

Cours

# PROGRAMMATION ORIENTEE WEB

***SL.Hamidat***

# Sommaire

Conseils Généraux .....	3
Séquence 1 : Introduction à la POW .....	5
Séquence 2 : Présentation du site .....	16
Séquence 3 : HTML .....	22
Séquence 4 : Travail sur l'interface .....	35
Séquence 5 : JavaScript : langage client .....	55
Séquence 6 : PHP : langage serveur .....	78
Séquence 7 : XML et flux RSS.....	92
Séquence 8 : Images et objets dynamiques.....	105
Séquence 9 : Bases de données .....	125
Séquence 10 : Transferts entre pages .....	133
Séquence 11 : Templates et Frameworks.....	151
Séquence 12 : Back office .....	158
Séquence 13 : Mise en ligne.....	163

# Conseils généraux

## 1. Organisation du cours

Ce cours se présente en plusieurs séquences progressives, chaque séquence étant illustrée par une nouvelle étape de construction d'un site commercial complet, excepté la première séquence qui apporte uniquement des connaissances théoriques, mais cependant indispensables.

A chaque grande étape, il vous est demandé de réaliser une sauvegarde de votre travail sous un nom et numéro de version précis. Vous trouverez dans la correction officielle, la même organisation avec différentes versions : cela vous permettra plus facilement de contrôler votre travail.

## 3. Outils utilisés

Pour réaliser les exercices du cours, vous allez avoir besoin de 2 outils que vous allez récupérer sur internet :

- WampServer (Windows + Apache + MySQL + PHP) : serveur apache + interpréteur PHP + le sgbdr MySQL. Cet outil va vous permettre de tester vos pages en local, comme si elles étaient installées sur un serveur web. Vous trouverez cet outil sur le site <http://www.wampserver.com/>. La version utilisée au moment de la création de ce cours est la version 2.0h. Vous pouvez installer une version plus récente. Pour télécharger WampServer, vous pouvez laisser vos coordonnées cependant ce n'est pas obligatoire. WampServer est gratuit. Lors de l'installation, choisissez Internet Explorer comme navigateur par défaut, sauf si vous êtes totalement allergique à ce navigateur. De toute façon, vous aurez ensuite à tester le site sur plusieurs navigateurs. Nous verrons dans ce cours les différences entre les navigateurs principaux.
- PSPad : éditeur multi-langages. Cet outil va vous servir pour écrire le code de vos pages. Vous pourriez utiliser un éditeur simple (comme Notepad) cependant PSPad apporte des aides supplémentaires. Vous pourriez utiliser des outils plus complets comme Dreamweaver ou Zend Studio, très performants mais malheureusement payants. PSPad est un bon compromis et de plus il est gratuit. Vous le trouverez sur le site <http://www.pspad.com/fr/>. Si vous avez l'habitude d'utiliser un autre éditeur pour créer vos pages web, ce n'est pas un problème.

## Séquence 1

# Introduction à la POW

Cette séquence présente les notions fondamentales liées à Internet et au développement de sites. Il n'y a pas de mise en pratique mais les notions abordées doivent être connues avant de passer à la suite.

### ► Capacités attendues en début de séquence

Savoir ce qu'est un langage de programmation.

### ► Contenu

1. Petit historique .....	6
2. Les services accessibles .....	6
3. Les protocoles .....	7
4. Le WEB .....	7
5. Qu'est-ce qu'un site ? .....	8
6. Page statique .....	8
7. Page dynamique .....	9
8. Exemple de code .....	9
9. Architecture 3 tiers .....	11
10. Les données sur le web .....	13
11. Les outils de développement .....	13
12. La norme W3C .....	14

## 1. Petit historique

Rapide historique :

- 1964 : naissance du projet ARPANET (système distribué). C'est un réseau militaire créé pendant la guerre froide. Objectif : pouvoir communiquer même avec des lignes coupées (en prévision de bombardements). Seuls les militaires sont concernés.
- 1973 : réseau mis en place par les scientifiques (naissance du protocole TCP/IP, machines possédant une adresse IP unique). Seules certaines universités sont concernées.
- 1989 : naissance du WWW grand public (il faudra quelques années pour que cela devienne vraiment "grand public", cependant l'expansion a vite été exponentielle)

L'internet représente l'**inter**connexion de réseaux (**net**work) dans le monde entier.

## 2. Les services accessibles

Plusieurs services sont accessibles sur Internet. Certains ne sont plus ou quasiment plus utilisés, d'autres au contraire ont pris beaucoup d'ampleur :

- **WWW** (World Wide Web) : ensemble des sites. C'est ce que vous utilisez en naviguant sur la toile. Utilise le protocole http.
- **FTP** (File Transfer Protocol) : téléchargement de fichiers. Vous accédez parfois à des liens dont l'adresse commence par FTP et qui vous permettent de télécharger des fichiers. Utilise le protocole ftp.
- **Courrier électronique** : messagerie. Tout ce qui touche aux mails. Utilise les protocoles smtp, pop3...
- **News Groups** : groupes de discussions. Très utilisés dans les premières années, ils sont maintenant essentiellement remplacés par les forums. Utilise le protocole nntp.
- **IRC** (Internet Relay Chat) : discussion en direct. C'est le "chat" que vous devez certainement beaucoup pratiquer. Utilise le protocole irc.
- etc...

Conclusion, ne confondez pas Internet et Web. Le second n'est qu'un des services offerts par Internet.

### 3. Les protocoles

#### Pourquoi des protocoles ?

Chaque service offert par Internet fonctionne avec un protocole précis qui définit les règles d'encodage et de transfert des informations. Ces règles étant universelles, tous les réseaux reliés à l'Internet sont capables de comprendre les informations qui sont véhiculées.

#### Les principaux protocoles

Ces protocoles, vous aurez l'occasion de les étudier plus en détail dans le cours approprié. Vous apprendrez, entre autre, qu'ils n'interviennent pas tous au même "niveau". Certains sont des protocoles d'application, d'autres des protocoles réseau... Voici juste à titre indicatif les protocoles les plus utilisés sur Internet :

- **TCP** (Transmission Control Protocol) : protocole de transport
- **IP** (Internet Protocol) : protocole réseau
- **HTTP** (Hyper Text Transfer Protocol) : protocole d'application
- **FTP** (File Transfer Protocol) : protocole d'application

#### Le protocole IP

Chaque machine est identifiée sur le réseau mondial par une adresse IP unique de type :

X.X.X.X (avec X compris entre 0 et 255 =  $2^8$ )

Il est possible d'affecter un nom unique à une adresse IP. Dans ce cas, un serveur DNS s'occupe de la résolution de nom. C'est le principe utilisé sur Internet où, pour accéder à un site, vous ne tapez pas son adresse IP mais un nom composé.

### 4. Le WEB

World Wide Web (la toile d'araignée mondiale)

#### C'est quoi ?

Le Web représente l'ensemble des sites accessibles par Internet.

#### Qu'est-ce qu'un site ?

Un site est un ensemble de pages consultables à partir d'un navigateur. Un site possède une adresse unique : l'adresse du serveur qui l'héberge.

#### Qu'est-ce qu'un serveur Web ?

Un serveur Web est un logiciel qui permet d'héberger et de gérer des sites. Très souvent une machine est dédiée à ce travail ce qui fait que quand on parle de serveur web, on parle directement de la machine qui héberge les sites.

Le nom du serveur Web est généralement www. Le serveur étant dans un domaine, son nom complet est sous la forme www.nomdomaine.

Le port d'écoute d'un serveur Web est par défaut le **port 80**. Ce port peut être changé. Cependant, sauf en intranet et pour des applications spécifiques, il est conseillé de ne pas toucher au port, car cela forcerait les utilisateurs à préciser le port dans l'adresse.

## 5. Qu'est-ce qu'un site ?

Vous savez ce que c'est un site, mais voyons plutôt les catégories.

### Les catégories d'accès

Les sites ne sont pas tous accessibles de la même façon :

- **sites publics** : accessibles à tous
- **sites d'accès limités** : accessibles avec mot de passe (cela peut concerner un site entier, ou uniquement certaines pages)
- **sites protégés (ou cryptés)** : accessibles avec le protocole https (généralement utilisé pour le paiement sécurisé : là aussi, cela peut concerner un site entier ou certaines pages)
- **sites privés** : accessibles dans un réseau fermé, non connecté à Internet

### Les catégories de contenu

Dans un site, les pages sont classées en 2 grandes catégories, en fonction de leur contenu.

- **pages statiques** : informations fixes. Les données contenues dans les pages statiques sont directement intégrées "en dur" dans les balises et ne sont pas modifiables (sauf en modifiant directement le code de la page)
- **pages dynamiques** : informations variables. Les données contenues dans les pages dynamiques proviennent en partie d'origines extérieures, comme de bases de données, de fichiers XML ou autre. Du coup, lors de la modification des données extérieures, le contenu des pages varie.

### Contenu d'un site

Un site peut être constitué d'une ou plusieurs pages. Un site peut comporter des pages statiques, des pages dynamiques ou un mélange des deux. Vous allez apprendre à coder ces deux catégories de pages.

## 6. Page statique

### Qu'est-ce que c'est ?

Elle ne contient que des informations fixes. C'est généralement une page d'information simple, par exemple la présentation d'une entreprise.

### Quels langages ?


Un page statique utilise essentiellement le langage HTML qui permet juste de formater des informations.

Le langage JavaScript est normalement considéré comme un langage de pages statiques puisqu'il ne s'exécute que côté client, donc sans utiliser des données provenant de l'extérieur. Cependant, il est difficile de parler de "statique" en parlant de JavaScript, pour deux raisons : d'abord parce qu'on peut faire de jolies choses "qui bougent" à l'écran et ensuite parce que JavaScript peut, avec Ajax, se connecter au serveur sans que cela se voit. Toutes ces notions seront détaillées par la suite.

### **Comment ça marche ?**

Les balises html sont interprétées par le navigateur et ainsi permettent de formater l'affichage des informations directement insérées dans la page.

Par exemple, pour afficher le mot "bonjour" en gras :

 `<strong>Bonjour</strong>`

## **7. Page dynamique**

### **Qu'est-ce que c'est ?**

Elle contient des informations fixes et des informations variables. C'est par exemple une page affichant des articles avec leurs prix : les données sont censées changer dans le temps, donc il est logique qu'elles soient stockées par exemple dans une base de données.

### **Quel langage ?**

Les langages utilisés doivent être capables de manipuler des informations qui sont stockées côté serveur, comme une base de données, un fichier XML...

Côté serveur, on peut trouver par exemple les langages suivants : PHP, ASP, JSP...

Le choix du langage doit se faire en fonction de différents critères (puissance, sécurité...) mais aussi des possibilités d'hébergements. Voilà pourquoi, pour de nombreux sites personnels et même professionnels, le PHP est favorisé car il est gratuit et du coup la majorité des hébergeurs le proposent.

### **Comment ça marche ?**

Le code dynamique est exécuté côté serveur afin de récupérer les données nécessaires, puis le résultat (sous forme HTML et éventuellement JavaScript) est envoyé au navigateur client. C'est le principe de l'architecture 3 tiers qui va être étudié plus loin.



## 8. Exemple de code

### Aspect multi-langages

La particularité de la programmation d'un site par rapport à une application classique réside dans l'aspect multi-langages.

Un site contient :

- des parties comportant du code qui sert à afficher les informations (les balises html),
- d'autres parties comportant du code un peu plus évolué mais qui s'exécute directement dans le navigateur sur la machine cliente (le JavaScript)
- et enfin d'autres parties comportant du code pour manipuler des informations côté serveur (comme des bases de données) et qui s'exécute donc uniquement côté serveur (le PHP, par exemple).

