name="revisit-after" content="30 days" />
    <title>Ma première page PHP</title>
  </head>
  <body>
    <div>
```

```
Bonjour nous sommes le :  
O4-01-2006  
    </div>  
  </body>  
</html>
```

Le navigateur reçoit et interprète toujours du code HTML car le code PHP est évalué au niveau du serveur.

Contrairement à une page HTML, on ne peut donc pas tester une page PHP directement en l'ouvrant depuis le système de fichier sans passer par une connexion HTTP sur un serveur de type Apache.

### 3. Notions de base

Code PHP au sein d'une page HTML

- Le code PHP est inséré directement au sein d'une page contenant du HTML.
- Le code PHP doit cependant être contenu dans des balises spéciales : `<?php ... ?>` ou `<? ... ?>`.
- Du code HTML est produit grâce à l'instruction `echo`.
- On peut mettre plusieurs instructions au sein d'une même balise `<? ?>`.
- Une instruction est toujours terminée par un `;`.
- `<? echo $variable; ?>` peut être condensé en `<?= $variable ?>`.
- Pour être reconnu par l'interpréteur PHP, le fichier doit porter l'extension `.php`. Syntaxe de base
- L'instruction `echo` permet de générer de l'affichage HTML au sein d'une page.
- Les chaînes de caractères sont écrites entre `"` : on doit faire précéder un `"` d'un `\` pour l'inclure au sein d'une chaîne.
- Les noms de variables commencent toujours par le caractère `$`.
- Les variables n'ont pas besoin d'être déclarées, ni d'être typées ; elles sont créées implicitement lors de la première utilisation.
- Le langage fournit un certain nombre de fonction prédéfinies ; lors d'un appel de fonction, les paramètres sont passés entre `( )` et séparés par des `,`.
- Les commentaires sont écrits en commençant par `//` sur une ligne ou entre `/*` et `*/` sur plusieurs lignes.
- Les accolades `{ }` permettent de délimiter des blocs d'instructions.

**Remarque :** lorsqu'on respecte la norme XHTML, les caractères `<?` de la première ligne sont interprétés comme une balise PHP. Il faut donc obligatoirement générer cette première ligne avec une instruction PHP `echo`.

### 4. Structures de contrôle

PHP propose toutes les structures de contrôle classiques.

Tests conditionnels

```
<?
    if (is_numeric($texte)) {
        echo "La variable contient un nombre";
    }
    else {
        echo "La variable contient du texte";
    }
?>
```

On peut enchaîner des tests avec l'instruction elseif. On peut utiliser l'instruction switch lorsqu'on a beaucoup de conditions.

#### Schémas itératifs

```
<?
    for ( $i = 1; $i <= 10 ; $i++)
    {
        echo $i . ' ';
    }
    i = 1;

    while ($i<=10) {
        echo $i . ' ';
        $i++;
    }
}
```

## 5. Interactions avec l'utilisateur

### Principe général

Dans une application web, l'interaction avec l'utilisateur est essentiellement fondée sur la validation de formulaire HTML, qui provoque l'envoi de données vers le serveur et l'affichage de nouvelles informations en provenance de celui-ci.

Des technologies telles que Javascript permettent la gestion d'événements, mais celle-ci reste locale au navigateur et donc limitée.

Formulaire : côté client

Soit un fichier formulaire.php :

```
...
<body>
    <div>
        <form action="validation_formulaire.php" method="post">
            <p><label for="nom">Nom :</label>
            <input type="text" size="30" name="nom" /></p>
            <p><label for="prenom">Prénom :</label>
            <input type="text" size="20" name="prenom" /></p>
            <p><input type="submit" name="btAction" value="OK" />
            <input type="reset" name="btAnnuler" value="Annuler" /></p>
        </form>
    </div>
```

</body>

...

Il faut préciser dans la balise form le nom du fichier PHP qui traitera les données, ainsi que la méthode de transfert de ces données (POST conseillé).

Il est important de donner un nom à chaque élément du formulaire : ce nom permet ensuite de retrouver les valeurs associées saisies par l'utilisateur.

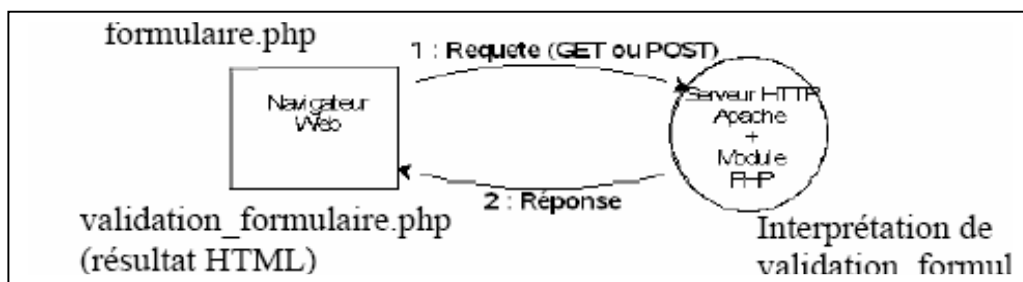
Formulaire : côté serveur

Le formulaire précédent est traité par validation\_formulaire.php :

...

```
<body>
  <div>
    <?
      if (empty($_POST["nom"])
        or empty($_POST["prenom"])) {
        echo "Vous devez saisir un nom et un prénom !";
      }
    else {
      $nom = $_POST["nom"];
      $prenom = $_POST["prenom"];
      echo "Bonjour $prenom $nom";
    }
  <?>
</div>
</body>
```

## Enchaînement



1. Le navigateur rassemble les informations saisies dans le formulaire (nom et prénom) et les envoie par une méthode POST au serveur.
2. Le serveur interprète validation\_formulaire.php avec les données reçues et génère un flux HTML résultat renvoyé au navigateur.

## 6. Formulaire validé par lui-même

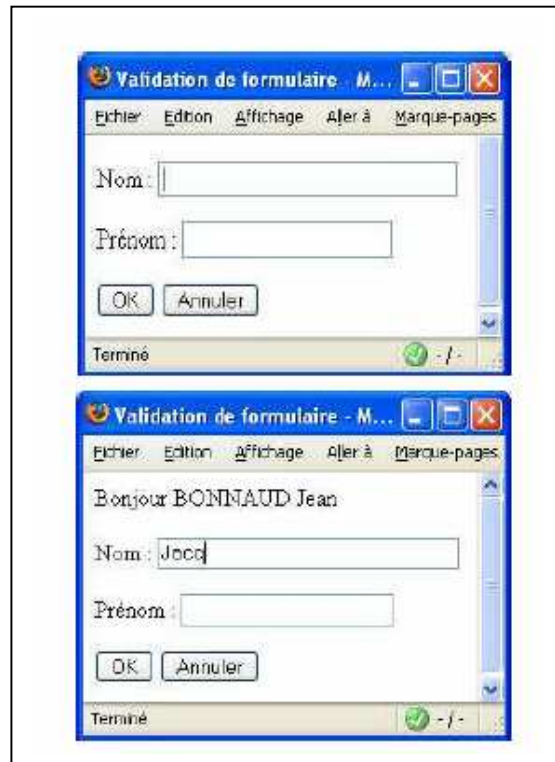
Principe

Dans l'exemple précédent, lorsque l'utilisateur s'est trompé ou s'il veut faire une nouvelle saisie, il soit revenir en arrière en utilisant la touche back du navigateur.

Il est souvent plus judicieux d'afficher le formulaire et le résultat sur une même page :

- lors du premier accès, la méthode GET est employée ;
- lors de la validation, la méthode POST est employée.

Exemple



...

```
<body>
  <div>
    <?
      if ($_SERVER["REQUEST_METHOD"] == "POST") {
        // on est appelé par la méthode POST, donc
        // en validation du formulaire
        if (empty($_POST["nom"])
            or empty($_POST["prenom"])) {
          echo "Vous devez saisir un nom et un prénom !";
        }
        else {
          $nom = $_POST["nom"];
          $prenom = $_POST["prenom"];
          echo "Bonjour $prenom $nom";
        }
      }
    </?>
  </div>
</body>
```

```
        }  
    ?>  
    <form action="toutenun.php" method="post">  
        <p><label for="nom">Nom :</label>  
        <input type="text" size="30" name="nom" />  
        </p>  
        <p><label for="prenom">Prénom :</label>  
        <input type="text" size="20" name="prenom" />  
        </p>  
        <p><input type="submit" name="btAction" value="OK" />  
        <input type="reset" name="btAnnuler" value="Annuler" />  
        </p>  
    </form>  
</div>  
</body>  
...  

```

## 7. Tableaux

Exemples précédents

Dans les précédents exemples de code, on a :

```
$_SERVER["REQUEST_METHOD"]  
$_POST["nom"]  
$_POST["prenom"]
```

**\$\_SERVER** et **\$\_POST** sont en fait des tableaux associatifs pré-remplis.

Tableaux associatifs

Un tableau associatif permet de faire correspondre des valeurs à des clés :

```
<?php  
    // Création d'un tableau de 3 éléments  
    $tableau = array(1=>"rouge", 2=>"bleu", 3=>"jaune");  
    // Ajout d'un élément à la fin du tableau  
    $tableau[] = "vert";  
    // Modification du 2ème élément  
    $tableau[2] = "vert";  
    // Affichage du 3ème élément  
    echo $tableau[3];  
    // Ajout d'un élément associé à "rien"  
    $tableau["rien"] = "transparent";  
    // affichage de tout tableau :  
    // Array ( [1] => rouge [2] => vert ... )  
    print_r($tableau);  
    // Parcours du tableau  
    foreach ($tableau as $index => $couleur) {  
        echo "<br /> $index => $couleur";  
    }  

```



```
}  
// Effacement du 2ème élément  
unset($tableau[2]);  
// Effacement de tout le tableau  
unset($tableau);
```

?>

### Quelques fonctions prédéfinies

Chaînes de caractères	<code>strlen(\$chaine)</code> : longueur d'une chaîne <code>strpos(\$chaine, \$car)</code> : position d'un caractère <code>strtolower(\$chaine)</code> : conversion minuscules <code>strtoupper(\$chaine)</code> : conversion majuscules
Dates	<code>getdate()</code> : date et heure <code>gettimeofday()</code> : heure actuelle <code>date(\$format, \$date)</code> : formatage
Tableaux	<code>count(\$tableau)</code> : nombre d'éléments <code>sort(\$tableau)</code> : tri

Fonctions définies par l'utilisateur

PHP permet de créer des fonctions :

```
<?php  
function factorielle($nombre) {  
    $resultat = 1;  
    for ( $i = 2; $i <= $nombre ; $i++) {  
        $resultat = $resultat * $i;  
    }  
    return $resultat;  
}
```

?>

On peut définir des fonctions dans des fichiers .php et les utiliser dans d'autres fichiers grâce à l'instruction include :