### Exemple de code

Pour que les choses soient tout de suite plus claires, voyons un exemple de page qui mélange ainsi les 3 langages :

```
<html>

  <script language="JScript">
    function Hello(message) {
      var chaine = "Vous avez cliqué sur : " + message
      window.alert(chaine)
    }
  </script>

  Liste des classes (cliquez sur le bouton de votre
  classe) :
```

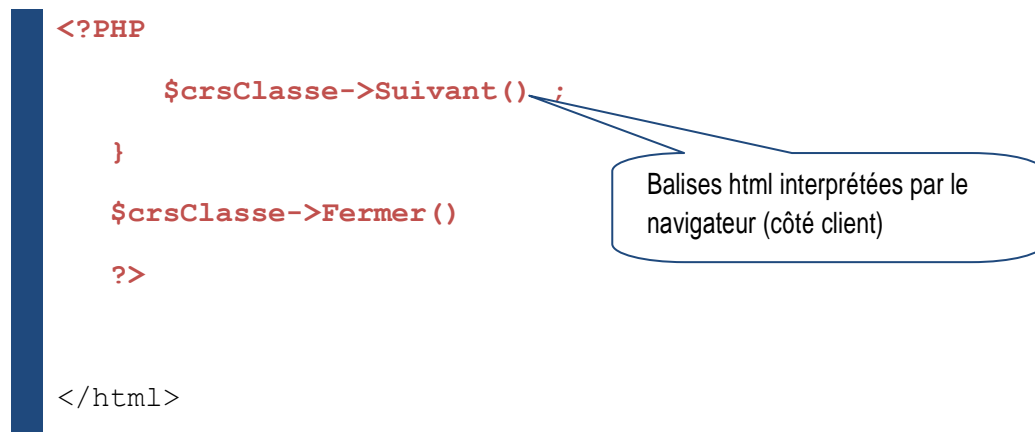
Code JavaScript client interprété par le navigateur (côté client)

```
<?php
  $crsClasse = new Curseur() ;
  $crsClasse->reqSelect("select nom from classe") ;
  while (!$crsClasse->Eof()) {
    ?> $val = $crsClasse->Champs("nom") ;

    <input type="BUTTON" value="<?php echo $val ?>"
      onclick="Hello('<?php echo $val ?>')">
```

Code PHP interprété par le serveur de traitements (côté serveur)

```
<?PHP  
  
    $crsClasse->Suivant();  
}  
  
$crsClasse->Fermer()  
  
?>  
  
</html>
```



Balises html interprétées par le navigateur (côté client)

Que fait cette page de code ?

Le code JavaScript du haut de la page n'est exécuté qu'au moment où la fonction concernée est appelée. Donc pour le moment, on l'oublie. Ensuite une ligne de titre s'affiche ("Liste des classes...").

Puis le code PHP va interroger la base de données avec un curseur, pour récupérer les noms des classes. Une boucle permet d'afficher une balise html qui va s'occuper d'afficher un bouton avec comme titre, le nom de la classe (récupérée dans le curseur) et sur le clic du bouton, la fonction JavaScript est appelée avec comme paramètre le nom de la classe.

La fonction JavaScript s'occupe d'afficher une fenêtre avec comme message "vous avez cliqué sur " suivi du paramètre, donc du nom de la classe.

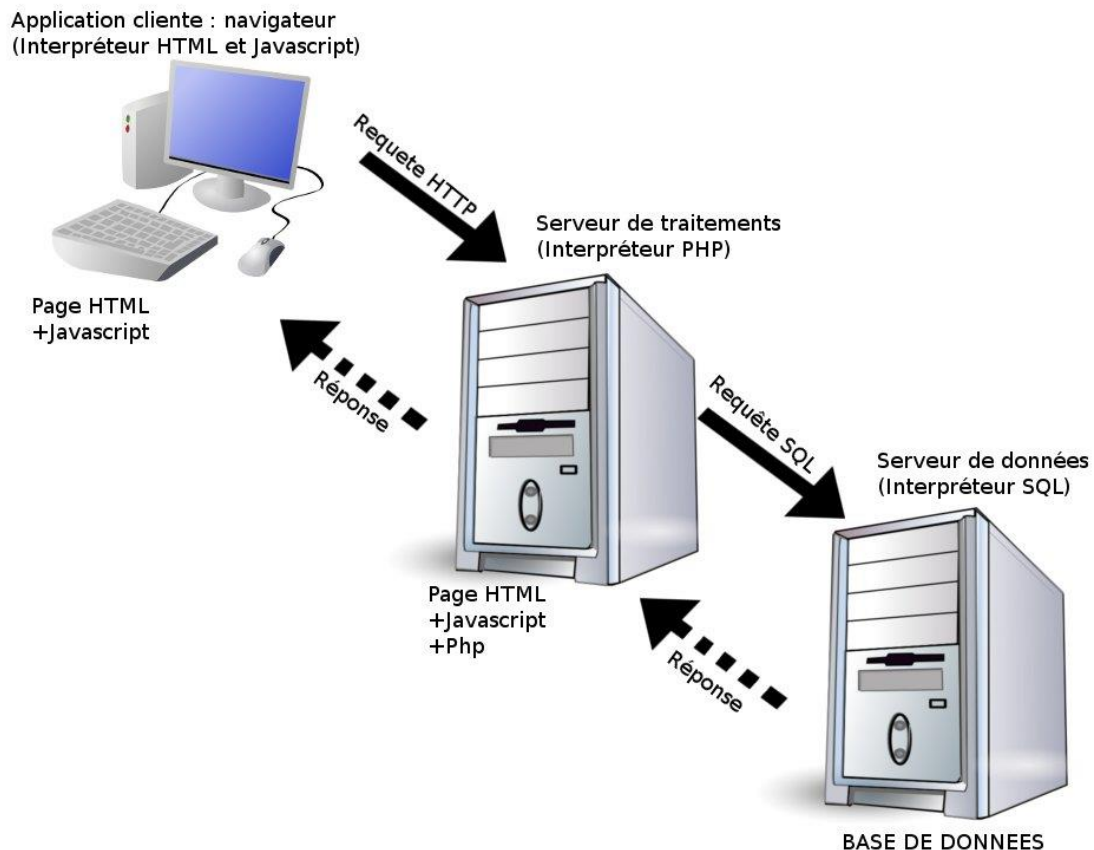
### Le serveur de données

Le code côté serveur a souvent besoin d'accéder aux données du serveur, par exemple à une base de données, comme on le voit dans ce petit exemple. Dans ce cas, le serveur de traitements, qui s'occupe d'interpréter le code serveur, va faire appel au serveur de données, à travers par exemple des requêtes SQL. Ce dernier va retourner le résultat de la requête au serveur de traitements.

## 9. Architecture 3 tiers

L'exécution d'une page de code d'un site se fait sur le principe de l'architecture 3 tiers. Un client fait appel à un serveur de traitements qui lui-même fait appel à un serveur de données.

Dans le cas de pages statiques, seul le client et le serveur de traitements interviennent. Le serveur de données n'intervient que sur les pages dynamiques.



Les serveurs de traitements et de données peuvent éventuellement être installés sur la même machine.

#### Principe :

- vous tapez, dans la barre d'adresse du navigateur, l'adresse d'une page de site (ou vous cliquez sur un lien)
- une requête http est envoyée au serveur web qui héberge la page
- le serveur web invoque l'interpréteur PHP (ou autre langage serveur) pour traduire le code PHP de la page en code html (le code JavaScript n'est pas modifié)
- le code PHP qui manipule des curseurs envoie des requêtes SQL au serveur de données auquel il s'est connecté
- le serveur de données qui héberge la base de données sollicitée, renvoie en réponse, au serveur web, le résultat de la requête SQL
- l'interpréteur PHP inclut le résultat de la requête dans le code html de la page
- La page, une fois interprétée, donc ne contenant plus que du code client (html et JavaScript), est envoyée en réponse au client, donc au navigateur

- La page, une fois côté client, est interprétée par le navigateur qui va utiliser les balises html et exécuter le code JavaScript pour afficher la page

## 10. Les données sur le web

Les sites dynamiques, nous l'avons vu, manipulent des données qui sont mémorisées côté serveur. Il existe différentes solutions pour mémoriser des données :

### Les bases de données

Elles apportent toute la puissance de la gestion d'une base de données (sécurité, verrous...) ainsi que la possibilité d'exploiter les données par le SQL. Une base de données peut aussi être partagée, réutilisée par d'autres applications.

Tous les formats de bases de données sont utilisables sur un site : MySQL, SQL Serveur, Oracle, posgreSQL, Access....

Le langage PHP, comme tous les langages serveurs, fonctionne donc avec tous les formats de bases de données, mais on l'associe plus classiquement avec le SGBDR MySQL.

Quand vous choisissez un sgbdr, si vous n'avez pas de contraintes particulières, il est préférable d'en choisir un qui est majoritairement proposé par les hébergeurs. C'est le cas de MySQL, entre autre parce qu'il est gratuit.

### Les fichiers XML

Un fichier XML est un fichier texte balisé (dans le même esprit que le HTML) avec des balises nommées par le créateur du fichier et respectant certaines règles de présentation. Ce format de fichier est universel, directement reconnu par les navigateurs. Par rapport à une base de données, le format est nettement plus léger et transportable, mais il n'offre pas la puissance d'une base de données (par de SQL, pas de sécurité ni gestion de verrous). Cependant, la recherche d'informations dans un fichier XML est maintenant facilitée dans tous les langages qui incluent des fonctionnalités spécifiques aux fichiers XML (appelées des "parseurs" de fichiers XML).

### Les autres fichiers (textes, images...)

Le langage serveur peut exploiter d'autres fichiers côté serveur, comme de simples fichiers texte, ou des PDF, des images, des vidéos...

## 11. Les outils de développement

Il existe plusieurs méthodes pour créer un site :

- en écrivant le code directement dans un éditeur simple (Notepad...) : cela suppose que vous ayez une bonne connaissance des syntaxes.
- En écrivant le code dans un éditeur ou environnement plus évolué qui reconnaît la syntaxe des langages utilisés (PSPad...) : vous devez toujours connaître les

syntaxes, cependant des facilités sont apportées, comme la colorisation du code.

- En écrivant le code dans un générateur de code (Dreamweaver, Zend Studio...) : vous pouvez directement construire la page en utilisant les outils graphiques, le code se générant automatiquement. La saisie du code se fait avec l'aide assistée (liste des méthodes qui apparaissent automatiquement...). Cette solution a cependant comme inconvénient de générer un code plus lourd.

Nous utiliserons la seconde solution.

## 12. La norme W3C

Depuis 1994, Le W3C (World Wide Web Consortium) a mis au point un ensemble de règles précises à respecter dans l'écriture d'un site (uniquement le code client). Ces règles sont là pour palier aux importantes différences qui existent entre les navigateurs. Vous avez certainement remarqué que parfois, d'un navigateur à l'autre, les pages ne s'affichent pas tout à fait de la même façon. La norme W3C, lorsqu'elle est respectée, permet d'obtenir un site qui fonctionnera de façon quasi équivalente quelque soit le navigateur.

A la fin du tp, vous apprendrez comment contrôler le contenu d'un site en respectant cette norme.

### Synthèse

Protocole :

Ensemble de règles afin de faire en sorte que 2 entités qui communiquent entre elles soient capables de se comprendre.

Les services accessibles :

WWW (World Wide Web) : ensemble des sites

FTP (File Transfer Protocol) : téléchargement de fichiers

Courrier électronique : messagerie

IRC (Internet Relay Chat) : discussion en direct

News Groups : groupes de discussions (remplacés par les forums)

Les protocoles les plus utilisés sur Internet :

TCP (Transmission Control Protocole) : protocole de transport

IP (Internet Protocole) : protocole réseau déterminant le formatage des adresses

HTTP (Hyper Text Transfer Protocol) : protocole d'application utilisé pour les pages web

FTP (File Transfer Protocol) : protocole d'application utilisé pour le transfert des

fichiers

Site internet :

Ensemble de pages statiques (contenu fixe) et/ou dynamiques (contenu variables).

Un site internet peut être public (accessible à tous), d'accès limité (accessible par mot de passe), protégé (avec le protocole https), privé (accessible dans un réseau fermé).

Langages du web :

Langages clients : HTML (balisage pour l'affichage), JavaScript

Langages serveurs : PHP, ASP, JSP...

Données sur le web (côté serveur) :

Bases de données : MySQL (très répandu), PostgreSQL, SQL Serveur, Oracle,...

XML : format texte balisé, standard

Architecture 3 tiers :

Un client (navigateur) envoie une requête http au serveur web qui interprète le code serveur (par exemple PHP) en interrogeant éventuellement le serveur de données et retourne au client une page ne contenant que du code client (html et JavaScript).

## Séquence 2

# Présentation du site

Cette séquence présente les points essentiels à aborder lors de la réalisation d'un site, avant même de commencer à coder : recenser les fonctionnalités, les organiser correctement, repérer les données nécessaires et les structurer.

### ► Capacités attendues en début de séquence

Avoir quelques notions de gestion de projet. Savoir ce qu'est une base de données.

### ► Contenu

1. But du site .....	17
2. Structure du site .....	18
3. Structure des données .....	19
4. Construction du projet .....	19

## 1. But du site

Avant de commencer à créer un site, il faut se poser les bonnes questions.

### Pourquoi un site ?

La décision d'élaboration d'un site suppose qu'un objectif précis est recherché. Il faut dans un premier temps se demander dans quel cadre doit être utilisé le site :

**site en intranet** : il permet, dans le cadre d'un intranet donc d'un nombre d'utilisateurs restreints, de mettre à disposition des informations et des traitements. Ce type de site est souvent utilisé en entreprise. Il a parfois une ouverture sécurisée sur internet, pour que les collaborateurs puissent y accéder de chez eux.

**site public sécurisé** : il est accessible sur internet, mais en accès limité. Certains sites sont entièrement sécurisés, d'autres ne possèdent que certaines pages sécurisées.

**site grand public** : il est accessible à tous et souvent cherche à être connu de tous.

Dans le cadre de notre exemple, le site sera une boutique en ligne pour vendre des t-shirts personnalisés et des articles divers. Ce sera donc un site grand public commercial.

### Pour qui ?

Le but est de déterminer le public concerné par le site. Cela peut aller d'un groupe de personnes faisant parti d'un service d'une entreprise, à la planète entière. La détermination du public va pousser à mettre en place d'éventuelles restrictions d'accès.

Dans le cadre d'un site commercial, sauf cas particulier, il faut éviter les restrictions qui pourraient faire fuir les éventuels acheteurs.

### Quelles sont les fonctionnalités nécessaires ?

Qu'est-ce que les utilisateurs doivent pouvoir consulter ou faire sur le site ? La détermination des fonctionnalités doit être d'une grande précision.

Pour notre site, voici les fonctionnalités nécessaires :

- consultation et possibilité d'achats des différents articles
- constitution et achat de t-shirts personnalisés
- consultation de news
- consultation et modification des données personnelles
- possibilité de laisser un message
- affichage de liens vers d'autres sites

### Comment et par qui va-t-il être géré ?

Ce point est souvent occulté et pourtant il est fondamental. Tout site doit être géré par au moins une personne. Même un simple site vitrine doit être mis à jour régulièrement. Il ne faut donc pas penser juste à la création du site mais aussi à son suivi. Plusieurs points sont à prévoir :

- mise à jour du contenu informatif fixe des pages (normalement rare, sinon les données ne doivent pas être mises en fixe)



- mise à jour des informations variables intégrées dans les pages (par exemple, les prix des articles, les news...)
- traitement des messages (si l'internaute a la possibilité de laisser un message)
- modération (si l'internaute a la possibilité de laisser des informations, par exemple dans un livre d'or ou un forum)
- traitement des commandes (pour un site commercial)

Une fois que le but du site et son fonctionnement sont clairement définis, il est possible de passer à la phase de conception du site.

## 2. Structure du site

Un site se présente sous la forme d'un ensemble de pages reliées entre elles. Dès le début, il faut réfléchir sur la structure du site en déterminant les différentes pages et les liens entre elles. La réflexion sur la structure est importante car elle va avoir un impact non négligeable sur le comportement de l'internaute. Si la navigation est trop complexe, trop longue, si les informations ne sont pas claires, l'utilisateur va se décourager et partir du site.

Il faut donc respecter quelques règles :

- limiter les manipulations de navigation
- construire des pages claires donc peu surchargées
- éviter de demander des informations personnelles inutiles et surtout trop tôt dans la navigation

Un des points les plus complexes est de trouver le bon compromis entre "pages non surchargées" et "nombre de pages limitées".

Pour notre site, voici la liste des pages qui seront construites :

- Page d'accueil : elle contiendra une présentation rapide de la boutique, les news, le menu qui permet d'accéder aux différentes parties du site, et les liens. Cette page permettra aussi de s'authentifier.
- Page t-shirt : elle permettra de construire un t-shirt personnalisé et de l'ajouter dans le panier.
- Page boutique : elle affichera les articles mis en vente et permettra d'en ajouter dans le panier.
- Page panier : elle affichera le panier, permettra de retirer certains articles, et offrira la possibilité de passer commande.
- Page enregistrement de la commande : cette page ne sera accessible qu'après avoir validé le panier, et ce ne sera qu'à ce moment là qu'on demandera des informations sur le client.
- Page personnelle : elle permettra à un internaute de s'inscrire, consulter ou modifier ses informations personnelles.

### 3. Structure des données

Dans le cas d'un site manipulant des données variables, il faut prévoir le moyen de stocker et de mettre à jour ces données. Bien sûr, il n'est pas envisageable de stocker ses informations directement dans le code des pages, et de modifier le code à chaque fois que cela est nécessaire.

#### Base de données

C'est la forme la plus classique. La plupart du temps, vous choisirez cette solution.

#### Fichier XML

Adapté pour stocker des informations non modifiables par les internautes (ce n'est pas irréalisable, mais le fichier XML ne gère pas les concurrences d'accès, donc à éviter) et avec peu de recherches. Le fichier XML est très adapté pour le stockage des news, car il peut ensuite être utilisé comme flux rss.

#### Fichier image

Il est possible de stocker des informations derrière certains formats d'images. Ce système de stockage d'informations est rare mais très adapté dans le cas d'une gestion de galeries.

#### Fichier texte

Son intérêt est très limité. Il peut cependant être pratique pour mémoriser un texte informatif et l'afficher.

Dans le cadre de notre site, le but va être d'aborder le maximum de structures de données possibles. La base de données va mémoriser les informations sur les clients et les commandes, le fichier XML contiendra les news, les fichiers image seront utilisés pour mémoriser les informations des ajouts possibles sur les t-shirts et les articles.

### 4. Construction du projet

Il est temps de mettre en place le projet du site qui va être construit étape par étape et qui va permettre de réaliser un ensemble de tests.

#### Dossier de travail et Serveur

Si vous n'avez pas encore installé WampServer, faites-le. Dans le dossier www de wamp (normalement c:\wamp\www si vous n'avez pas changé le chemin d'installation) vous allez créer un dossier qui contiendra tous les fichiers de votre site : nommez le **maBoutiquePerso**.

Pour tester les pages, le serveur doit être lancé : lancez WampServer. Normalement un icône s'est ajouté dans la barre en bas à droite. En fin de lancement, après avoir pris différentes couleurs (rouge et jaune) il se présente sous la forme d'un demi-cercle blanc. S'il n'est pas complètement blanc, c'est qu'un ou plusieurs services ne se sont pas lancés correctement. Vérifiez alors que rien ne rentre en conflit avec WampServer (par exemple un serveur IIS qui serait déjà installé et qui utiliserait

déjà le port 80, ou peut-être tout simplement votre firewall qui bloque le démarrage des services).

### Travail sous l'éditeur

Si vous n'avez pas encore installé PSPad, faites-le (sauf si vous préférez utiliser un autre éditeur). Les explications suivantes sont adaptées à PSPad mais facilement adaptables à un autre éditeur. A vous d'adapter, si besoin.

Lancez PSPad. On va commencer par lier le dossier créé à un nouveau projet : allez dans le menu "Projets/Créer projet d'un répertoire...". Sélectionnez le répertoire que vous venez de créer, puis OK. Remarquez dans la partie gauche de l'éditeur, le dossier porte le nom de maBoutiquePerso. Quelque soit l'éditeur utilisé, ce lien peut apporter des fonctionnalités pratiques mais n'est pas obligatoire : vous pouvez travailler sur les pages sans passer par un projet.

Pour créer un nouveau fichier, faites "Fichier/Nouveau...". Vous remarquez un large choix de type de fichiers. Sélectionnez HTML puis OK. Remarquez que le fichier qui vient de s'ouvrir contient déjà un peu de code. Nous verrons plus loin à quoi correspond ce code. Remarquez aussi l'onglet qui s'est ajouté au dessus à gauche du code et qui contient le nom "Nouveau.html". Vous comprenez que vous pourrez travailler avec plusieurs fichiers à la fois, dans des onglets différents. Pour travailler sur le fichier (changer son nom, l'enregistrer...), le plus rapide est de faire un clic droit sur l'onglet. Faites-le pour sélectionner "Ajouter ce fichier au projet". Si cette option est en grisée, commencez par cliquer à gauche sur "maBoutiquePerso" pour sélectionner le projet, puis recommencez. Cette fois le fichier apparaît à gauche dans l'arborescence du projet. Là encore, ce lien entre fichier et projet n'est pas obligatoire. Le fichier n'est toujours pas enregistré : faites à nouveau un clic droit sur l'onglet et choisissez "Enregistrer sous...". Normalement, dans la fenêtre qui vient de s'ouvrir, vous êtes automatiquement positionné sur le dossier du projet. Donnez au fichier le nom de index.html et Enregistrez.

### Tests

Faisons un tout premier test : dans le fichier, entre les balises <body> et </body>, écrivez juste :

 Bonjour

Enregistrez (avec ctrl-S, c'est encore plus rapide).

Voici la procédure pour tester une page. Pour cela, WampServer doit être lancé. Le petit bonhomme suivant sera présent pour chaque test.



Faites un clic normal (gauche) sur l'icône de WampServer et sélectionnez localhost. Vous tombez sur une page spécifique, et vers le bas, dans la partie "Vos Projets", vous devriez trouver "maBoutiquePerso". Cliquez dessus. Vous devriez obtenir une page blanche avec juste marqué "bonjour".

Si vous avez obtenu "bonjour" et que dans la barre d'adresse vous avez une adresse qui commence par http://localhost, c'est que les étapes précédentes ont été correctement réalisées. Logiquement, en cliquant sur maBoutiquePerso, vous auriez du voir le contenu du dossier. Mais lorsqu'un dossier contient un fichier index, ce dernier se lance automatiquement. Ceci est la configuration par défaut des serveurs web, configuration qui peut éventuellement être changée.

La procédure qui vient d'être expliquée en détail pour tester le site ne sera pas réexpliquée par la suite. Il sera juste demandé de "tester le site".

Vous pouvez fermer le navigateur.

## Synthèse

Déterminer le but du site :

Intérêt du site, public concerné, fonctionnalités attendues, responsables du site.

Fixer la structure du site :

Déterminer les différentes pages et les fonctionnalités par page.

Structurer les données :

Recenser les données nécessaires et les organiser, généralement dans une base de données.

## Séquence 3

# HTML

Cette séquence présente les notions fondamentales à connaître au niveau du langage de balisage HTML : langage nécessaire à la création d'une page.

### ► Capacités attendues en début de séquence

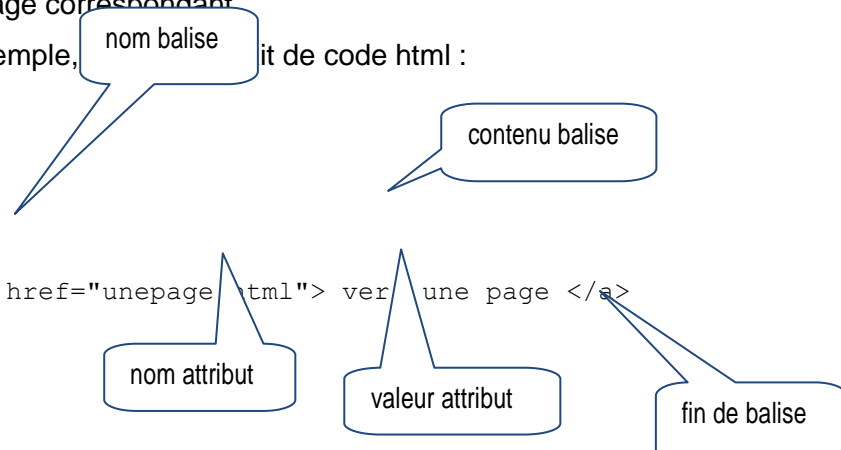
Aucune capacité attendue.

### ► Contenu

1. Introduction .....	23
2. Structure d'une page HTML .....	24
3. Les balises de l'entête .....	26
4. Les balises simples .....	28
5. Les liens et objets réactifs .....	30

## 1. Introduction

Le HTML (Hyper Text Markup Language) est le langage de présentation des pages. Il est constitué exclusivement de balises ayant chacune une fonction précise. Le navigateur va interpréter la balise et réaliser le traitement associé afin de gérer l'affichage correspondant.