```
<?php  
include "fonctions.php";  
echo factorielle(5);
```

?>

## 8 Passage et transmission de variables

### 8.1-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>
<form method="post" action="destination.php">
  <input type="text" name="nom" size="12">
  <br> <input type="submit" value="OK">
</form>

<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 1 : 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 passé de formulaire.html vers resultat.php.

Exemple 2 :

L'exemple précédent peut tenir dans un seul fichier qui s'enverrait à lui même les paramètres  
Fichier formulaire.php

```
<?
if ($_POST['submit'] == "OK") {
<input type="submit" name="submit" value="OK">
//Le formulaire a été transmis
//Récupération des paramètres passés
$prenom = $_POST['prenom'];
$nom = $_POST['nom'];
//affichage des paramètres
echo "<center>Bonjour $prenom $nom</center>";
} else
{
?> <!-- Affichage du formulaire de saisie -->
<form method="post" action="formulaire.php">
Nom : <input type="text" name="nom" size="12"><br>
Prénom : <input type="text" name="prenom" size="12">
</form>
<?
}
?>
```

Remarque : La définition de la destination du formulaire (action="formulaire.php") aurait pu ne pas être nominative mais exploiter une variable serveur PHP\_SELF puisque les paramètres sont envoyées à la page elle même. Ainsi \$\_SERVER['PHP\_SELF'] retournera naturellement formulaire.php. Cela donnera donc :

```
<form method="post" action="<? echo $_SERVER['PHP_SELF']; ?>" >
à la place de <form method="post" action="formulaire.php">
```

## 8.2-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

```
<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']; ... ?>
```

Exemple : Dans l'exemple suivant nous allons créer un fichier menu.php contenant un menu fait d'hyperliens. Chacun de ces hyperliens enverra des paramètres différents. Ce menu sera appelé dans une page qui réagira différemment selon le paramètre envoyé.

#### ***Fichier menu.php***

```
<table width="200" border="0" cellspacing="0" cellpadding="1">
<tr>
    <td> <a href="page.php?menu=1">Menu1</a> </td>
</tr>
<tr>
    <td> <a href="page.php?menu=2">Menu2</a> </td>
</tr>
<tr>
    <td> <a href="page.php?menu=3">Menu3</a> </td>
</tr>
</table>
```

#### ***Fichier page.php***

```
<? Include("menu.php") ;
echo "<br><br>" ;
$menu= $_GET['menu'] ;
switch ($menu) {
    case 1: echo "Ceci est la page obtenue du Menu1" ; break;
    case 2: echo "Ceci est la page obtenue du Menu2" ; break;
    case 3: echo "Ceci est la page obtenue du Menu3" ; break; default: echo "Ceci est la page
d'accueil" ; } ?>
```

En exécutant à travers le serveur web page.php, la sélection de chacun des menus chargera à nouveau la page en envoyant des paramètres différents qui seront traités par la page.

## **9. Les fichiers**

### **9.1- Ouverture / fermeture de fichier**

Avant toute opération de lecture ou écriture sur un fichier il y a nécessité de l'ouvrir. Et à la fin de tout traitement d'un fichier il y a nécessité de le fermer.

Syntaxe

```
<?
// Ouverture de fichier
"
$monfichier = fopen(nom_du_fichier", "r+");
// Traitements sur le fichier
// .....
// Fermeture du fichier fclose($monfichier);
?>
Fopen() prend en entrée :
```

-le nom du fichier (ou meme une url)

-le mode d'ouverture du fichier Il retourne un handle.

Fclose() prend en entrée le handle envoyé par le fopen.

Les modes d'ouverture d'un fichier sont les suivants :

- 'r' Ouvre en lecture seule, et place le pointeur de fichier au début du fichier.
- 'r+' Ouvre en lecture et écriture, et place le pointeur de fichier au début du fichier.
- 'w' Ouvre en écriture seule ; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. Si le fichier n'existe pas, on tente de le créer.
- 'w+' Ouvre en lecture et écriture ; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. Si le fichier n'existe pas, on tente de le créer.
- 'a' Ouvre en écriture seule ; place le pointeur de fichier à la fin du fichier. Si le fichier n'existe pas, on tente de le créer.
- 'a+' Ouvre en lecture et écriture ; place le pointeur de fichier à la fin du fichier. Si le fichier n'existe pas, on tente de le créer.

Exemple

<?

```
$handle = fopen("http://www.example.com/", "r");
```

```
$handle = fopen("test/monfichier.txt", "r+") t ;
```

?>

## 9.2- Lecture de fichier

### a- fgets()

string fgets ( resource handle ,int length )

fgets retourne la chaîne lue jusqu'à la longueur length - 1 octet depuis le pointeur de fichier handle , ou bien la fin du fichier, ou une nouvelle ligne (qui inclue la valeur retournée), ou encore un EOF (celui qui arrive en premier). Si aucune longueur n'est fournie, la longueur par défaut est de 1 ko ou 1024 octets.

Syntaxe

<?

```
$monfichier = fopen("nom_du_fichier", "r+");
```

```
fgets ($monfichier, $taille) ; fclose($monfichier);
```

?>

La fonction prend en paramètre :

- le handle retourné par la fonction fopen()

- la taille en octets de lecture (optionnel)

Elle retourne en sortie une chaîne de caractères.

Exemple :

L'exemple suivant va permettre de lire dans un fichier texte (fichier.txt) ligne par ligne puis d'afficher le résultat à l'écran en mettant un numéro à chaque ligne.

Fichier fichier.txt

Tarak Joulak

Guest0

abc123

Fichier page.php

```
<?
$i=0;
$fichier = 'fichier.txt';
$fp = fopen($fichier,'r') //ouverture du fichier en lecture seulement
while (feof($fp)) // tant qu'on n'est pas a la fin du fichier
{
    $ligne = fgets($fp); //Lecture ligne par ligne
    echo 'Ligne '$i++.'--->' . $ligne . '<br>';
}
fclose($fp);
?>
```

Le résultat obtenu sera :

Ligne 1 ---> Tarak Joulak  
Ligne 2 ---> Guest0  
Ligne 3 ---> abc123

### **b -fread()**

string fread ( resource handle , int length )

Une autre manière de faire de la lecture dans un fichier est d'utiliser fread()

fread lit jusqu'à length octets dans le fichier référencé par handle . La lecture s'arrête lorsque length octets ont été lus, ou que l'on a atteint la fin du fichier, ou qu'une erreur survient (le premier des trois).

Syntaxe

```
<?
$monfichier = fopen("nom_du_fichier", "r+"); fread ($monfichier, $taille) ;
fclose($monfichier);
?>
```

### **c- file()**

array file (string filename)

Encore une autre manière de lire dans un fichier est d'utiliser la fonction file() Identique à readfile, hormis le fait que file retourne le fichier dans un tableau. Chaque élément du tableau correspondant à une ligne du fichier, et les retour–chariots sont placés en fin de ligne.

Syntaxe

```
<?
file ($nom_de_fichier)
?>
```

```
<?
// Lit un fichier, et le place dans une chaîne
$filename = "fichier.txt";
$handle = fopen ($filename, "r");
$content = fread ($handle,filesize ($filename));
echo "-->". $content; fclose ($handle);
?>
```

Le résultat obtenu sera :

--> Tarak Joulak  
Guest0  
abc123

Rq : la fonction file() ne nécessite pas l'usage de open() et close().

Exemple Fichier page.php

```
<?php
$filename = "test/fichier.txt";
$fcontents = file($filename);
echo $fcontents[0]."<br>";
echo $fcontents[1]."<br>";
?>
```

Le résultat obtenu sera :

Tarak Joulak

Guest0 3

- Ecriture dans fichier La fonction pour l'écriture dans un fichier est fputs()

int fputs ( int handle , string string , int length )

fputs écrit le contenu de la chaîne string dans le fichier pointé par handle . Si la longueur length est fournie, l'écriture s'arrêtera après length octets, ou à la fin de la chaîne (le premier des deux).

fwrite retourne le nombre d'octets écrits ou FALSE en cas d'erreur.

#### **Syntaxe**

```
<? $monfichier = fopen("nom_du_fichier", "r+");
fputs ($monfichier, "Texte à écrire");
fclose ($monfichier);
?>
```

#### **Exemple**

```
<? $filename = "fichier.txt"; $monfichier = fopen ($filename, "a+");
$content="ceci est le texte a ecrire \r\n";
fputs($monfichier, $content);
fclose ($monfichier);
?>
```

Le résultat obtenu sera :

Le fichier fichier.txt aura une ligne supplémentaire contenant « ceci est le texte à écrire »

### **9.3- Fonctions diverses de traitement de fichier :**

-feof (\$handle) fonction qui retourne un booleen pour indiquer la fin du fichier parcouru. Elle prend en entrée le handle retourné par la fonction fopen().

-filesize (\$nom\_du\_fichier) fonction qui retourne la taille du fichier en octets.

-file\_exists (\$nom\_du\_fichier ) fonction qui retourne un booleen indiquant si le fichier existe ou non.

## **10 Variables persistantes: Cookies et Session**

Les variables ont une durée de vie limitée : celle du script qui les appelle. Ainsi que nous l'avons vu dans le chapitre précédent l'unique moyen de transmettre ces variables de pages en pages consiste à effectuer un passage de paramètres (méthode GET ou POST) ce qui d'une manière générale est contraignant à plus d'un titre :

- Contraignant pour le développeur puisque il doit gérer par code ces passages de paramètres,

- Contraignant pour la sécurité si l'on ne désire pas que le client accède à certaines informations. PHP offre un mécanisme de stockage d'informations de manière persistante. Autrement dit tout au long de la navigation à l'intérieur d'un site des variables seront accessibles sans avoir pour autant à les passer en paramètres. Deux types de variables persistantes existent :

- Les variables persistantes côté client : les cookies
- Les variables persistantes côté serveur : la session

### 10.1- les Cookies

Les cookies sont un mécanisme d'enregistrement d'informations sur le client, et de lecture de ces informations. Ce système permet d'authentifier et de suivre les visiteurs d'un site. PHP supporte les cookies de manière transparente.

#### a- setcookie() :

Cette fonction permet de définir un cookie qui sera envoyé avec le reste des en-têtes. int setcookie (string nom\_variable [,string valeur\_variable ],[int expiration ],[string chemin ],[string domaine ],[int sécurité ])

- nom\_variable : nom de la variable à stocker
- valeur\_variable : Valeur de la variable à stocker
- expiration : durée pour l'expiration du cookie
- chemin : le chemin du répertoire où doit être lu le cookie
- domaine : le nom domaine
- sécurité : le type d'entête (http, https)

Tous les arguments sauf nom\_variable sont optionnels. A l'instar de la fonction header(), setcookie() doit impérativement être appelée avant tout affichage de texte.

Syntaxe