Par exemple,  it de code html :

```
<a href="unepage.html"> voir une page </a>
```

Cet exemple va afficher un lien afin d'accéder à une autre page. Il montre la structure classique d'une balise qui comporte un nom, 0 à plusieurs attributs, un contenu (optionnel) et une marque de fin de balise.

```
<nombalise attributs> contenu </nombalise>
```

Une balise peut ne pas avoir de contenu, dans ce cas il existe un raccourci d'écritures.

```
<nombalise attributs />
```

Suivant le type de contrôle sélectionné (ce point va être développé un peu plus loin) le navigateur ne fait pas d'erreur si la marque de fin de balise (entière ou raccourcie) est omise. Il est cependant important de ne pas l'oublier : les nouvelles normes vont dans ce sens. En fin de cours, on testera d'ailleurs le site avec la norme W3C.

A quoi servent les balises ? Elles permettent d'afficher du texte formaté, des images, des liens, des contrôles graphiques (listes, boutons...), etc...

Le texte non entouré de balises est affiché tel quel dans une page, excepté certains caractères spéciaux et les accents. Suivant les encodages utilisés (qui seront développés plus loin), il faut utiliser des correspondances pour les obtenir. Voici quelques exemples de correspondances :

Caractère spécial	Correspondance	Caractère spécial	Correspondance
&acute;	é	&ccedil;	ç
&Eacute;	É	&lt;	<
&egrave;	è	&gt;	>

&ecirc;	ê	&amp;	&
&agrave;	à	&quot;	"
&iuml;	ï	&nbsp;	Forcer un espace

## 2. Structure d'une page HTML

La structure d'une page HTML respecte un certain nombre de règles. Encore une fois, suivant les normes utilisées, l'absence de certaines balises n'empêche pas la gestion de l'affichage. Cependant, pour aller dans le sens des nouvelles normes, une certaine rigueur est préconisée pour obtenir des résultats optimaux. Le respect de la norme W3C permet de se garantir d'un affichage compatible avec 100% des navigateurs.

Voici la structure classique d'une page html :

```
<html>

  <head>

    <title>le titre de la page</title>

  </head>

  <body>

    corps de la page avec balises d'affichage

  </body>

</html>
```

Vous retrouvez d'ailleurs ces balises dans votre fichier index.html, avec d'autres balises supplémentaires qui seront abordées plus loin.

Petite information complémentaire sur la balise `<html>` : avec la norme xhtml, cette balise doit respecter une certaine syntaxe pour respecter le w3c. Même si la balise `<html>` en l'état, fonctionne, voici sa nouvelle syntaxe, que vous utiliserez dans votre site :

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr">
```

Dans la balise `<head>`, plusieurs balises peuvent être insérées sans avoir d'influence sur l'affichage. La balise `<title>` garde seule une influence puisque le titre est celui qui s'affichera dans la barre de titre du navigateur. La balise `<head>` va être détaillée juste après.

Tout le texte et les balises d'affichage et de formatage de la page vont se trouver dans la balise `<body>`.

Cette structure permet de mettre en relief un autre élément du langage HTML : la structure en arbre. Les balises doivent obligatoirement s'emboîter les unes dans les autres. Voici 2 exemples pour illustrer ce point :

```

<baliseA>

  <baliseB>
    contenu
  </baliseB>
</baliseA>

```

CORRECT

Ce premier exemple est correct car la baliseA ne se ferme qu'après la fermeture de la baliseB puisque la baliseB a été ouverte après la baliseA.

```

<baliseA>
  <baliseB>
    contenu
  </baliseA>
</baliseB>

```

INCORRECT

Ce second exemple est incorrect : les balises ne peuvent pas s'entrelacer mais uniquement s'emboîter.

Les indentations présentées dans ces exemples, sans être obligatoires, sont fortement conseillées, justement pour contrôler la bonne utilisation des balises.

Avant même la première balise HTML, la norme w3c impose une balise DOCTYPE qui permet de préciser la norme respectée par le code de la page et le fichier DTD à utiliser pour contrôler cette norme. Un fichier DTD contient tout simplement toutes les règles que doit respecter un fichier écrit dans un langage de balise (HTML, XML...) : on verra en détail le contenu de ce type de fichier plus loin dans ce cours. Voici donc un exemple de ligne qui peut être ajoutée (à mettre sur une seule ligne) :

```

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

```

Ici on demande le respect de la norme HTML 4, qui reste assez permissif. Si vous choisissez la norme XHTML, vous ne pourrez pas oublier de fermer les balises, par exemple. Voici un exemple de déclaration avec la norme XHTML 1.1 :

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

```

Utilisez de préférence toujours la norme la plus récente (donc à l'heure de la création de ce tp, la dernière ligne, utilisant XHTML 1.1).

Pour avoir un premier fichier correctement formaté (le fichier index.html va nous servir comme fichier de référence), remplacez la ligne doctype du fichier par la ligne du dessus (avec la norme XHTML 1.1) ou une ligne proposant une norme plus récente, si vous le désirez.



### 3. Les balises de l'entête

Dans la balise <head>, plusieurs balises peuvent être insérées, sans être obligatoires.

#### Le titre

La balise <title> permet de préciser le titre qui va s'afficher dans la barre de titre du navigateur.

```
<title>Ma Boutique Personnelle</title>
```

Mettez ce titre dans le fichier index.html.



Faites un test pour contrôler que dans la barre de titre, vous obtenez bien "Ma Boutique Personnelle".

#### Les balises meta

Les balises <meta> apportent des informations sur la page et sont utilisées pour le référencement.

```
<meta name="author" content="Ed" />

<meta name="keywords" content="boutique, t-shirt" />

<meta name="description" content="Vente de t-shirts et
objets de collection" />

<meta name="date" content="2009-08-01T12:10:15+01:00" />
```

Ces balises ne vont rien modifier au niveau affichage de la page. En revanche elles sont scrutées par les robots de référencement et apportent des informations intéressantes sur l'auteur de la page (author), les mots clés qui caractérisent la page (keywords), la description textuelle du contenu de la page, la date de création de la page, etc...

Dans index.html, enlevez la balise meta du nom de "generator" (si elle est présente) et ajoutez les balises meta citées au dessus. Vous remarquerez, lors de la saisie, des petites aides apportées par l'éditeur (colorisation, insertion automatique de certains caractères...).

#### L'encodage de caractères

Un même caractère peut être interprété de différentes façons suivant l'encodage utilisé. Préciser l'encodage permet d'éviter certaines surprises comme le problème des accents mal interprétés.

```
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1" />
```

En France, on utilise classiquement 3 types d'encodage :

- iso-8859-1 (ou latin1) : permet d'encoder tous les caractères utilisés en français (accents...). Chaque caractère est codé sur 1 octet.
- iso-8859-15 (ou latin9) : ajoute à l'encodage précédent le signe "€" et le "œ".

- utf-8 : permet normalement de coder toutes les langues. Le principe est que chaque caractère est codé sur 1 à 6 octets ! (s'il est codé sur 1 octet, il commence par le bit 0 et contient un code ascii classique, respectant donc la même norme que iso-8859-1)

Il existe aussi des encodages spécifiques à Windows (c'est ce que propose la version de PSPad utilisée pour réaliser ce tp). Il est préférable d'éviter tous les encodages trop spécifiques (comme celui de Windows). Chaque éditeur utilise donc un encodage précis pour enregistrer ses fichiers. Ne pas utiliser le même encodage pour ensuite lire le fichier (en utilisant un éditeur ou un navigateur) peut créer des surprises.

Le fait de préciser l'encodage utilisé pour l'enregistrement de la page, en ajoutant la balise meta correspondante, permet normalement de forcer le navigateur à utiliser cet encodage pour afficher la page. Mais attention, ce n'est pas si simple. Cet encodage sera pris en compte que si le serveur ne précise pas un encodage par défaut. Normalement, le serveur Apache et le langage PHP fonctionnent par défaut avec iso-8859-1. Il est possible de modifier cette configuration de la façon suivante (ceci est donné à titre indicatif, en cas de besoin, mais là, ne le faites pas) :

- pour apache : dans le fichier httpd.conf, pour passer par exemple en utf-8, il faut ajouter (ou modifier) la ligne suivante :

```
AddDefaultCharset UTF-8
```

- pour PHP : dans le fichier php.ini, pour passer par exemple en utf-8, il faut ajouter (ou modifier) les lignes suivantes :

```
default_charset = "UTF-8"
mbstring.language = UTF-8
mbstring.internal_encoding = UTF-8
mbstring.http_input = UTF-8
mbstring.http_output = UTF-8
mbstring.detect_order = auto
iconv.input_encoding = UTF-8
iconv.internal_encoding = UTF-8
iconv.output_encoding = UTF-8
```

Conclusion sur ces encodages : si vous développez en PHP sur un serveur apache, partez du principe que le serveur est certainement configuré en iso-8859-1, ce qui ne vous empêche pas d'ajouter dans le code la balise meta de l'encodage.

Donc, dans la page index.html, modifiez la balise meta http-equiv en précisant le charset iso-8859-1 (ou ajoutez la balise, si elle n'est pas présente).

### Les scripts

La balise <script> permet d'intégrer du code client, souvent du JavaScript, sous forme de fonctions qui pourront être sollicitées dans le reste de la page. Ce code est

exécuté directement par le navigateur. La balise contient un attribut qui précise le type de langage utilisé. On y reviendra en détail à travers la séquence sur le JavaScript.

### Les link

Il est possible de lier des fichiers à la page active. Ces fichiers peuvent être de différentes natures : un fichier css (feuille de style), un flux rss, un fichier icône pour l'affichage dans la barre de titre...

```
<link rel="stylesheet" type="text/css" href="monstyle.css"
/>

<link rel="alternate" type="application/rss+xml"
title="news de ma boutique" href="rss.xml" />

<link rel="shortcut icon" type="image/x-icon"
href="monIcône.ico" />
```

Parmi les fichiers récupérés, copiez le dossier "images" dans le dossier du projet. Ce dossier contient toutes les images nécessaires au site. Dans ce dossier, vous pouvez entre autre remarquer le fichier boutiqueIcône.ico qui va servir d'icône pour le site.

Pour le moment, nous n'avons pas de feuille de style ni de fichier rss, mais on va au moins tester l'icône : dans le fichier index.html, ajoutez au moins le lien pour l'icône (donc la dernière ligne de code mentionnée ci-dessus, sans oublier de mettre le bon nom de fichier et son chemin : allez voir dans le dossier images).



Faites un test pour contrôler que l'icône apparaît bien dans la barre d'adresse du navigateur (c'est un petit t-shirt bleu).

## 4. Les balises simples

Le HTML propose de nombreuses balises. Sans toutes les aborder, nous allons voir dans un premier temps plusieurs balises simples, et en profiter pour commencer à construire le début de la page d'index. La présentation de la page sera fortement modifiée par la suite pour prendre en compte ce que l'on apprendra.

### Les balises de présentation de texte

Le texte peut être écrit simplement ou entouré d'une ou plusieurs balises de présentation. Voici quelques exemples :

```
<h1>titre niveau 1</h1>
<h2>titre niveau 2</h2>
...
<h6>titre niveau 6</h6>
```

gros titre

petit titre

```

<strong>texte épais</strong>

<i>texte en italique</i>

<center>texte centré dans la ligne</center>

<p>paragraphe</p>

<font color="red">texte en couleur rouge</font>

```

Il est bien sûr possible de combiner plusieurs balises, toujours en respectant la règle d'emboîtement :

```

<font color="red"><strong><i>texte en rouge, gras et
italique</i></strong></font>

```

Ces formatages de texte directement dans le code html, même s'ils existent, seront par la suite remplacés par un formatage séparé du code, en utilisant des styles.

Pour finir, voici une autre balise un peu particulière mais très pratique, qui permet juste d'ajouter un retour à la ligne :

```

<br />

```

Attention, quand vous êtes dans le code de la page, le fait d'aller à la ligne pour saisir du texte ne fera pas aller à la ligne dans l'affichage sous le navigateur. La balise br est donc importante pour marquer ce retour à la ligne. Cependant, le retour à la ligne est automatique après certaines balises (comme les balises de titre).

## Les balises de listes

Il existe des listes à puce. Voici un exemple de syntaxe :

```

<ul>
  <li>(....)</li>
  <li>(....)</li>
  <li>
    <ul>
      <li>(....)</li>
      <li>(....)</li>
      <li>(....)</li>
    </ul>
  </li>
  <li>(....)</li>
</ul>

```

Ces listes sont paramétrables (type de puce...).

Il existe aussi des listes numérotées, des listes de définitions... Inutile de rentrer dans le détail : les syntaxes sont faciles à trouver sur internet.




Dans la page index.html, utilisez les balises précédentes pour obtenir l'affichage suivant (le titre est en h1) :



Faites un test pour contrôler que l'exécution de votre code correspond bien à l'écran présenté.

## Les images

Vous allez souvent avoir besoin d'intégrer des images dans vos pages. Voici la balise correspondante :

```
 
```

L'attribut src permet de préciser le fichier image, éventuellement avec son chemin. La balise alt va afficher une info bulle lorsque la souris passe sur l'image (ou afficher l'information dans le cadre vide si le navigateur bloque l'affichage de l'image). L'attribut alt est maintenant obligatoire. Il est aussi très important pour le référencement.

Il existe plusieurs autres attributs facultatifs, comme la hauteur, largeur, bordure...

Dans index.html, juste à gauche du titre (dans la balise h1), ajoutez l'image logo.jpg qui se trouve dans le dossier "images" ("images/logo.jpg") et mettez en description "logo du site".



Faites un test pour contrôler si l'image s'affiche bien.

## Les objets

Il est possible d'insérer des objets multimédia (vidéo) ou de type application (application java par exemple, mais pas une applet qui possède une balise particulière) avec la balise object.

## 5. Les liens et objets réactifs

Une des possibilités les plus importantes du HTML est la capacité à rediriger vers une autre page ou une autre zone de la même page à travers les liens qui peuvent être placés sur du texte, une image ou d'autres éléments réactifs.

### Lien dans la page

Un lien dans la page consiste à placer un lien qui permet de rediriger vers un autre endroit de la page. Ceci est utile pour les pages un peu longues (ce qui est d'ailleurs déconseillé). Le principe est le suivant, avec une balise <a> (ancrage) et l'attribut name pour nommer la balise, il faut placer un point d'ancrage à l'endroit où l'on veut être repositionné, puis avec toujours une balise <a> mais l'attribut href, il suffit de

donner le nom de la balise d'ancrage pour le repositionnement. Voici un exemple de ces deux balises :

```
<a href="#uneMarque">positionnement sur la marque</a>
...
<a name="uneMArque"></a>
```

Le texte "positionnement sur une marque" va apparaître en souligné et d'une autre couleur. En plaçant la souris sur le texte, celle-ci change de forme (normalement en prenant la forme d'une main) et il est possible de cliquer sur le texte.

### Lien vers une autre page

Cette fois la balise permet d'accéder directement à une autre page. Donc seule la balise `<a>` avec l'attribut `href` est utilisée. Il n'est plus nécessaire de placer une ancre puisque le nom donné dans l'attribut `href` est directement le nom de la page destination. Voici la syntaxe :

```
<a href="nouvellepape.html">texte du lien</a>
```

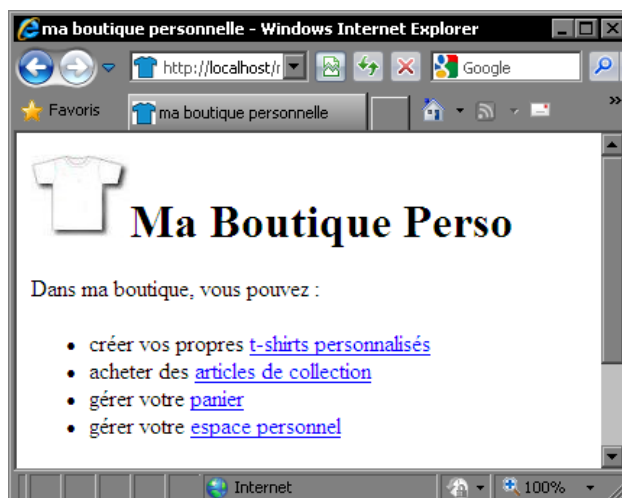
Dans ce cas, une nouvelle page du navigateur s'ouvre pour recevoir le lien. Si vous ajoutez l'attribut `target`, vous pouvez préciser dans quelle page doit s'ouvrir le lien. L'attribut `target` sera vu plus en détail, plus loin, dans la partie "frames".

Pour gérer des liens dans notre page index, on va commencer par créer la base des futures pages correspondantes : créez les pages `tshirt.html`, `boutique.html`, `panier.html`, `perso.html`. Dans chaque page, modifiez les lignes `doctype` et toutes les lignes de l'entête pour qu'elles soient identiques à celles de `index.html`. On verra comment éviter ces répétitions plus tard. Dans le body de chaque page, mettez juste un commentaire rappelant le nom de la page (pour que l'on puisse facilement repérer quelle page vient d'être ouverte).

Dans la page `index.html`, mettez les extraits de texte suivants en lien vers les pages correspondantes : "t-shirts personnalisés", "articles de collection", "panier", "espace personnel".



Faites un test pour contrôler que les liens fonctionnent bien (vous obtenez la bonne page) et vérifiez que l'affichage de la page d'index ressemble à ceci :



Les liens peuvent aussi être placés autour des images. Nous verrons plus loin une autre méthode pour rendre une image réactive.

### Lien vers un autre site

Avec le même principe que le lien vers une autre page, il est possible de faire un lien vers un autre site. Cependant, le chemin donné doit être complet, protocole compris. Voici un exemple :

```
<a href="http://www.nomServeur.com">texte du lien</a>
```

Vous pouvez être plus précis et accéder directement à une page précise d'un site, tout simplement en complétant l'adresse.

Dans votre page index.html, on va ajouter des liens vers d'autres sites. Normalement les liens sont en relation avec le contenu du site, sauf si ces liens risquent de faire perdre des internautes ! Dans le cas d'un site boutique, on va donc éviter les liens vers des sites commerciaux. Prenez 2 sites que vous voulez pour faire vos tests : moi j'ai pris le site de météo France et le site de l'horloge parlante... Modifiez la page pour qu'elle prenne en compte les 2 liens externes et donc qu'elle ressemble à ceci :



Le titre "Mes Liens Favoris : " est en h3 et en gras.



Faites un test pour contrôler que l'affichage correspond bien à ce qui est attendu et que les liens fonctionnent (redirection vers les bonnes pages).

### **Liens vers un objet**

Toujours avec la même balise, il est possible de gérer un lien vers un objet, par exemple un fichier. Si le système reconnaît l'extension, il ouvrira l'application concernée. Dans le cas d'objet exe ou compressé, celui-ci sera proposé au téléchargement.

### **Zone réactive**

Il est possible aussi de créer des zones réactives (rectangulaires, circulaires ou même de forme quelconque) avec la balise area. Cependant ce principe est de moins en moins utilisé et pose quelques problèmes de positionnement. Nous éviterons de le mettre en œuvre.

Le site va subir de nombreuses évolutions. Comme vous le retrouverez dans la correction officielle, vous allez faire des sauvegardes intermédiaires des différentes versions du site. Dans votre dossier de projet, créez un dossier "versions" et un sous-dossier "version 3 HTML". Faites une copie de tous les fichiers en racine du projet (donc toutes les pages) dans ce sous-dossier. Le numéro de version correspond à la séquence : ce sera plus pratique.



## Synthèse

Balise html :

```
<nombalise attributs> contenu </nombalise>
```

```
<nombalise attributs />
```

Structure d'une page html :

```
<html>
```

```
  <head>
```

```
    <title>le titre de la page</title>
```

```
  </head>
```

```
  <body>
```

```
    Corps de la page avec balises d'affichage
```

```
  </body>
```

```
</html>
```

Respect du xhtml :

Modifier la balise html pour préciser la norme à respecter. Par exemple

```
<html xmlns=http://www.w3.org/1999/xhtml xml:lang="fr">
```

Respect de la norme W3C :

Ajouter en début de page la balise doctype pour que le code de la page respecte la norme W3C. Par exemple

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd>
```

Balises meta :

Elles apportent des informations sur la page mais n'affichent rien. Elles sont utilisées pour le référencement.

Balises link :

Elles permettent de gérer des liens : vers un icône, un fichier rss, une feuille de style...

## Séquence 4

# Travail sur l'interface

Cette séquence aborde l'aspect "présentation" du site en faisant le tour des différents choix possibles pour organiser les informations sur une page. L'intérêt et le principe des feuilles de styles sont expliqués en détail.

### ► Capacités attendues en début de séquence

Avoir acquis les notions de base de la création d'un site et du HTML abordées dans les séquences précédentes.

### ► Contenu

1. Introduction .....	36
2. Les frames et iframes .....	36
3. Les tableaux .....	39
4. Les calques .....	41
5. Les styles : positionnement .....	43
6. Les styles : définition .....	45
7. Les images .....	47
8. Le travail de l'infographiste .....	48

## 1. Introduction

Après le début de la création de cette première page, un constat s'impose : c'est moche. Au-delà du manque de recherche graphique, on sent bien un premier problème apparaître : le positionnement des différents éléments de la page. Pour le moment, les balises manipulées n'ont permis que de placer les informations linéairement ou les unes en dessous des autres : impossible de positionner le texte et les images à des emplacements spécifiques, en fonction de coordonnées, par exemple.

Il manque donc des outils pour positionner les informations. Il existe en fait plusieurs méthodes. Nous allons étudier les 3 plus importantes méthodes, en commençant par la plus obsolète (les frames) pour aller vers la plus efficace (les calques).

Ensuite, nous aborderons les styles qui apportent une palette de possibilités de présentation mais aussi de formatage de pages.

Enfin, certains points sont à approfondir concernant les images, en particulier pour minimiser leur temps de chargement.

Tous ces éléments se combinent avec le travail de l'infographiste.

## 2. Les frames et iframes

Considérées comme obsolètes et abandonnées à cause d'un mauvais référencement, les frames ont été utilisées pendant un certain temps pour palier au problème du temps de rechargement des pages. Avec les calques et Ajax que l'on étudiera plus loin, les frames n'ont plus de raison d'être et n'apportent que des inconvénients.

Vous vous posez alors la question : pourquoi les étudier ? Pour que vous ayez vu au moins une fois à quoi cela ressemble, et pour que vous sachiez lire un code existant qui contient des frames.

### Les frames

Diviser une page en frames revient à créer une page qui contient plusieurs "sous-pages" indépendantes. Du coup, on peut laisser certaines zones (frames) fixes pendant qu'une autre est rechargée.

Voici la syntaxe des balises des frames :

```
<frameset cols="val11, ..., val1N" rows="val21, ..., val2N">
```

```
    Déclaration des frames
```

```
<noframes>
```

```
    Contenu si le navigateur n'accepte pas les frames
```

```
</noframes>
```

```
</frameset>
```

La partie `<noframe>` est optionnelle, mais intéressante pour éviter une absence d'affichage dans le cas où le navigateur refuse d'exécuter les frames.

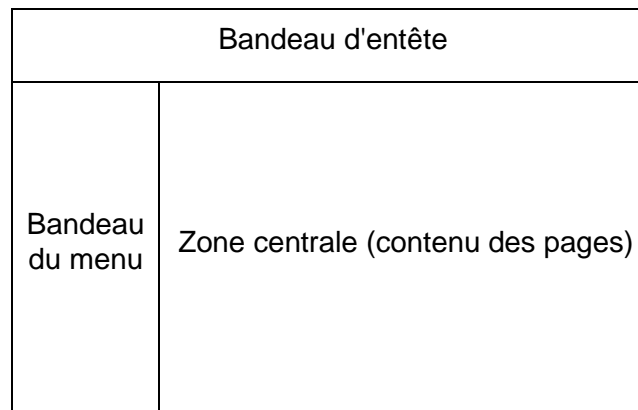
Attention, les balises `<frameset>` et `</frameset>` se substituent aux balises `<body>` et `</body>`.

Les attributs `cols` et `rows` sont optionnels : l'un, l'autre ou les 2 peuvent être omis. L'attribut `cols` permet de définir la taille de chaque colonne de division. L'attribut `rows` fait la même chose, mais en ligne. Chacun de ces attributs peut contenir de 1 à N valeurs. Chaque valeur représente la taille soit en fixe (ex : 70 pixels), soit en pourcentage par rapport à la taille du navigateur (ex : 10%), soit une étoile pour signifier de prendre la taille qui reste (\*).

Chaque balise `<frameset>` doit contenir autant de balises `<frame>` que de zones définies par les lignes et colonnes. Voici la structure d'une balise `<frame>` :

```
<frame src="pageAaficher" name="nomDeLaFrame" />
```

Pour obtenir une division plus complexe, il est possible d'imbriquer les balises `frameset`. La division la plus courante consiste à avoir un bandeau d'entête (donc une ligne contenant une seule colonne en haut de page), puis un bandeau vertical sur la gauche pour le menu, et une zone centrale plus grande pour le contenu des pages. Cela donne globalement cet aspect :



Il existe bien sûr des tas d'autres variantes possibles. Voici un exemple de code associé à cette présentation :

```
<frameset rows="110,*">
  <frame src="pageTop.html" name="topFrame" />
  <frameset cols="80,*">
    <frame src="pageLeft.html" name="leftFrame" />
    <frame src="pageMain.html" name="mainFrame" />
  </frameset>
</frameset>
<noframes>
  <body>
```

```

        Le site ne peut s'afficher sur ce navigateur

    </body>

</noframes>

</frameset>

```

L'attribut src permet de définir la première page qui doit s'afficher dans la frame. L'attribut name peut contenir le nom de votre choix : très souvent, on utilise les noms donnés dans cet exemple pour ce type de découpage.

Même si au départ une page est associée à une frame, par la suite il est possible de changer cette page. Par exemple, si vous ajoutez un lien sur un texte, il suffit d'ajouter l'attribut target pour rediriger vers une des frames de la page.