```
<? setcookie("variable1",$valeur1, $durée,$chemin,$domaine,$securite); ?>
```

Exemples

```
<? $valeur= "Ceci est la valeur de la variable" ; setcookie ("TestCookie",$value,time()+3600); /*  
expire dans une heure */ ?> 31
```

#### b- lire un cookie

Syntaxe <? \$HTTP\_COOKIE\_VARS["\$nom\_de\_la\_variable"] ; ?>

Exemple <? echo \$HTTP\_COOKIE\_VARS["TestCookie"]; ?>

Le résultat obtenu sera : Ceci est la valeur de la variable

#### c- Tableau de variables dans un cookie

Il est possible d'envoyer un tableau de variables à stocker dans un cookie

Exemple

```
<? setcookie( "tcookie[trois]", "troisième cookie" );  
setcookie( "tcookie[deux]", "second cookie" );  
setcookie( "tcookie[un]", "premier cookie" ); }?>
```

Exemple de lecture d'un tableau stocké dans un cookie

```
<? //récupération du tableau cookie dans la variable $cookie  
$cookie=$HTTP_COOKIE_VARS["tcookie"];  
//Vérification si la variable est vide ou non
```



```
if ( isset( $cookie ) ) {  
    while( list( $name, $value ) = each( $cookie ) ) {  
        echo "$name == $value<br>\n";  
    }  
}  
?>
```

Le résultat obtenu sera :  
trois == troisième cookie  
deux == deuxième cookie  
un == premier cookie

### **d-Suppression d'un cookie**

Pour supprimer un cookie envoyez un cookie avec une variable sans valeur et un délai dépassé.

Exemple

```
<? Setcookie ("TestCookie","",time()-100); ?>
```

### **e- Limitation des cookies**

Le problème majeur du cookie c'est que le client a le pouvoir de le refuser (en configurant spécifiquement son browser). Votre application risque donc de ne pas pouvoir fonctionner. Il y a aussi des risques plus graves quant à la sécurité. L'usurpation d'identité, car ce fichier peut être recopié facilement sur un autre ordinateur et modifié puisque ce n'est qu'un fichier texte.

## **10.2- les sessions**

La gestion des sessions avec PHP est un moyen de sauver des informations entre deux accès. Cela permet notamment de construire des applications personnalisées, et d'accroître l'attrait de votre site.

Chaque visiteur qui accède à votre site se voit assigner un numéro d'identifiant, appelé plus loin "identifiant de session". Celui-ci est enregistré soit dans un cookie, chez le client, soit dans l'URL.

Les sessions vous permettront d'enregistrer des variables pour les préserver et les réutiliser tout au long de la visites de votre site. Lorsqu'un visiteur accède à votre site, PHP vérifiera si une session a déjà été ouverte. Si une telle session existe déjà, l'environnement précédent sera recréé. L'inconvénient précédemment évoqué concernant les cookies est dépassé dans la mesure où tout est stocké sur le serveur même.

### **a- session\_start()**

session\_start() permet de démarrer une session pour le client .

Cette commande doit figurer dans toutes les pages elle perpétue le transfert des variables de session au cours de la navigation dans le site.

Le compilateur PHP va alors créer dans le répertoire de sauvegarde des sessions, un fichier dont le nom commence par sess\_ et se termine par un identifiant généré de manière aléatoire.

L'identifiant de session peut être affiché par la commande session\_id(). Vous pouvez également gérer vous-même ce nom de session en utilisant session\_name() avant le démarrage de la session.

La durée de vie d'une session est paramétrée par session.cache\_expire

La session est perdue définitivement pour l'utilisateur lorsque :

1- Il n'a exécuté aucune action (POST ou GET) au delà de la durée de vie définie .

2- Il ferme son navigateur.

3- La commande session\_destroy est appelée.

#### **b- Ajouter des variables dans la session**

L'ajout de variable dans l'environnement de session se fait au travers d'une affectation classique. Toutefois la variable session définie doit être appelée via le tableau \$\_SESSION[]

Dans cet exemple une variable du nom de ville contenant la valeur Tunis a été enregistrée dans la session courante.

```
<?
```

```
session_start() ;
```

```
?>
```

```
<?
```

```
session_start() ;
```

```
$_SESSION['nom_variable'] = $valeur;
```

```
?>
```

```
<?
```

```
session_start() ;
```

```
$_SESSION['ville'] = "Tunis";
```

```
?>
```

#### **c- Lire une variable dans la session**

```
<?
```

```
session_start() ;
```

```
$variable = $_SESSION['nom_variable'] ;
```

```
?>
```

```
<?
```

```
session_start() ;
```

```
echo ' Le contenu de la variable VILLE de la session est : ' . $_SESSION['ville'] ;
```

```
?>
```

Le résultat obtenu sera :

Le contenu de la variable VILLE de la session est : Tunis

#### **d- Supprimer une variable de la session**

```
<?
```

```
session_start() ;
```

```
unset ($_SESSION['nom_de_la_variable']);
```

```
?>
```

```
<?
```

```
session_start() ;
```

```
unset ($_SESSION['ville']); I /
```

```
/Verification de la suppression
```

```
if ( isset ($_SESSION['ville'])) I
```

```
{  
$resultat = "La suppression a échoué .";  
}  
else  
{  
$resultat = "La ville a été effacée.";  
}  
echo $resultat;  
?>
```

Le résultat obtenu sera :  
La ville a été effacée.

### **e- Supprimer l'environnement de session**

Dans l'exemple précédent nous avons vu comment supprimer une variable de l'environnement de session.

Il reste toutefois possible de supprimer tout un environnement de session donné. Pour cela il suffit de vider le tableau global des sessions en le réinitialisant.

```
<?  
session_start() ;  
$_SESSION = array();  
?>
```

## **11. PHP et les bases de données**

Vous pouvez utiliser pratiquement toutes les bases de données avec votre application Web, si vous possédez les pilotes de base de données requis.

Si vous prévoyez de créer de petites applications peu onéreuses, vous pouvez utiliser une base de données fichier, créée par exemple sous Microsoft Access. Si vous prévoyez de créer des applications stratégiques robustes, vous pouvez utiliser une base de données serveur, créée par exemple avec Microsoft SQL Server, Oracle 9i ou MySQL.

Si votre base de données réside sur un système autre que votre serveur Web, assurez-vous qu'il existe une connexion rapide entre les deux systèmes pour un fonctionnement efficace et rapide de votre application Web. Nous utiliserons dans le cadre de ce cours le SGBD MySQL.

### **11.1- 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 avons 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.

### **11.2- Concepts fondamentaux : Bases, tables, champs, enregistrements.**

Une base de donnée correspond donc à un ensemble de données sauvegardées de manière structurée. La structuration de ses données est hiérarchisée.

Une base de données peut regrouper plusieurs tables.

Ainsi une base de données BIBLIOTHEQUE par exemple contiendra une table PERIODIQUE pour stocker les périodiques, une table LIVRE pour stocker les livres.

La table LIVRE regroupera plusieurs champs destinés à décrire et qualifier les livres.

Ainsi cette table intégrera un champs NISBN, un champs TITRE et un champs AUTEUR par exemple.

La consultation de ces données se fait au travers du langage SQL.

### **11.3- Administration de MYSQL**

EasyPHP installe un produit pour l'administration d'une base de données MYSQL appelé PHPMYADMIN. Ce produit est intégralement développé en PHP. Toutefois nous lui préférons MySQL-Front qui reste plus simple d'utilisation.

MYSQL-Front est une application qui permet d'administrer une base de données MYSQL à distance (Création de base, de tables, d'enregistrements, exécution de requêtes, gestion des comptes, ...)

### **11.4- SQL : petit récapitulatif du langage**

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

#### **a-Création d'une base**

CREATE DATABASE exercice

Cette requête SQL va permettre de créer une base vide du nom de exercice.

#### **b-Création d'une table**

```
CREATE TABLE `t_personne` (  
  `id` INT (6) UNSIGNED AUTO_INCREMENT, `Nom` VARCHAR (50),  
  `Prenom` VARCHAR (50), `Age` TINYINT (3) UNSIGNED,  
  UNIQUE(`id`), INDEX(`id`))
```

Cette requête va permettre de créer une table du nom de 't\_personne' contenant 4 champs :

id : un identifiant entier qui s'auto incrémentera à chaque nouvel ajout d'enregistrement

Nom : une chaîne de caractères d'au maximum 50 caractères

Prenom : une chaîne de caractères d'au maximum 50 caractères

Age : un entier

#### **c-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 champs Nom prendra pour valeur Joulak

Le champs Prenom prendra pour valeur Tarak

Le champs Age prendra pour valeur 32

Il est à noter que le champs id étant en mode auto incrémental il ne faut pas lui assigner de valeur.

Le compteur s'incrémentant tout seul à chaque nouvel enregistrement.

#### **d-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)

#### **e-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.

#### **f-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).

### **11.5- 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.

#### **a- Connection à un serveur MYSQL**

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

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

#### **b- 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.

```
<?
```

```
mysql_connect("$nom_serveur_MYSQL", "$utilisateur", "$mot_de_passe");
```

```
?>
```

```
<?
```

```
$nom_serveur_MYSQL="localhost" ;
```

```
$utilisateur="root" ;
```

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

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

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

### **c- 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.

Dans l'exemple précédent une ligne d'enregistrement a été ajoutée à la table t\_personne contenant Tarak Joulak 32.

En remplaçant la requête de l'exemple précédent par les requêtes SQL de modification et suppression vues dans le paragraphe précédent on obtient les résultats déjà observés en exécutant directement la requête SQL.

### **d- 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\_row() 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.

<?

...

```
$res = mysql_query("$requete_sql");
```

?>

<?

```
$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");
```

?>

<?

...