```

<a href="pageAafficher" target="nomFrame">texte du
lien</a>

```

Maintenant que vous savez comment construire les frames, modifiez votre page afin de mettre le logo et le titre dans le bandeau du haut, le menu dans le bandeau de gauche et faites en sorte qu'à chaque clic sur une option du menu, on obtienne



dans la frame centrale la page correspondante. Au chargement, la frame centrale doit afficher ce que la page index affichait au départ, excepté le logo et le titre. Après modification, vous devez donc obtenir un affichage qui doit ressembler à ceci :

Attention, pour que les liens de gauche fonctionnent en changeant le contenu de la frame centrale, il ne faut pas oublier l'attribut target dans la balise de lien. Ce n'est

par contre pas obligatoire pour les liens dans la frame centrale puisque le lien amène vers la même frame.



Faites un test pour contrôler que l'affichage correspond bien à ce qui est attendu et que tous les liens fonctionnent, aussi bien les liens du menu que les liens dans la page centrale (changement du contenu de la frame centrale).

Il existe des attributs sur les balises frameset et frame qui permettent des tas de choses, comme enlever le trait de séparation, réduire les espaces entre frames, enlever les ascenseurs, etc... Mais on ne va pas s'attarder sur ces options puisque de toute façon ce n'est pas la solution des frames que l'on va retenir pour réaliser le site.

Comme vous avez créé un dossier "version 3 HTML", créez un dossier "version 4.2 Frames" et sauvez dans ce dossier toutes les pages de votre nouveau site.

### Les iframes

Une iframe est une frame un peu particulière : l'idée n'est pas de diviser la page complète en différentes zones, mais juste de prévoir une zone de la page comme étant une frame. La page garde dans ce cas une structure normale (body) et contient juste une zone indépendante, possédant une taille bien définie. Du coup, dans cette zone, il sera possible de charger une page indépendante. Cette option est intéressante en particulier pour charger un objet particulier dans une zone, par exemple pour ouvrir un document PDF sans prendre la page complète.

Voici la syntaxe de l'iframe :

```
<iframe src="page.html" name="nomIframe"></iframe>
```

Plusieurs attributs sont possibles, comme width, height, scrolling (pour accepter ou non l'ascenseur), etc...

## 3. Les tableaux

Dès les premiers sites, les tableaux se sont révélés être intéressants pour positionner les informations sur une page. Le principe consiste à créer des cellules, sachant que chaque cellule peut contenir aussi des tableaux. A l'intérieur d'une cellule, l'information peut subir un alignement horizontal et vertical. Les cellules peuvent avoir une taille fixe ou variable. Toutes ces possibilités permettent de faire quasiment tout au niveau présentation. Cependant, le code peut s'avérer rapidement complexe, avec de nombreuses imbrications qui finissent par être illisibles.

Voici un exemple de syntaxe de tableau :

```
<table width="200" border="0">
  <tr align="left" height="30">
    <td valign="center" width="20">
      Contenu 1ere case de la 1ere ligne
```

```

        </td>

        ...

        <td valign="top" width="180">

            Contenu Neme case de la 1ere ligne

        </td>
    </tr>

    <tr>

        ...

    </tr>
</table>

```

La balise `<table>` permet de définir un tableau. Elle peut contenir différentes balises dont la balise `<tr>` qui définit des lignes. Les balises `<tr>` peuvent aussi contenir des balises `<td>` pour définir des cellules. Chacune de ces balises a des attributs optionnels, comme montré dans l'exemple : `valign` (alignement vertical), `align` (alignement horizontal), `width` (largeur, en pixel ou pourcentage), `height` (hauteur), etc... Attention, si vous définissez une cellule d'une certaine largeur, toutes les cellules de la même colonne seront de la même taille : dans le cas où des tailles différentes seraient définies, c'est normalement la plus grande qui est prise en compte. Il arrive que certaines cellules soient vides : dans ce cas, il faut les remplir avec un espace forcé (**&nbsp;**).

En racine de votre projet, supprimez les pages actuelles (de la version frames que vous avez sauvé dans "version 4.2 Frames") et placez une copie des pages de la première version ("version 3 HTML"). Ce sera plus facile de partir de cette version pour faire les tableaux. Le but est de créer dans la page index un tableau de 2 lignes et 3 colonnes afin de disposer les informations pour obtenir ce résultat :



Inutile de préciser les largeurs pour chaque cellule : la largeur d'une cellule par colonne suffit. Définissez la première colonne de largeur 100px et la seconde de largeur 50%. Ne définissez pas de largeur pour la 3eme colonne.



Faites un test pour contrôler que l'affichage correspond bien à ce qui est attendu. Modifiez la taille du navigateur pour voir comment s'adapte la taille des cellules du tableau.

Avec ce système de présentation, remarquez la position du titre par rapport à l'image : le titre est verticalement aligné sur le haut de l'image (si vous avez utilisé l'attribut valign à top), alors que jusqu'à maintenant, il était aligné sur le bas de l'image.

Créez un dossier "version 4.3 tableaux" et enregistrez vos pages actuelles dans ce dossier.

Même s'ils sont plus adaptés que les frames, les tableaux ne représentent pas une bonne solution pour présenter une page complète. Ils peuvent s'avérer ponctuellement intéressants pour l'alignement d'informations dans des petits tableaux. Vous les utiliserez plus loin.

## 4. Les calques

Les calques représentent la méthode la plus optimisée pour gérer la présentation d'une page. Un calque se positionne à des coordonnées précises ou relatives par rapport à un autre calque. De nombreuses options sont possibles sur les calques. Nous verrons d'ailleurs plus loin qu'il est même possible de rendre les calques réactifs pour que l'utilisateur puisse les redimensionner ou les déplacer.

La balise <div> permet de créer un nouveau calque. Son contenu sera affiché dans le calque et ses attributs permettent de déterminer les caractéristiques du calque (position, taille, couleur ou image de fond, alignement intérieur...). Un calque peut tout contenir : informations, tableau, autre calque... Voici une base de syntaxe du calque :

```
<div id="idCalque" style="width:200px; height:100px;
overflow:auto; left:20px; top:20px; position:absolute;
```



```
background-image: url(ficImage);
background-color: red; ">
    Contenu
</div>
```

Vous trouvez ici les premiers attributs fondamentaux d'une balise div, sachant que tous ces attributs sont optionnels. L'id permet de donner un identifiant pour ensuite accéder à la balise et modifier son contenu. L'attribut style contient de nombreuses options pour définir la taille (width, height) et la position (left, top) mais aussi son image ou couleur de fond. La position en "absolute" permet de positionner le calque en fonction de son conteneur et non en fonction du calque précédent. L'overflow permet de définir comment va être géré l'affichage du contenu (auto = ascenseurs si nécessaire ; none = refus d'ascenseurs...).

Au-delà de ces attributs déjà bien pratiques, il existe plusieurs autres possibilités pour déplacer, redimensionner ou rendre invisible le calque : ces actions pourront être directement réalisées par l'utilisateur. Nous étudierons ces possibilités plus tard.

Pour travailler, reprenez dans le dossier de projet, les fichiers de la première version ("version 3 HTML"). A partir de là, créez 4 calques (id : divLogo, divTitre, divPresentation, divLiens) avec respectivement les coordonnées suivantes (left, top, width, height) :

divLogo(0, 0, 72, 61), divTitre(40, 30, 300, 80), divPresentation(40, 100, 300, 200), divLiens(350, 180, 250, 120).

Petite remarque sur le choix des noms des id : prenez l'habitude de nommer les id de façon à repérer le type de la balise avec les 3 premières lettres, puis un nom parlant, commençant par une majuscule. Cette règle est classique.

Donnez les autres caractéristiques pour obtenir un affichage proche de celui-ci :



Pour les couleurs des calques, j'ai utilisé lightblue et lightgrey.



Faites un test pour contrôler que l'affichage correspond bien à ce qui est attendu.

Remarquez le chevauchement entre le calque du titre et le calque du logo : ceci n'est pas possible avec des tableaux. Il est possible de définir l'ordre de superposition avec la propriété de style z-index. Nous reviendrons plus tard sur ces détails de styles.

Vous avez compris qu'avec un tel outil, on peut faire à peu près tout ce que l'on veut au niveau positionnement. Enregistrez cette version dans un dossier "version 4.4 calques".

### **Le positionnement pour les autres balises**

En réalité, la balise <div> n'a pas l'exclusivité du positionnement. En ajoutant à n'importe quelle balise des informations de positionnement, vous pouvez placer la balise où vous voulez ! Mais alors pourquoi parler plus spécifiquement de calques ? Parce que la balise <div> est un conteneur, donc elle peut contenir des informations et d'autres balises. Du coup son positionnement devient pertinent et structuré, plutôt que de gérer des positionnements pour chaque balise séparée. Il est aussi plus pratique d'affecter des styles qui seront spécifiques à toutes les balises <div>.

## **5. Les styles : positionnement**

Définir un style consiste à regrouper un ensemble de caractéristiques sous un même nom afin de pouvoir l'appliquer autant de fois que nécessaire sur les balises qui acceptent les styles.

Par exemple, si vous voulez que certains titres soient tous de taille 8, en gras, police verdana et en bleu, vous pouvez définir un style comportant toutes ces caractéristiques et ainsi l'appliquer à tous les titres concernés.

### **Style directement dans une balise**

Cette possibilité a déjà été vue dans les calques : il suffit d'ajouter l'attribut style dans la balise concernée, et mettre entre guillemets toutes les caractéristiques que l'on désire appliquer à cette balise. Cette option est à éviter pour 2 raisons : d'abord il risque d'y avoir répétition de style si les caractéristiques sont identiques entre plusieurs balises, ensuite il faut au maximum éviter de mélanger information et présentation afin de clarifier au maximum le code.

### **Style directement dans une page**

Avec ce système, on résout le problème de la répétition mais uniquement dans une page. Si vous avez besoin du même style dans une autre page, il faudra alors le redéfinir.

Voici comment définir un style dans une page : il faut utiliser la balise <style> qui doit être placée dans le head de la page.

```
<style type="text/css">
```

```
.unStyle {
    propriete1 : valeur1 ;
    propriete2 : valeur2 ;
    ...
}
</style>
```

La propriété est par exemple "width" et sa valeur "200px". Vous avez compris qu'il est possible de donner plusieurs caractéristiques à un même style. Vous verrez un peu plus loin comment utiliser un style ainsi créé.

### Style dans une feuille de style externe

Pour éviter de mélanger la description des styles avec le code de la page, et pour une réutilisation plus performante, il est préférable de stocker les styles dans un fichier séparé. C'est ce qu'on appelle la feuille de style.

La feuille de style est un fichier css qui contient tout simplement un ensemble de styles. Lorsque vous rattachez une feuille de style à une page, vous pouvez ensuite utiliser dans la page tous les styles contenus dans la feuille de style.

#### Créer une feuille de style

Pour créer une feuille de style, il faut donc juste créer un fichier avec l'extension css et insérer dans le fichier la description de tous les styles que l'on veut utiliser. La syntaxe de la description est la même que celle qui a été présentée juste au dessus.

Nous allons par la suite voir aussi comment gérer plusieurs feuilles de style pour bien organiser les informations. Pour le moment, dans le dossier de votre projet, créez un dossier css et dans ce sous-dossier, créez le fichier principal.css qui sera pour le moment vide.

#### Attacher une feuille de style

Comment attacher une feuille de style à une page ? En ajoutant dans les balises <head> la balise suivante (que nous avons déjà vu plus haut) :

```
<link rel="stylesheet" type="text/css" href="monstyle.css"
/>
```

Dans la page index, ajoutez la ligne pour insérer la feuille css/principal.css.

#### Utiliser un style de la feuille de style

Le principe est le même que pour utiliser un style de la page : il suffit d'intégrer le style dans la balise concernée.

## 6. Les styles : définition

Nous savons maintenant où placer les styles et comment les utiliser. Voyons plus en détail les différentes catégories de styles et comment les définir.

### Sélecteur simple

Vous pouvez affecter un style directement à un type de balise. Vous n'aurez alors même pas à affecter le style lors de l'utilisation de la balise. Par exemple, si vous définissez le style suivant :

```
P {
    text-align : justify ;
    color : grey ;
}
```

Le nom du style est le nom de la balise concernée

Là, vous affectez à toutes les balises <p> le style justifié et texte en gris.

Pour affecter un style à un type de balise, il suffit de mettre comme nom de style directement le nom de la balise.

Dans principal.css, créez un style pour toutes les div afin de mettre les 2 paramètres qui sont communs à vos divs : position absolute et overflow auto. Enlevez ces informations des styles des 4 balises div dans la feuille index.

### Style multi-balises

Cette fois, vous définissez un style indépendant, et c'est au moment de son utilisation que vous préciserez l'appel de ce style dans l'entête de la balise concernée. Un style de ce type peut être affecté à n'importe quelle balise. Voici un exemple de définition de ce type de style :

```
.texteSimple {
    text-align : justify ;
    color : grey ;
}
```

Le nom du style est libre mais doit OBLIGATOIREMENT commencer par un point

Voici comment, dans le code, affecter ce type de style à une balise :

```
<uneBalise class="texteSimple" ...>
```

C'est avec l'attribut class que vous allez pouvoir affecter ce type de style à une balise. Ceci est valable pour n'importe quel type de balise. Il est possible d'affecter plusieurs styles à une même balise en mettant les noms des styles dans les guillemets de l'attribut class, juste en les séparant par un espace.

### Style affectable à un seul type de balise

Il est possible de restreindre un style multi-balises à un seul type de balise. Quel est l'intérêt par rapport au premier type de style précédemment vu, qui modifiait les caractéristiques de toutes les balises d'un même type ? Justement, ce nouveau style n'affecte pas toutes les balises d'un même type mais seulement certaines, celles à qui on affectera le style. Voici un exemple de définition :

```
div.texteSimple {
    background-color : lightgrey ;
}
```

A la suite de cette définition de style, si vous affectez le style "texteSimple" à une balise <div>, alors le style influencera la balise. Si vous affectez ce même style à un autre type de balise, cela n'aura aucune incidence.

### Style attaché à un id

Il est possible de déclarer un style spécifique à un id, donc à une seule balise de la page, possédant cet id (sachant qu'il ne peut pas y avoir 2 balises avec le même id dans une même page). Cependant, au moment de la définition du style, il n'est pas interdit de regrouper plusieurs id. Voici un exemple de définition :

```
#divPresentation, #divTitre {
    position : absolute ;
    left : 10px ;
}
```

Le nom du style correspond au nom d'id et doit OBLIGATOIREMENT commencer par un #

Ici, l'exemple se base sur les calques qui ont été définis précédemment, avec les id divPresentation et divTitre. Les id étant directement précisés, le style va s'affecter automatiquement, sans avoir à le préciser.

Dans principal.css, créez 4 styles spécifiques aux 4 id des calques, en insérant les paramètres restants. Utilisez text-align pour mettre le texte centré dans le dernier div. Dans la page index, enlevez les styles des 4 calques.



Faites un test pour contrôler que rien n'a changé. Si les alignements ne sont pas corrects, contrôlez que vous avez mis les bonnes valeurs et que vous avez bien lié la feuille de style à votre page.

Remarquez que votre page html est plus légère et que tout doucement il y a séparation de la présentation et du contenu.

Enregistrez cette nouvelle version (sans oublier le dossier css) dans le dossier "version 4.6 styles".

### Modification d'un style prédéfini

Certaines balises possèdent des styles prédéfinis. Par exemple la balise <a> (pour les liens) possède 3 styles prédéfinis : link (à visiter donc non encore utilisé), active (lors du clic de la souris ou lorsque le nom de la page = le lien de la page), visited (déjà utilisé). Voici comment modifier un style prédéfini (si les caractéristiques de départ ne vous conviennent pas) :

```
a:link {
    color : blue ;
    font-style : italic ;
```

Le nom du style correspond au nom de la balise, suivi de ":" puis du nom du style prédéfini

```
    }  
}
```

Encore une fois, ce type de style va être automatiquement affecté.

### Petite remarque sur les couleurs

Les couleurs peuvent se coder de différentes façons :

- par leur nom : red, blue, yellow...
- par leur code hexa : #rrggbb (chaque partie représente un nombre codé en hexa)
- par leur valeur décimale : rgb(val1, val2, val3) où val est compris entre 0 et 255

### Style du curseur

Beaucoup de choses peuvent être gérées avec les styles : on y reviendra au fur et à mesure de la construction du site. Mais pour le plaisir, voyons un détail bien pratique : le changement de l'aspect du curseur. Il suffit d'appliquer à la propriété cursor une des valeurs suivantes :

- auto : apparence par défaut adaptée au contexte
- default : apparence par défaut (flèche)
- crosshair : la croix
- pointer : apparence indiquant un lien (doigt pointé)
- move : élément en cours de déplacement
- texte : apparence pour une zone de texte ( | )
- help : apparence pour l'accès à l'aide ( ? )

On exploitera ces possibilités par la suite.

### Commentaires

Dans une feuille de style, il est aussi possible d'ajouter des commentaires :

```
/* ceci est un commentaire */
```

## 7. Les images

Le problème des images sur le net est le temps de téléchargement. Plus une image est volumineuse, plus elle met du temps à être transférée du serveur. Cependant on ne peut pas se passer d'images pour la présentation d'un site. Voici donc quelques conseils.

### Choisir le bon format d'image

Évitez le format bmp : beaucoup trop lourd. Le format jpg est à favoriser pour les images de type photo. Le format png est à favoriser pour les images de type schéma ou comportant de larges zones de couleurs identiques et peu de couleurs. Il y a aussi le format gif cependant évitez les gif animés : d'abord parce qu'il y a plusieurs images à télécharger, ensuite parce qu'un gif animé fait généralement "gadget" et donc pas professionnel.

**Réduire la taille des images**

Ne réduisez pas une image une fois téléchargée (ce qui est possible avec certains attributs de la balise `img`). Réduisez les images en amont, donc avant de les mettre sur le serveur pour qu'elles aient directement la taille désirée.

Pour réduire une image en cherchant un bon compromis entre qualité et poids, vous pouvez utiliser des logiciels spécialisés. Par exemple, Photoshop offre une option "enregistrer pour le web" qui présente plusieurs possibilités de réduction en montrant le résultat. On obtient ainsi des images de qualité acceptable pour le web et nettement plus légères.

**Réduire le nombre de téléchargement**

Si vous rechargez plusieurs fois une page, les images vont se recharger (sauf si elles sont dans le cache). On verra plus loin comment réduire le nombre de chargement des pages en utilisant de nouvelles technologies comme AJAX.

Tout ceci sera concrètement abordé par la suite.

## **8. Le travail de l'infographiste**

Même si les outils précités permettent de quasiment tout gérer, il faut bien prendre conscience que présenter des pages est un travail à part entière : celui de l'infographiste. Dans le cadre de ce cours, on va essayer de respecter quelques règles de présentation, mais c'est avant tout l'aspect technique et le code qui nous intéresse. Nous ne prétendons donc pas remplacer le travail de l'infographiste.

A notre niveau, une des bonnes méthodes pour présenter correctement un site est avant tout d'aller voir à quoi ressemblent les autres sites professionnels du même thème.

**Dessin de l'aspect final**

La première démarche de l'infographiste est de réfléchir sur l'aspect final de la page. Pour cela, il réalise un dessin (papier ou en utilisant un logiciel comme photoshop). Normalement, il réalise plusieurs solutions afin d'offrir un panel de propositions. Le demandeur du site va alors choisir celle qui lui convient le plus.

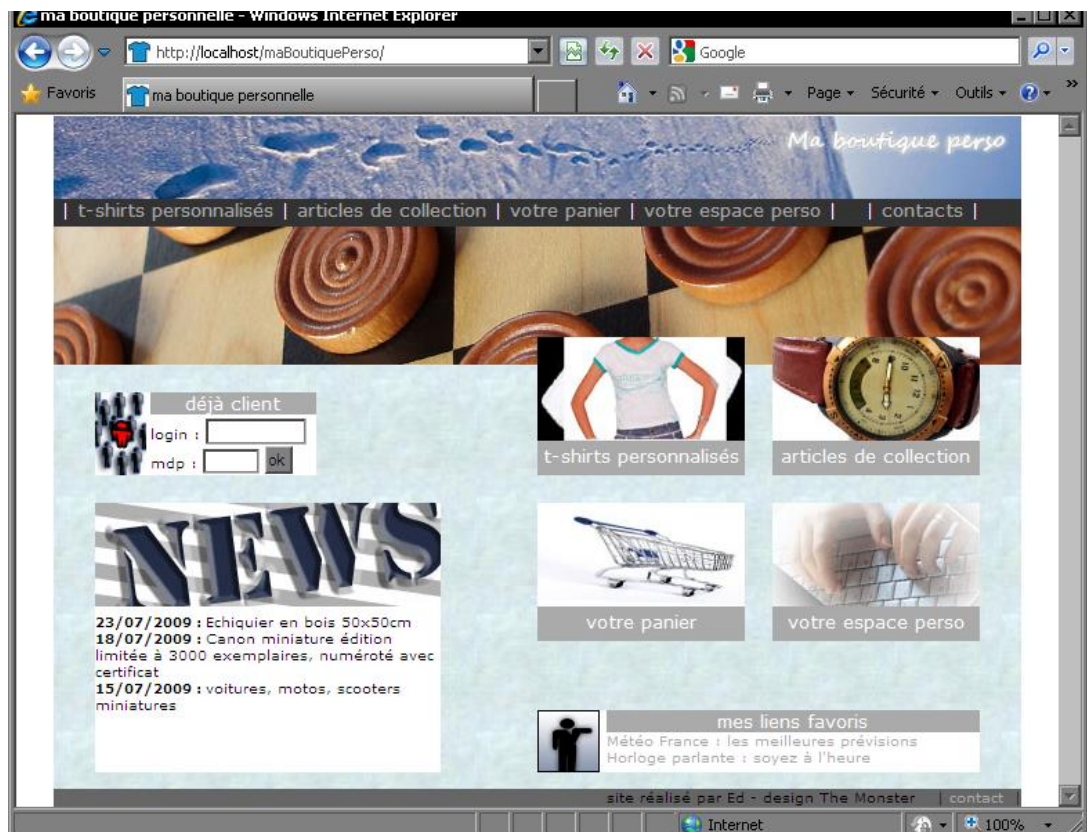
Reprenons notre site. Jusque là, la présentation était peu attirante. Entre les mains d'un infographiste, les choses vont changer. Ce dernier va mettre en avant plusieurs aspects :

- il faut avant tout réfléchir sur le positionnement des informations sur la page, afin de guider l'utilisateur vers certaines fonctionnalités. N'oublions pas que spontanément le regard de l'utilisateur part du coin haut gauche de l'écran et descend en diagonale vers le coin bas droit.
- Il faut ensuite se fixer une charte de couleurs : les couleurs doivent être harmonieuses et respecter le thème du site. Les couleurs trop contrastées sont à éviter ainsi que tout ce qui fait trop gadget (les images clignotantes ou animées, par exemple).
- Il faut enfin trouver les bonnes images et textures : un site doit être visuel et l'utilisateur doit comprendre qu'il va accéder à une fonctionnalité rien que par



l'image, sans même avoir besoin de lire de texte. Les images et textures vont donner aussi une ambiance générale au site. L'utilisateur doit trouver cette ambiance agréable.

Notre site vend des t-shirt personnalisés et des articles personnels divers, donc le site doit faire sentir ce thème de création et de vente tout en gardant un aspect professionnel. Voici une présentation que l'on pourrait imaginer :

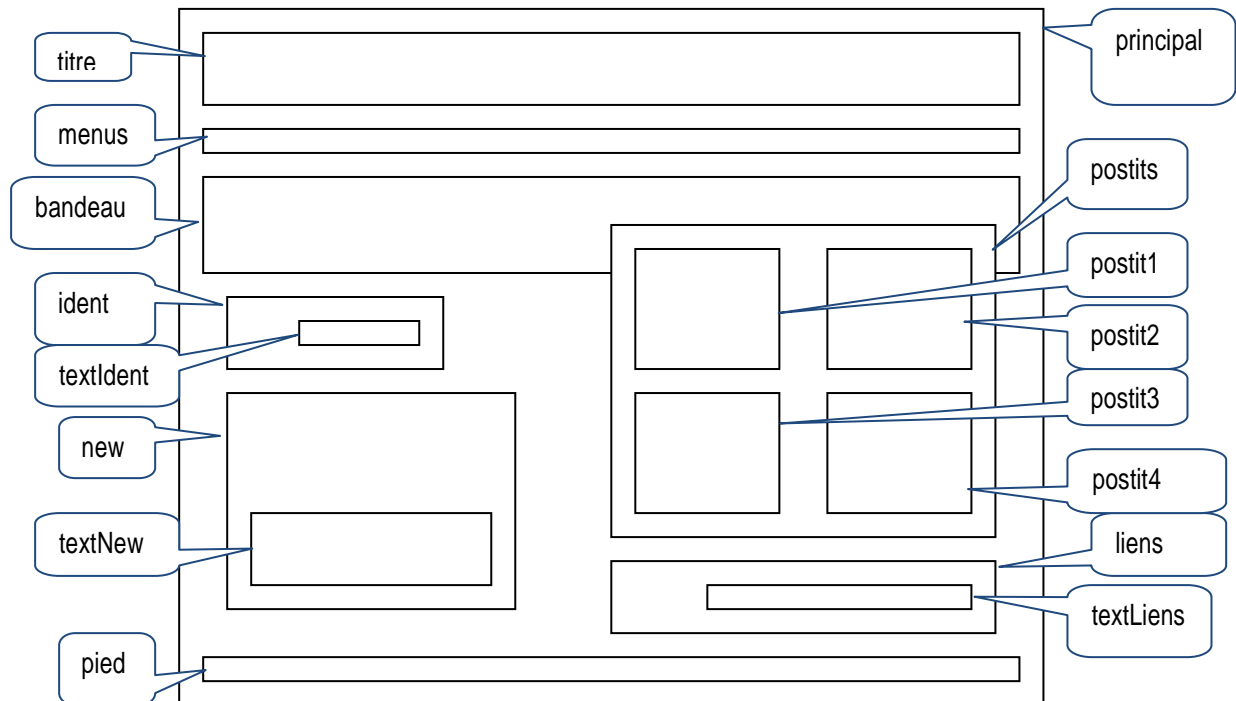


Cette fois, vous voyez l'aspect final de la page index, même si le code de cette page va encore évoluer dans les séquences suivantes. Plusieurs images vous ont été fournies et vont vous servir à gérer cette présentation. Vous remarquerez au passage le format et le poids des images : n'oubliez pas de faire attention à cet aspect lorsque vous aurez à créer un site.

### Détermination des calques

En observant l'écran, on peut facilement déterminer les calques qui seront nécessaires pour gérer cette présentation. Commençons donc par déterminer ces calques. Chaque calque doit être nommé (les noms ont été simplifiés : pensez à mettre div et la première lettre en majuscule pour le nom de chaque calque). Ne créez pas encore les calques dans votre page index : vous aurez plus d'indications par la suite.





Certains espaces ont été volontairement ajoutés pour que vous puissiez mieux repérer les différents calques et l'emboîtement entre certains calques.

### Création des calques dans le CSS

Dans la feuille de style, supprimez tout ce que vous avez déjà fait et créez les styles spécifiques à chaque calque. Les coordonnées entre parenthèses correspondent, dans l'ordre, à left, top, width, height. N'oubliez pas que les valeurs doivent être suivies de px, sauf pour les pourcentages. Le signe "-" indique qu'il ne faut pas utiliser cette propriété pour ce calque :