```
$data = mysql_fetch_row($resultat_de_mysql_query) ;
```

?>

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

## 11.6 Extraction des données

---

**`mysql_fetch_row($result)`** : retourne une ligne de résultat sous la forme d'un tableau. Les éléments du tableau étant les valeurs des attributs de la ligne.  
Retourne FALSE s'il n'y a plus aucune ligne.

*Exemple 1 :*

```
$requet = "SELECT * FROM users";
```

```
if($result = mysql_query($requet)) {
```

```
    while($ligne = mysql_fetch_row($result)) {
```

```
        $id = $ligne[0];
```

```
        $name = $ligne[1];
```

```
        $address = $ligne[2];
```

```
        echo "$id - $name, $address <br />";
```

```
    }
```

```
} else {
```

```
    echo "Erreur de requête de base de données.";
```

```
}
```

Ici, on accède aux valeurs de la ligne par leur indice dans le tableau.

# Chapitre 8 Etude d'un éditeur HTML : Dreamweaver

## 1. Introduction

Ce guide a pour objectif de vous faire découvrir les principales fonctions de Macromedia Dreamweaver MX et leur utilisation. Les leçons de ce guide expliquent comment créer un site Web simple et fonctionnel. De nombreux éditeurs HTML existent, nous pouvons citer : WebEdit, Namo Web Editor, etc.

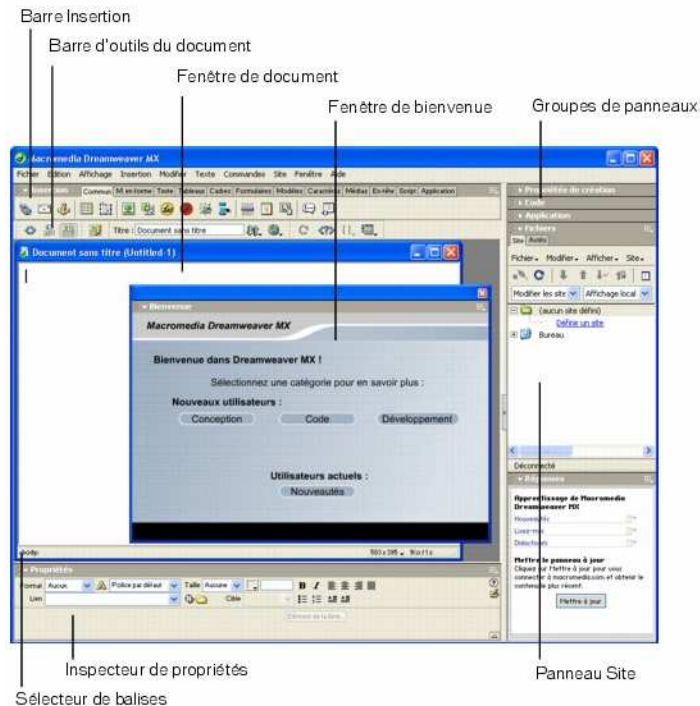
Macromedia Dreamweaver MX est un éditeur HTML professionnel destiné à la conception, au codage et au développement de sites, de pages et d'applications Web. Quel que soit l'environnement de travail utilisé (codage manuel HTML ou environnement d'édition visuel), Dreamweaver propose des outils qui vous aideront à créer des applications Web.

Les fonctions d'édition visuelles de Dreamweaver vous permettent de créer rapidement des pages sans rédiger une seule ligne de code. Si vous préférez faire appel au codage manuel, Dreamweaver intègre également de nombreux outils et fonctions de codage. Avec Dreamweaver, vous pouvez créer des applications Web dynamiques reposant sur des bases de données à l'aide de langages serveur tels que ASP, ASP.NET, ColdFusion Markup Language (CFML), JSP et PHP.

## 2. Présentation des fenêtres et des panneaux

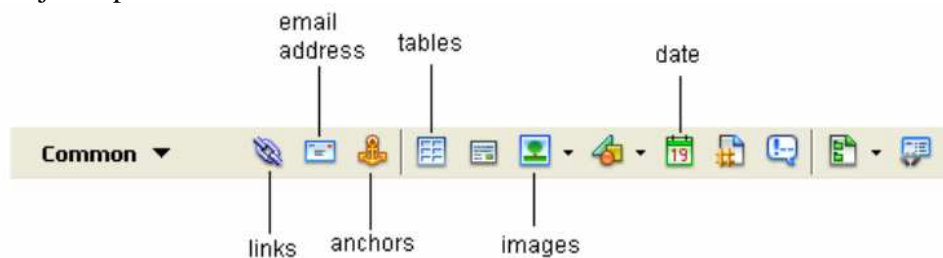
La figure ci-dessous donne quelques descriptions succinctes des fenêtres et autres éléments figurant dans l'espace de travail de Dreamweaver. Des descriptions plus détaillées relatives à l'utilisation de ces fenêtres sont fournies plus loin d





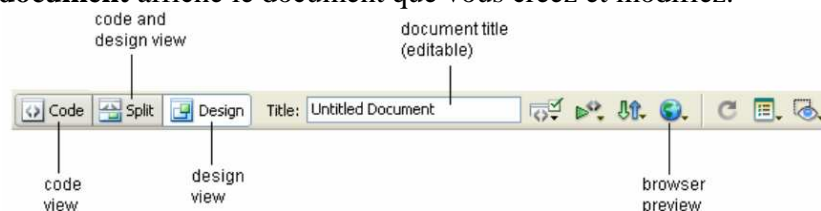
**La fenêtre Bienvenue** fournit des conseils de configuration de l'espace de travail selon les besoins, ainsi que des informations sur les nouvelles fonctionnalités de Dreamweaver à l'attention des utilisateurs des versions précédentes du logiciel.

**La barre Insertion** contient des boutons permettant d'insérer divers types d'« objets », tels que des images, tableaux et calques dans un document. Chaque objet est une portion de code HTML vous permettant de définir des attributs lors de son insertion. Par exemple, vous pouvez insérer une image en cliquant sur l'icône Image dans la barre Insertion. Si vous le préférez, vous pouvez insérer les objets à partir du menu Insertion.

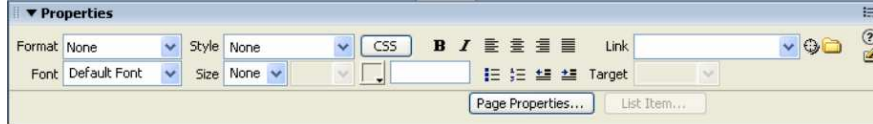


**La barre d'outils du document** contient des boutons et des menus déroulants permettant d'accéder aux différents modes d'affichage de la fenêtre du document (tels que le mode Création ou le mode Code), de définir les différentes options d'affichage et d'effectuer certaines opérations courantes, telles que la prévisualisation dans un navigateur.

**La fenêtre de document** affiche le document que vous créez et modifiez.



**L'inspecteur de propriétés** permet de visualiser et de modifier diverses propriétés de l'objet ou du texte sélectionné. Chaque objet contient des propriétés différentes.



**Les groupes de panneaux** sont des ensembles de panneaux connexes ancrés ensemble sous un même en-tête. Pour développer un groupe de panneaux, cliquez sur la flèche d'agrandissement située à gauche du nom du groupe ; pour détacher un groupe de panneaux, faites glisser la poignée d'ancrage sur le côté gauche de la barre de titre du groupe.

**Le panneau Site** permet de gérer les fichiers et les dossiers composant votre site. Pour plus d'informations, voir Définition d'un site local. Il permet également d'afficher tous les fichiers situés sur votre disque local, tout comme l'Explorateur Windows (Windows) ou le Finder (Macintosh).

Dreamweaver propose de nombreux autres inspecteurs, panneaux et fenêtres qui ne sont pas illustrés ici, tels que le panneau Historique et l'inspecteur de code. Pour ouvrir les panneaux, inspecteurs et fenêtres Dreamweaver, utilisez le menu Fenêtre.

### 3. Présentation des menus

Cette section présente les menus de Dreamweaver.

**Le menu Fichier et le menu Edition** contiennent des éléments de menu standard, tels que Nouveau, Ouvrir, Enregistrer, Couper, Copier et Coller. Le menu Fichier contient également plusieurs autres commandes permettant d'afficher un aperçu du document dans un navigateur ou d'imprimer du code, par exemple. Le menu Edition inclut des commandes de sélection et de recherche, telles que Sélectionner balise parente, Rechercher et Remplacer, et permet d'ouvrir l'Editeur de raccourcis clavier ainsi que l'Editeur de la bibliothèque de balises. Vous pouvez également l'utiliser pour accéder aux Préférences, sauf si vous disposez d'un ordinateur Macintosh fonctionnant sous Mac OS X dans lequel les préférences se trouvent dans le menu Dreamweaver.

**Le menu Affichage** permet de modifier l'affichage du document (mode Création ou mode Code, par exemple) et d'afficher ou de masquer plusieurs types d'éléments de page et d'outils Dreamweaver.

**Le menu Insertion et la barre Insertion** permettent d'insérer des objets dans votre document.

**Le menu Modifier** permet de modifier les propriétés de l'élément de page sélectionné. Vous pouvez l'ouvrir pour modifier les attributs de balises, des tableaux et leurs éléments et effectuer diverses opérations avec les éléments de bibliothèque et les modèles.

**Le menu Texte** permet de formater facilement le texte.

**Le menu Commandes** propose une commande de formatage du code qui tient compte de vos préférences de formatage, une commande de création d'album photos et une commande d'optimisation des images dans Macromedia Fireworks.

**Le menu Site** contient des éléments de menu permettant de créer, d'ouvrir et de modifier des sites, ainsi que de gérer les fichiers du site courant si vous utilisez un ordinateur Macintosh.

**Le menu Fenêtre** permet d'accéder à tous les panneaux, inspecteurs et fenêtres de Dreamweaver.

**Le menu Aide** permet d'accéder à la documentation de Dreamweaver, aux systèmes d'aide relatifs à l'utilisation de Dreamweaver et à la création d'extensions vers Dreamweaver, ainsi qu'à des références en plusieurs langues.

Outre les menus de la barre de menus, Dreamweaver propose plusieurs menus contextuels qui permettent d'accéder rapidement à des commandes utiles en rapport avec la zone ou la sélection courante. Pour afficher un menu contextuel, cliquez avec le bouton droit de la souris (Windows) ou en maintenant la touche Contrôle enfoncée (Macintosh) sur un élément qui vous intéresse dans une fenêtre. Tous les éléments des menus contextuels figurent également dans la barre de menus.

#### **4. Affichage d'un exemple de site**

Les exemples utilisés dans ce guide sont extraits d'un petit exemple de site développé pour une société fictive appelée « Global Car Rentals », affichez l'exemple de site dans un navigateur pour vous faire une idée de l'objectif à atteindre.

Pour afficher l'exemple de site dans un navigateur :

Ouvrez le dossier Samples dans le dossier de l'application Dreamweaver. Ouvrez le dossier GettingStarted, puis le dossier FinalSite.

Double-cliquez sur le fichier index.htm dans le dossier FinalSite pour afficher le site dans un navigateur.

#### **5. Définition d'un site local**

En général, les personnes créant un site Web à l'aide de Dreamweaver, créent et modifient des pages sur leur disque local et en téléchargent un exemplaire sur un serveur Web afin de les mettre à la disposition d'autres utilisateurs. Il est possible d'utiliser Dreamweaver d'autres manières (en exécutant un serveur Web sur votre ordinateur local, en téléchargeant des fichiers vers un serveur intermédiaire ou en utilisant un disque monté comme s'il s'agissait de votre disque local), mais les leçons proposées dans ce guide supposent que vous utilisez un ordinateur local et que vous téléchargez vos pages vers un serveur distant.

Dans Dreamweaver, le mot site fait référence aux éléments suivants :

Un site Web : ensemble de pages sur un serveur pouvant être visualisées par tout visiteur du site disposant d'un navigateur Web.

Un site distant : fichiers stockés sur un serveur qui constituent un site Web, du point de vue de l'auteur (vous) plutôt que de celui du visiteur.

Un site local : fichiers stockés sur votre disque local correspondant à ceux que vous avez téléchargés sur le site distant. Les fichiers sont modifiés sur votre disque dur, puis téléchargés vers le site distant.

Une définition de site Dreamweaver : ensemble de définitions d'un site local, plus des informations sur la façon dont le site local correspond à un site distant.

Généralement, la création d'un site Web est précédée d'une étape de planification qui détermine les éléments suivants : le nombre de pages, le contenu de chaque page et les liens associant les pages entre elles. Dans cette leçon, le site que vous allez créer est très simple et la planification sera brève : il comportera deux pages Web reliées entre elles. Dans ce cas, vous pouvez passer l'étape de planification et commencer à créer une définition de site.

La boîte de dialogue Définition du site vous permet de créer une définition de site. Vous pouvez compléter les champs de cette boîte de dialogue dans l'un des deux affichages suivants : Élémentaire ou Avancé. L'onglet Élémentaire présente toutes les étapes de définition d'un site. Si

vous préférez modifier les informations relatives au site sans aide, cliquez sur l'onglet Avancé à tout moment.

La procédure suivante explique comment définir des options dans la version Elémentaire de la boîte de dialogue ; elle est également appelée Assistant de définition d'un site. Pour plus de détails sur la définition d'options dans la version Avancé, cliquez sur l'onglet Avancé, puis sur le bouton Aide.

Pour définir un site :

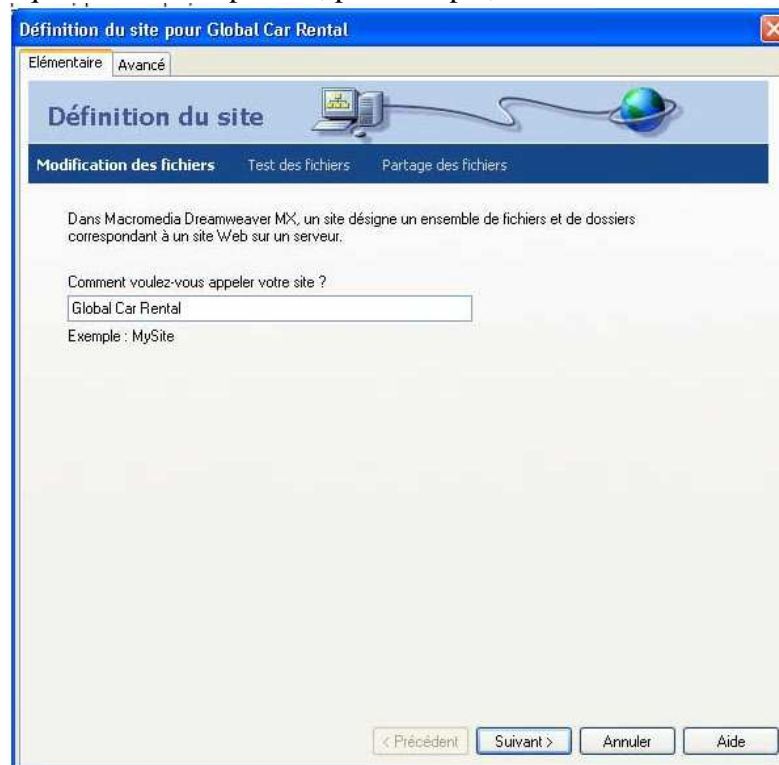
Choisissez Site > Nouveau site. (Choisissez Nouveau site dans le menu Site.)

La boîte de dialogue Définition du site s'ouvre.

Si elle affiche l'onglet Avancé, cliquez sur Elémentaire.

Le premier écran de l'Assistant de définition d'un site apparaît et vous demande d'attribuer un nom au site.

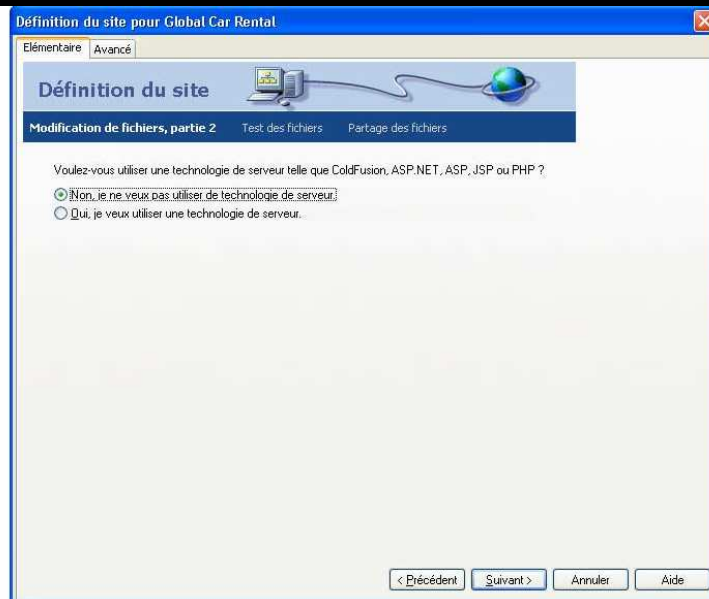
Dans la zone de texte, tapez un nom permettant d'identifier le site dans Dreamweaver. Il peut s'agir de n'importe quel nom. Vous pouvez, par exemple, nommer le site Global Car Rental.



Cliquez sur Suivant pour passer à l'étape suivante.

L'écran suivant de l'Assistant apparaît et vous demande si vous souhaitez utiliser une technologie de serveur.

Choisissez l'option Non pour indiquer que, pour l'instant, le site est statique et qu'il ne contient aucune page dynamique.



Pour définir un site dans le but de créer une application Web, vous devez choisir un type de document dynamique, comme Macromedia ColdFusion, Microsoft Active Server Pages (ASP), Microsoft ASP.NET, Sun JavaServer Pages (JSP), ou PHP: Hypertext Preprocessor (PHP), puis indiquer les informations concernant votre serveur d'application (pour plus d'informations, voir Création d'applications Web dans Dreamweaver MX).

Cliquez sur Suivant pour passer à l'étape suivante.

L'écran suivant de l'Assistant s'affiche et vous demande comment vous souhaitez travailler avec vos fichiers.

Sélectionnez l'option Modifier les copies locales sur ma machine, puis télécharger vers le serveur lorsque je suis prêt (recommandé).

Vous pouvez utiliser les fichiers de diverses façons pendant le développement du site, mais dans le cadre de cette leçon, choisissez cette option.

La zone de texte vous permet d'indiquer un dossier de votre disque dur dans lequel Dreamweaver doit stocker la version locale des fichiers du site. Pour indiquer un nom de dossier exact, il est préférable de parcourir l'arborescence pour trouver ce dernier que d'en indiquer le chemin d'accès dans la zone appropriée. Cliquez sur l'icône représentant un dossier située près de la zone de texte.

La boîte de dialogue Choisissez le dossier racine local pour le site Global Car Rental apparaît.

Dans cette boîte de dialogue, choisissez un dossier de votre disque local dans lequel vous voulez stocker tous vos sites. Ne cliquez pas encore sur OK.

Remarque : ce dossier doit contenir tous vos sites ; il est donc préférable de ne pas choisir le dossier racine de votre disque local. Vous allez créer ultérieurement pour ce site un dossier racine sur votre disque local dans le dossier des sites.

Si vous ne disposez d'aucun dossier de sites, créez-en un maintenant (en utilisant le bouton de création de dossier dans la boîte de dialogue Choisissez le dossier racine local pour le site Global Car Rental). Nommez le dossier Sites. L'emplacement le mieux approprié au dossier des sites dépend de votre système d'exploitation :

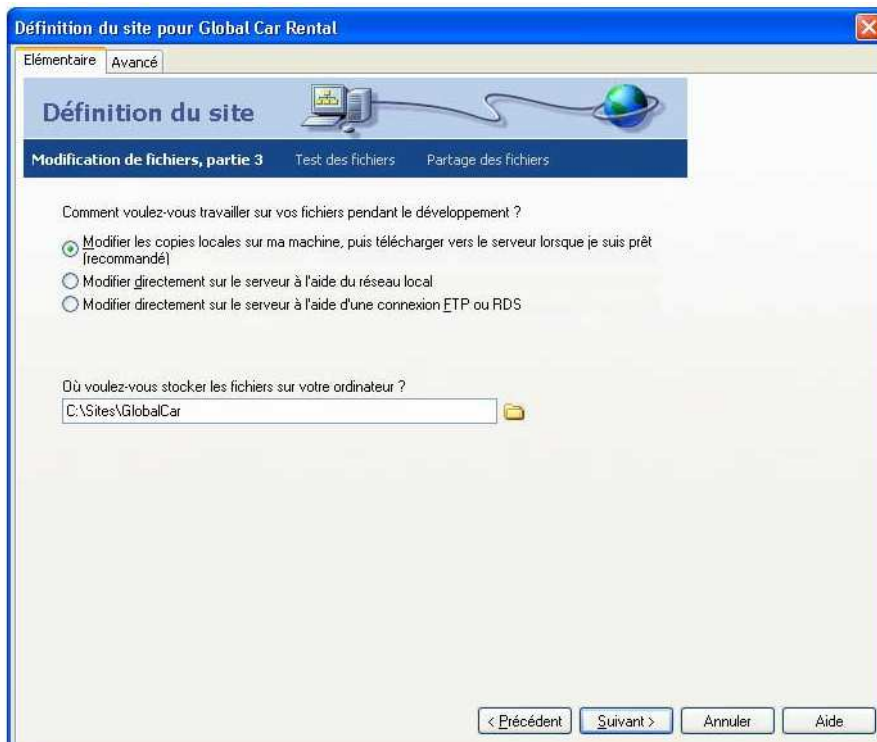
Sous Windows, si vous ne disposez d'aucun répertoire de stockage des sites, créez un dossier au niveau supérieur de l'arborescence de votre disque C et nommez le dossier Sites. Le chemin d'accès au dossier est donc le suivant : C:\Sites.

Sous Mac OS 9, si vous ne disposez d'aucun répertoire de stockage des sites, créez un dossier au niveau supérieur de votre disque et nommez-le Sites.

Sous Mac OS X, votre dossier de base (/Users/your\_user\_name) contient un dossier appelé Documents. Naviguez vers ce dossier, puis créez à l'intérieur un dossier appelé Sites.

Dans la boîte de dialogue Choisissez le dossier racine local pour le site Global Car Rental, créez un dossier dans le dossier Sites. Nommez le nouveau dossier GettingStarted et cliquez sur OK pour fermer la boîte de dialogue Choisissez le dossier racine local pour le site Global Car Rental.

Ce nouveau dossier est le dossier racine local de votre site.



Cliquez sur Suivant pour passer à l'étape suivante.

L'écran de l'Assistant apparaît et vous demande de préciser le type de connexion au serveur distant.

Pour l'instant, choisissez Aucun dans le menu déroulant. Cliquez sur Suivant pour passer à l'étape suivante.

L'écran suivant apparaît, affichant un résumé de vos paramètres.

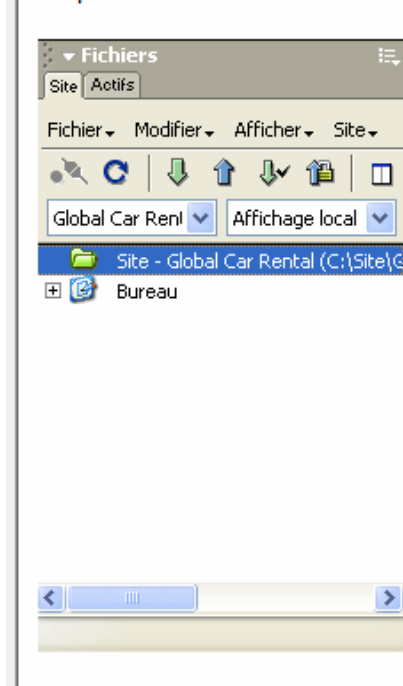
Cliquez sur Terminé.

Vous pouvez définir ultérieurement des informations concernant votre site distant (voir Configuration d'un site distant et publication) ; pour le moment, les informations concernant le site local suffisent pour créer une page.

Un message vous informe que Dreamweaver va créer un cache de site. Ce cache permet à Dreamweaver de stocker des informations concernant le site pour accélérer certaines opérations devant être effectuées sur ce dernier.

Cliquez sur OK pour permettre à Dreamweaver de créer le cache de site.

Le panneau Site affiche maintenant le nouveau dossier racine local de votre site actuel et une icône vous permet d'afficher tous vos disques locaux dans une arborescence hiérarchique. L'icône est intitulée Bureau (Windows) ou Ordinateur (Macintosh).



Le panneau Site affiche normalement tous les fichiers et dossiers de votre site, mais pour l'instant, votre site ne contient aucun fichier ni dossier. Lorsque votre site contiendra des fichiers, la liste de fichiers du panneau Site tiendra lieu de gestionnaire de fichiers et vous permettra de copier, de coller, de supprimer, de déplacer et d'ouvrir des fichiers tout comme le feriez sur le Bureau de votre ordinateur.

Si disposez déjà d'un ensemble de fichiers HTML locaux que vous souhaitez utiliser pour créer un site Web, vous pouvez utiliser le navigateur de fichiers du panneau Site pour les copier dans le dossier du site que vous venez de créer. Toutefois, vous voudrez peut-être suivre toutes les leçons de ce guide en utilisant les fichiers fournis avec Dreamweaver avant d'utiliser vos propres fichiers.

## 6. Création et enregistrement d'une nouvelle page

Une fois votre site défini, vous pouvez créer des pages Web pour le remplir.

Lorsque vous avez démarré Dreamweaver, un document HTML vierge a été automatiquement généré. Avant de poursuivre, fermez ce document.

Pour fermer le document vierge par défaut :

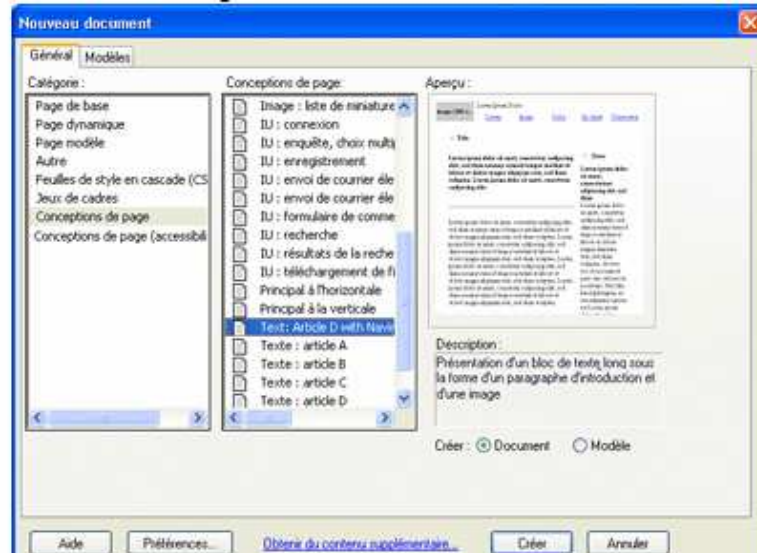
Choisissez Fichier > Fermer.

Pour créer une page :



1. Choisissez Fichier > Nouveau.

La boîte de dialogue Nouveau document s'affiche.



2. Dans la liste Catégorie se trouvant dans la partie gauche de la fenêtre, sélectionnez Conceptions de page.

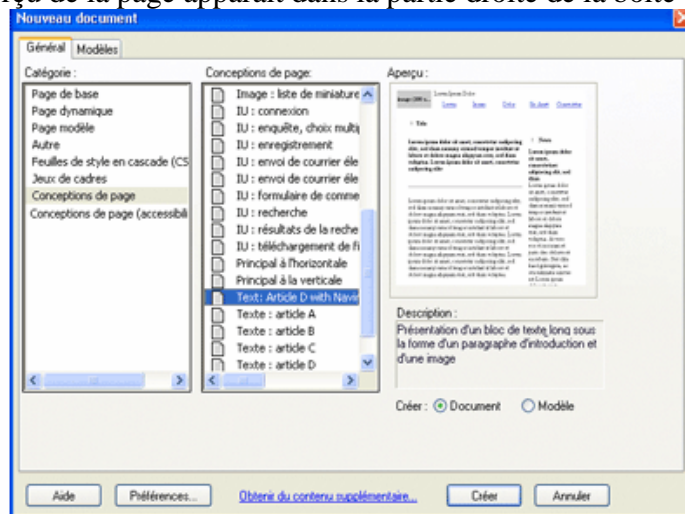
La liste se trouvant au milieu de la boîte de dialogue est renommée Conceptions de page. Une liste de pages préalablement conçues apparaît.

3. Faites défiler la liste Conceptions de page et choisissez l'élément appelé Text: Article D with Navigation.

Remarque : il existe un autre élément portant un nom semblable. Veillez à ne pas sélectionner l'élément appelé Text: Article D qui ne comporte aucune barre de navigation.



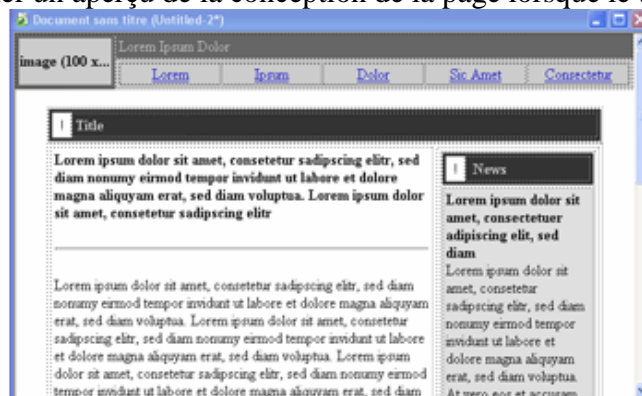
Un petit aperçu de la page apparaît dans la partie droite de la boîte de dialogue.



Si vous préférez, vous pouvez créer une page à l'aide d'une des autres conceptions de pages fournies ou créer une page sans conception prédéfinie en choisissant un élément dans la catégorie Page de base. Le reste de la leçon suppose que vous utilisez la conception de page Text: Article D with Navigation.

4. Veillez à ce que le bouton radio Document soit sélectionné dans la partie inférieure droite de la boîte de dialogue.
5. Cliquez sur Créer.

La nouvelle page qui apparaît affiche la mise en forme que vous avez choisie dans une nouvelle fenêtre de document. La page contient le texte « Lorem ipsum » d'indication de position pour donner un aperçu de la conception de la page lorsque le texte y sera inséré.



6. Enregistrez votre document.

Pour enregistrer votre page :

1. Choisissez Fichier > Enregistrer.
2. Dans la boîte de dialogue Enregistrer sous, recherchez le dossier Design figurant dans le dossier racine du site.

Rappel : le dossier racine du site est le dossier que vous avez créé lors de la définition du site dans Définition d'un site local.

3. Saisissez le nom de fichier index.htm.
4. Cliquez sur Enregistrer.

Le nom de fichier apparaît désormais dans la barre de titre de la fenêtre de document, entre parenthèses après les mots « Document sans titre ».

## 7. Mise en forme avec Dreamweaver (TP)

- Mise en forme de textes
- Mise en forme pages
- Ajout d'une image
- Création tableau
- Création frame
- Lien dans dreamweaver
- Etc.

## 8. Ajout de styles au texte

Plusieurs méthodes vous permettent d'ajouter des styles à un texte dans un document HTML. Vous pouvez utiliser des feuilles de style en cascade (CSS) pour définir des balises HTML formatées d'une certaine façon.

Cette leçon explique comment créer une feuille de style CSS simple à partir d'une feuille de style prédéfinie, puis l'appliquer au texte, et modifier les styles.

### Pour créer une feuille de style CSS :

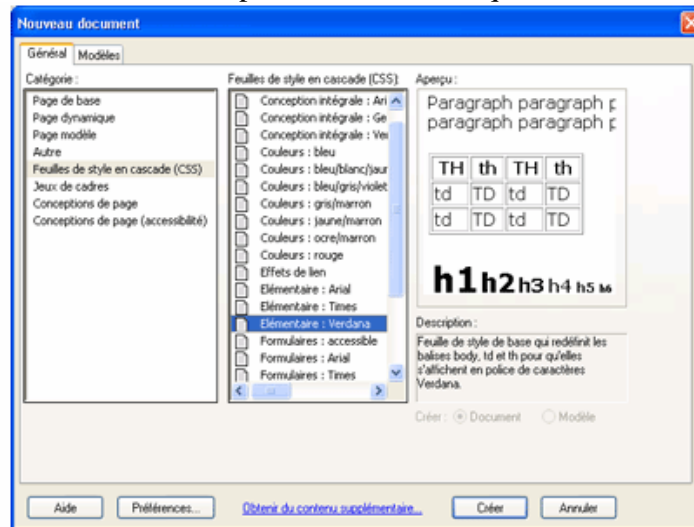
Choisissez **Fichier > Nouveau**.

Dans la boîte de dialogue Nouveau document, sélectionnez la catégorie Feuilles de style en cascade (CSS) dans la liste des catégories à gauche.

La liste se trouvant au milieu de la boîte de dialogue est renommée Feuilles de style en cascade (CSS). Une liste de feuilles de style prédéfinies apparaît.

Dans la liste Feuilles de style CSS, sélectionnez une feuille de style. Pour le site Global Car Rental, sélectionnez **Elémentaire : Verdana**, car ce style redéfinit les balises body, td et th en

leur attribuant des polices. Ensuite, cliquez sur **Créer**.



Dreamweaver crée un fichier texte contenant un ensemble limité de styles CSS prédéfinis.

Choisissez **Fichier > Enregistrer** pour enregistrer le nouveau fichier CSS. Enregistrez-le dans le dossier Assets du site sous le nom de text.css (ou tout autre nom de votre choix).

Choisissez **Fichier > Fermer** pour fermer le fichier CSS.

### **Pour appliquer à votre texte les styles de la feuille de style CSS :**

Dans le menu Fenêtre, choisissez un fichier HTML (comme index.htm).

**Remarque :** si vous avez choisi de ne pas afficher les extensions des fichiers, le fichier index.htm apparaît comme index.

Choisissez **Fenêtre > Styles CSS** pour afficher le panneau Styles CSS.

En haut du panneau Styles CSS, cliquez sur le bouton radio **Modifier les styles** pour afficher les styles disponibles.

Aucun style n'est disponible si vous n'en avez jamais défini pour ce document.

Dans la partie inférieure du panneau Styles CSS, cliquez sur le bouton **Attacher une feuille de style**.

La boîte de dialogue Ajouter une feuille de style externe apparaît.

Dans cette boîte de dialogue, cliquez sur **Parcourir** pour rechercher une feuille de style.

Dans la boîte de dialogue Sélectionner le fichier feuille de style, sélectionnez dans le dossier Assets la feuille de style que vous venez de créer et cliquez sur **OK** (Windows) ou **Choisir** (Macintosh) pour l'attacher.

Dans la boîte de dialogue Ajouter une feuille de style externe, cliquez sur **OK** pour attacher la feuille de style.

Le nom et le contenu de la feuille de style apparaissent dans le panneau Styles CSS. Les styles définis dans la feuille de style sont appliqués au texte du document HTML. Par exemple, le corps du texte apparaît en Verdana.

Enregistrez votre document.

### **Pour modifier les styles de la feuille de style :**

En haut du panneau Styles CSS, cliquez sur le bouton radio **Modifier les styles** pour afficher les styles disponibles.

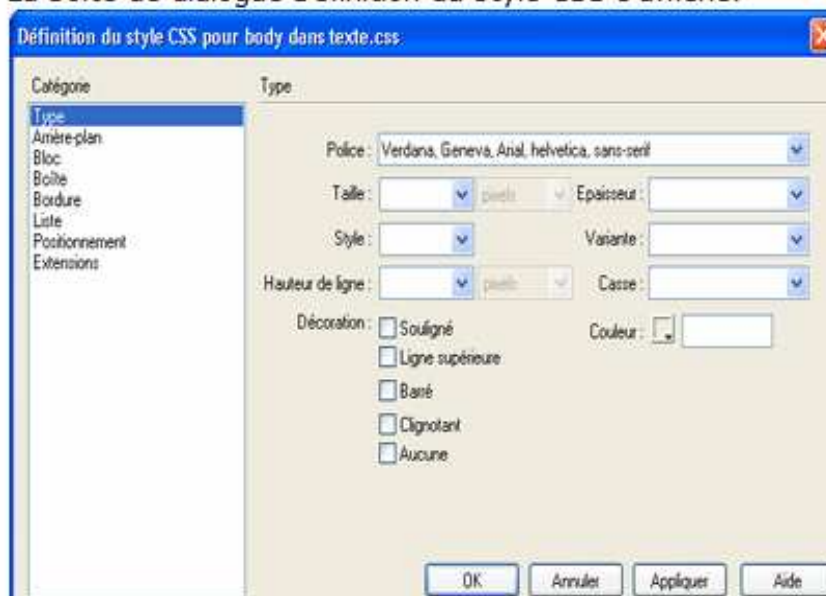
Sélectionnez le nom du fichier CSS dans le panneau Styles CSS et cliquez sur le bouton **Modifier une feuille de style** dans la partie inférieure du panneau Styles CSS.



La boîte de dialogue qui apparaît affiche le nom des styles de la feuille de style.

Sélectionnez un des styles, comme `body`, et cliquez sur **Modifier**.

La boîte de dialogue Définition du style CSS s'affiche.



Attribuez une taille au texte, comme 13 pixels. Modifiez les autres options comme vous le souhaitez.

Cliquez sur **OK** pour redéfinir le style.

Modifiez les autres styles. Pour créer les styles utilisés dans le site Global Car Rental, définissez les styles `body`, `td` et `th` sur une taille de 13 pixels.

Une fois l'édition des styles terminée, cliquez sur le bouton **Enregistrer** pour enregistrer vos modifications et fermer la boîte de dialogue de la feuille de style.

Les styles modifiés sont appliqués à votre document. Par exemple, le corps du texte apparaît en Verdana, 13 pixels.

# TD ET TP

## Partie HTML

### Exercice 1

L'étudiant doit coder, à l'aide d'un éditeur de texte, un document .htm ou .html nommé mon\_document (sans espace, accents ou caractères spéciaux).

La page doit contenir les éléments suivants :

- un titre ;
- deux niveaux d'en-tête ;
- deux paragraphes au minimum ;
- du texte en caractère gras et italique ;
- un bloc en retrait (citation) ;
- un hyperlien absolu (vers un site externe) avec titre et devant s'ouvrir dans une nouvelle fenêtre.
- une liste ordonnée (numérotée) ;
- une liste non ordonnée ;
- une table de trois rangées et quatre colonnes ;
- une image (avec spécification des dimensions et texte descriptif). L'image choisie doit être sauvegardée au préalable dans un répertoire « img » ou « images » créé à cette fin.

NOTE : Il n'est pas nécessaire que le texte des paragraphes, listes et tables soit pertinent. Il est possible de copier-coller !!!

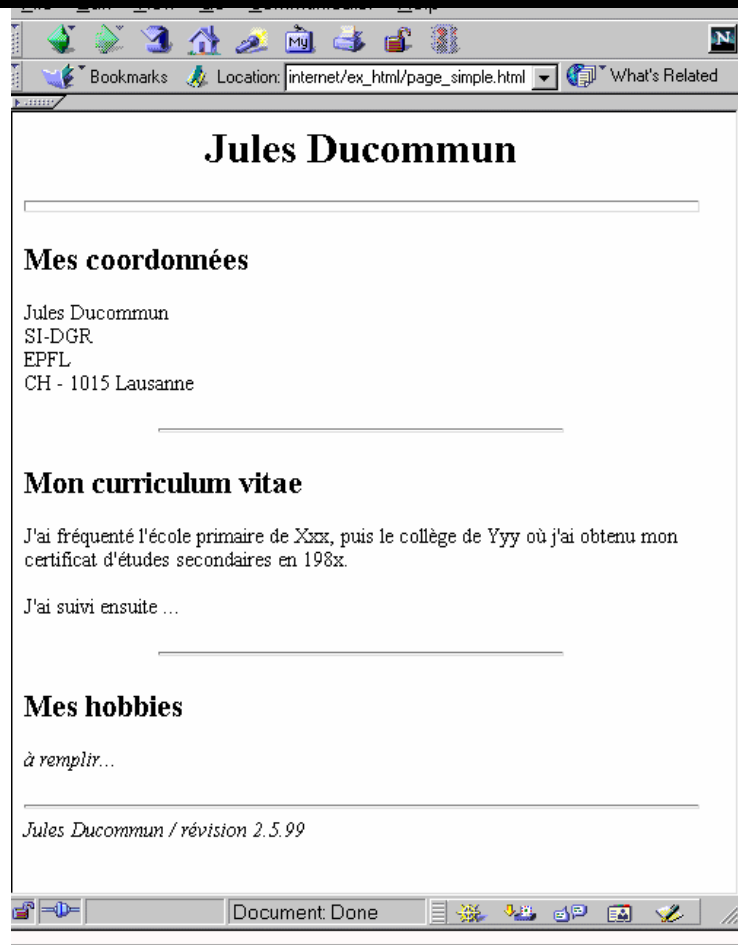
NOTE : Le fichier .html et les dossiers liés devront être conservés pour la suite (exercice avec les feuilles de style).

### Exercice 2

Réaliser la page Web simple illustrée par la copie-écran ci-dessous. Utilisez pour ce faire les 2 outils suivants :

- ouvrez le Bloc-notes (Notepad) Windows dans la moitié de gauche de l'écran
- lancez le navigateur dans la moitié de droite de l'écran

Après avoir créé quelques lignes de votre page HTML, sauvegardez-la et ouvrez-la dans Un navigateur pour la vérifier. Au fur et à mesure de l'avancement de votre travail, rechargez-la dans le navigateur (bouton [Reload this page...] ou View>Reload) pour la contrôler.



## Partie JAVASCRIPT

### Exercice 1

Construire un document HTML qui permet de lire deux nombres A et B dans les zones de texte, de choisir une opération dans une liste déroulante (+, -, \*, /, %) évalue et affiche dans une zone de texte C le résultat trouvé.

### Exercice 2

Modifier le document ci-dessus de façon à obtenir un document qui permet de lire les coefficients d'une équation du second degré et calculer ses solutions.

### Exercice 3

Transformer une durée exprimée en secondes en heures, minutes, secondes.

### Exercice 4

Calculer l'hypoténuse d'un triangle de côtés A et B. (SQRT = racine carré).

### **Exercice 5**

Calculer la tangente de A en n'utilisant que les fonctions sin & cos.

### **Exercice 6**

On donne une somme en €. On demande de déterminer le nombre de billets de 500, 200, 100, 50, 20, 10 et de 5 € et le nombre de pièces de 2, 1, 0.50, 0.20, 0.10, 0.05, 0.02, 0.01 € dont il faut disposer pour reconstituer cette somme.

### **Exercice 7**

On donne deux nombres x et y. On demande d'écrire un programme qui range le plus petit dans x et le plus grand dans y.

### **Exercice 8**

Un vendeur de voitures peut appliquer 2 taux de T.V.A. différents : si la puissance de la voiture est strictement inférieure à 115 kW, le taux est de 25%; si elle est supérieure, il est de 33%.

Ecrire un programme qui demandera le prix de base du véhicule et sa puissance, et qui donnera le taux de T.V.A. applicable, le montant de la T.V.A. et le prix total.

### **Exercice 9**

Ecrire un programme qui range trois nombres donnés x, y, z, dans l'ordre croissant ( $x < y < z$ ).

### **Exercice 10**

On donne trois nombres a, b, c, rangés dans l'ordre croissant et un quatrième nombre x. Ecrire un programme qui affiche les trois plus petites valeurs.

### **Exercice 11**

Ecrire un programme qui calcule le salaire mensuel net d'un individu connaissant son salaire horaire brut, le nombre d'heures prestées et la retenue de la sécurité sociale qui est de 22 % avec un plafond de 55 000 F.

### **Exercice 12**

Le tarif d'une compagnie de distribution d'eau est le suivant :

- redevance forfaitaire annuelle : 900 F.

Cette redevance donne droit à une consommation de 30 m<sup>3</sup> par an.

- les consommations supplémentaires sont facturées :

  - 23 F / m<sup>3</sup> du 31<sup>ème</sup> au 1000<sup>ème</sup> m<sup>3</sup>.

  - 20 F / m<sup>3</sup> du 1001<sup>ème</sup> au 5000<sup>ème</sup> m<sup>3</sup>.

  - 16 F / m<sup>3</sup> du 5001<sup>ème</sup> au 50000<sup>ème</sup> m<sup>3</sup>.

  - 12 F / m<sup>3</sup> au-delà de 50000 m<sup>3</sup>.

Ecrire un programme qui fournit les factures des clients en tenant compte d'une T.V.A. de 19 %.

### **Exercice 13**

Ecrivez un programme qui lit N nombres entiers au clavier et qui affiche leur somme, leur produit et leur moyenne. Choisissez un type approprié pour les valeurs à afficher. Le nombre N est à entrer au clavier. Résolvez

- a) en utilisant while ,
- b) en utilisant do - while
- c) en utilisant for

Laquelle des trois variantes est la plus naturelle pour ce problème ?

#### **Exercice 14**

Complétez la 'meilleure' des trois versions de l'exercice précédent :

Répétez l'introduction du nombre N jusqu'à ce que N ait une valeur entre 1 et 15.

Quelle structure répétitive utilisez-vous ? Pourquoi ?

#### **Exercice 15**

Calculez par soustraction successives le quotient entier et le reste de la division entière de deux entiers entrés au clavier.

#### **Exercice 16**

Calculez la factorielle  $N! = 1.2.3... (N-1).N$  d'un entier naturel N en respectant que  $0! = 1$ .

- a) Utilisez while ,
- b) Utilisez for .

#### **Exercice 17**

Calculez la somme, le produit d'une suite de chiffres non nuls entrés au clavier, sachant que la suite est terminée par zéro. retenez seulement les chiffres (0, 1 ... 9) lors de l'entrée des données et effectuez un signal sonore si les données sortent de ce domaine.

#### **Exercice 18**

Calculez le nombre lu à rebours d'un nombre positif entré au clavier en supposant que le fichier d'entrée standard contient une suite de chiffres non nuls, terminée par zéro (Contrôlez s'il s'agit vraiment de chiffres).

Exemple : Entrée : 1 2 3 4 0 Affichage : 4321

#### **Exercice 19**

Calculez le nombre lu à rebours d'un nombre positif entré au clavier en supposant que le fichier d'entrée standard contient le nombre à inverser.

Exemple : Entrée : 1234 Affichage : 4321

#### **Exercice 20**

L'application la plus importante dans l'utilisation du langage HTML est le contrôle des masques de saisies. En effet ceux-ci peuvent s'avérer coûteux en temps CPU sur le serveur HTTP chargé de réceptionner les données entrées par les utilisateurs.

Nous allons mettre en oeuvre une procédure chargée de contrôler ces saisies. Prenons la forme suivante que je vous propose de tester :

Entrez une chaîne vide :



Entrez une adresse :

Entrez un nombre :

Ecrire un script de nom « test » qui prend en paramètre le formulaire et vérifie s'il est bien remplis.

## Partie PHP

**EXERCICE 1 :** Recupérer une valeur d'une zone de texte et affectez 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.

**Objectifs :** Utilisation des instructions **WHILE** et **FOR**.



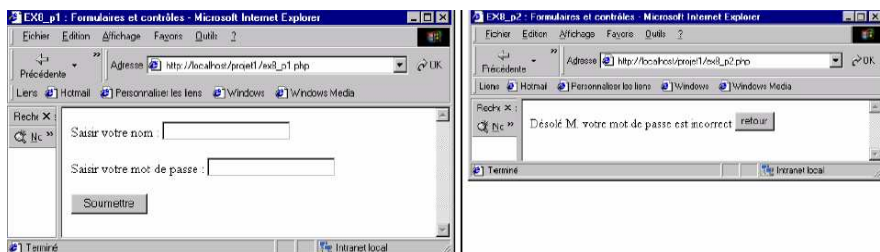
## EXERCICE 2

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.

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



## EXERCICE 3 :

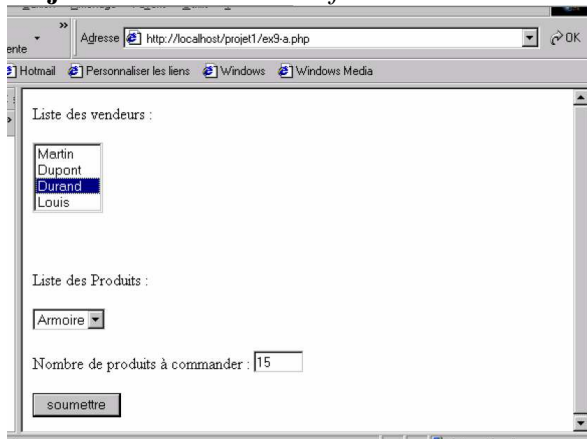
Construire une page qui permette d'afficher :

- une liste avec les noms des vendeurs suivants : M. Dupont, M. Louis, M. Martin et M. Durand (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. Martin).

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



#### EXERCICE 4

Reprendre l'exercice 5 mais effectuer le test de saisie sur le navigateur client.

**Objectifs :** *Tests de cohérence sur le client (Javascript).*

#### Exercice 5

Ouverture/lecture de fichiers

On souhaite créer avec un éditeur de texte le fichier suivant (appelé calepin.txt)

David | Martin | 3, impasse des Lilas | 64600 | Anglet

Etchebarne | Amia | 4, Bld du BAB | 64100 | Bayonne

Chirac | Joselyne | 125 avenue Paul Bert | 64200 | Biarritz

On va ouvrir ce fichier en PHP et réaliser un affichage comme suit :

Nom : David

Prénom : Martin

Adresse : 3, impasse des Lilas

CP : 64600

Ville : Anglet

#### Exercice 6

### Lecture/écriture dans un fichier

On désire réaliser un compteur de visites dans une page d'accueil. Celui-ci doit se mettre à jour à chaque chargement de la page, même si l'adresse IP de la personne est identique (cas entre autres des personnes passant par un proxy).

### Exercice 7

Premier formulaire pour un sondage

On désire créer un formulaire pour un site de dons caritatifs :

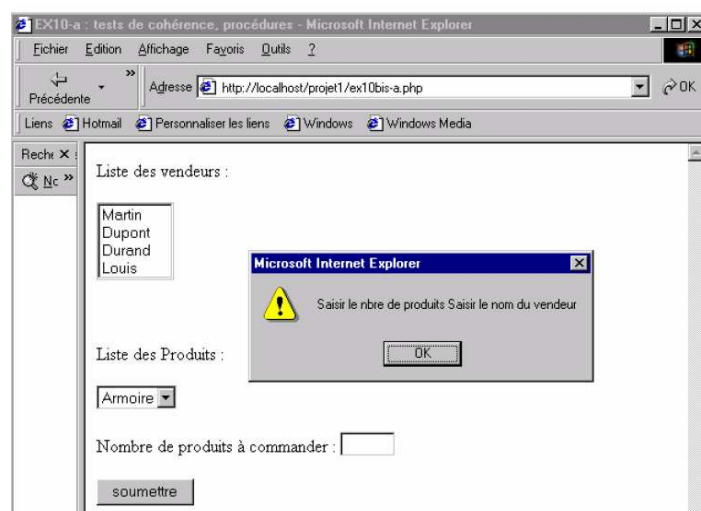
1. Nom
2. Age
3. Mail
4. Valeur en € du don

A chaque validation de formulaire (appuis sur un bouton 'OK'), on récupèrera les informations entrées par le visiteur. Ces informations seront enregistrées dans un fichier appelé 'resultats.txt' sous le format suivant :

Nom1 | Age1 | Mail1 | Don1

Nom2 | Age2 | Mail2 | Don2

On proposera une trace à l'écran des informations récupérées.



### Exercice 8

Accès aux bases de données

Objet du TP : Ce TP va permettre de voir les mécanismes nécessaires à la connexion à un SGBD de type MySQL (pour un autre SGBD, il faudra implemment adapter les noms de fonctions).

Vous créez sur votre compte mySQL une table agenda avec les informations suivantes :

Nom (50 car. Max), Prénom (50 car. Max), adresse (255 car. Max), age (entier), téléphone (10 car. Max).

Construire un formulaire permettant de remplir la table agenda avec 4 à 5 enregistrements.

Une fois ceci réalisé, vous proposerez une page permettant d'interroger la base, pour obtenir

- un classement par noms,
- un classement par age,
- la moyenne d'âge des personnes.
- Supprimer une personne de l'agenda.
- Modifier les informations d'une personne.

### **Exercice 9**

Couplage Image/BD

Sur iparla (nom d'un serveur), il existe une base appelée roose avec comme login roose et comme mot de passe drop64. Cette base contient une table appelée bourse avec deux champs : ville (chaîne de caractères) et indice (entier).

Le premier champ contient le nom d'une ville (Paris ou New York), le second contient l'indice (ne dépassant pas 20) de la bourse correspondante.

1- Construire le formulaire HTML qui permet de remplir cette table.

2- Dans un premier temps, on va accéder à cette base et afficher l'ensemble des indices classés par ville.

3- Dans un deuxième temps on va réaliser une page HTML simulant un site de bourse. Dans ce site de bourse, on affichera un histogramme généré en PHP affichant les indices de chaque bourse (une couleur différente/ville). Au dessus de chaque barre d'indice, on affichera la ville concernée avec son indice.

Ne pas oublier que pour générer une image en PHP, il faut mettre header("Content-type : image/jpeg") afin de dire que le code va produire un fichier JPEG.

Dans le fichier du site de bourse, on référencera l'histogramme généré de la manière suivante :

<IMG SRC=ficbourse.php>Un commentaire éventuel</IMG>

### **Exercice 10**

Variables de sessions

Objectif : Utilisation des variables de sessions

On désire réaliser une mire de login (login + mot de passe). Le mot de passe sera 'iut'. Dans le cas où le mot de passe sera correctement saisi, on affichera une page permettant d'accéder à la zone membre (membre.php), dans le cas contraire, on affichera login/mot de passe incorrect.

On veillera à ce que l'accès directement à la page membre.php soit refusé dans le cas où on saisisrait l'@ intégralement URL : http:// .../membre.php.