```
#divPrincipal(50%, 50%, 700, 500) ; #divTitre(0, 0, 100%, 60) ; #divMenus(0, 60, 100%, 20) ; #divBandeau(0, 80, 100%, 100) ; #divPostits(350, 160, 320, 220) ; #divPostit1(0, 0, -, -) ; #divPostit2(170, 0, -, -) ; #divPostit3(0, 120, -, -) ; #divPostit4(170, 120, -, -) ; #divIdent(30, 200, 160, 60) ; #divTextIdent(40, 0, 120, 100%) ; #divNews(30, 280, 250, 195) ; #divTextNews(0, 75, 100%, 120) ; #divLiens(350, 430, 320, 45) ; #divTextLiens(50, 0, 270, 100%) ; #divPied(0, 487, 100%, 13)
```

Voici les caractéristiques complémentaires de chaque calque (les valeurs à appliquer sont mises entre parenthèses) : complétez les styles correspondants.

#divPrincipal : margin-left(-350px), margin-top(-250px)\*

#divTextIdent : font-size(medium)

#divPied : font-size(x-small), text-align(right)

\*les propriétés margin-left et margin-top permettent de déterminer la marge gauche et haut du calque par rapport à la fenêtre (ou au calque conteneur). En mettant des marges négatives de la moitié de la taille du calque, puis en mettant à 50% de la

taille de la fenêtre les propriétés left et top, on utilise une petite astuce de calcul qui permet de centrer le calque dans la fenêtre, quelque soit la taille de la fenêtre. C'est ce que l'on veut pour le calque principal qui contient tous les autres.

Affectez les couleurs de fond aux calques correspondants :

```
#divMenus      (#333333),      #divIdent(#FFFFFF),      #divTextNews(#FFFFFF),  
#divLiens(#FFFFFF), #divPied(#666666)
```

Affectez aussi les images de fonds aux calques correspondants (attention, faites précéder les noms des fichiers par ../images/ car il faut remonter au dossier supérieur avec ".." puis aller dans le dossier "images" pour aller chercher les images) :

```
#divPrincipal(fond.jpg),      #divTitre(titre.jpg),      #divBandeau(bandeau0.jpg),  
#divIdent(ident.jpg en no-repeat), #divNews(news.jpg en no-repeat),  
#divLiens(lien.png en no-repeat)
```

Allez voir le contenu du fichier fond.jpg. Vous remarquerez qu'il représente une toute petite zone. Et pourtant il va servir pour remplir tout le fond du site. Comment est-ce possible ? Parce que par défaut, les images de fond sont répétées pour remplir tout le conteneur. C'est bien pratique pour remplir une zone avec une petite image qui ne pèse pas lourd. Bien sûr, cela n'est plus possible avec une image comme bandeau.jpg qui est une image complète. Quand "no-repeat" est précisé, c'est que justement on ne désire pas que l'image soit répétée : il faudra donc attribuer cette valeur à la propriété background-repeat.

### Création des styles généraux

En dehors des styles spécifiques à chaque calque, il va falloir déterminer des styles généraux et d'autres spécifiques à un groupe d'objets. Voici chaque style que vous allez devoir créer, avec leurs caractéristiques :

- un style pour les petits titres afin d'afficher des textes centrés, sur fond gris et en gras :  
.petitTitre : couleur de fond(#AAAAAA), font-weight(bold), width(100%), text-align(center)
- un style pour les petits textes en noir et plus petits caractères :  
.petitTexte : text-align(left), font-size(x-small), color(#000000)
- des caractéristiques communes à tous les calques (police verdana ou à défaut serif, texte en blanc et petit, calque en position absolue et sans ascenseur :  
div : overflow(hidden), position(absolute), font-family(verdana, serif), font-size(small), color(#FFFFFF)
- des caractéristiques communes aux 4 calques postit (donc un style postit utilisable que sur les calques) avec une taille fixe, gris foncé en couleur de fond, texte centré :  
div.divPostit : width(150px), height(100px), couleur de fond(#AAAAAA), text-align(center), cursor(pointer)
- un changement de caractéristiques par défaut pour les liens (plus de soulignement et une couleur gris clair)

a:link et a:visited : text-decoration(none), color(#AAAAAA)

Attention, sur ce dernier point, ne regroupez pas les 2 styles mais faites bien 2 styles différents : dans ce cas le regroupement n'est pas possible.

### Travail du développeur

Le travail de l'infographiste s'arrête là. C'est au tour du développeur de coder la page en utilisant le css. En réalité, les choses sont nettement moins "carrées". Une fois que l'infographiste a dessiné la page, le développeur code lui-même la feuille de style en passant des commandes précises à l'infographiste, pour que ce dernier réalise sous un logiciel de dessin les différentes images nécessaires qui seront intégrées dans la feuille de style et le site. Parfois à l'inverse, l'infographiste crée le css et même la structure de la page html, puis le développeur intervient ensuite pour insérer le code dynamique.

Dans la page index, supprimez le code qui se trouve dans les balises body et créez tous les calques nécessaires, sans vous tromper sur les imbrications entre les calques (par exemple, le calque d'id "principal" contient tous les autres calques). Voici quelques informations complémentaires pour vous aider à remplir les calques (pensez aussi à vous référer à la copie d'écran qui a été donnée plus haut) :

- Les balises divTitre et divBandeau sont vides.
- La balise divMenus contient un lien par option, séparés par des espaces (&nbsp;) et des caractères |.
- Il faut affecter aux 4 balises divPostit1 à divPostit4 le style "divPostit". Ces 4 calques contiennent à chaque fois une image avec la balise img (de menu1.jpg à menu4.jpg, sans oublier le chemin : images/ ) et le texte qui s'affichera sous l'image. Pour le moment, ces images et le texte en dessous ne seront pas réactifs : on s'en occupera plus tard. Vous devez vous poser la question : pourquoi des images dans la page alors qu'elles devraient apparaître dans la feuille de style ? Vous avez raison. Cependant je vous fais mettre ces images dans la page dans un simple aspect pédagogique, pour que vous appreniez tout de même à utiliser la balise image, que vous allez d'ailleurs réutiliser plus tard. Si vous aviez mis l'image dans le style, vous auriez alors soit du créer un nouveau sous calque, soit du intégrer cette image dans le fond de chaque balise postit en no-repeat.
- La balise divIdent contient la balise divTextIdent qui elle-même contient 2 balises (un div sans id portant le style petitTitre et contenant le texte "déjà client", un div sans id portant le style petitTexte et contenant (certaines balises n'étant pas encore vues, elles vous sont données et elles seront étudiées plus tard) :

```
login : <input type="text" maxlength="20" size="10"
class="petitTexte" /><br />

mdp : <input type="password" maxlength="20" size="5"
class="petitTexte" />

<input type="button" class="petitTexte" size="2"
value="ok" />
```

- La balise `divLiens` va fonctionner sur la même logique que la balise `divIdent`, excepté que le contenu texte portera sur les 2 liens.
- La balise `divNews` contient la balise `divTextNews` portant le style `petitTexte` et contenant, pour le moment, la balise `marquee` qui permet de faire défiler le texte (cette balise étant dépréciée et non standard par rapport à la norme w3c, on la changera par la suite). Voici la balise et son contenu :

```
<marquee direction="up" scrollamount="2" height="100%"
width="100%">

    <b>23/07/2009 : </b>Echiquier en bois 50x50cm<br />

    <b>18/07/2009 : </b>Canon miniature édition limitée à
3000 exemplaires, numéroté avec certificat<br />

    <b>15/07/2009 : </b>voitures, motos, scooters
miniatures<br />

</marquee>
```

- La balise `divPied`, en style `petitTexte`, contient le texte (regardez l'exemple dans la capture d'écran, plus haut), des espaces et un lien sur "contact". Voici le lien (qui permet d'envoyer un mail) :

```
<a href="mailto:test@free.fr">contact</a>
```

Pensez aussi à ajouter ce lien sur le contact qui est dans le menu.

Le code est terminé pour normalement obtenir le résultat présenté plus haut.



Faites un test pour contrôler que le résultat correspond bien à ce qui est attendu. Contrôlez aussi que les liens sur le menu fonctionnent correctement (pas encore sur les images).

Faites une sauvegarde de cette nouvelle version dans un dossier "version 4.8 infographie". N'oubliez pas d'y copier aussi le dossier `css`.

## Synthèse

Les frames :

Découpage d'une page en plusieurs zones, chaque zone comportant une page indépendante. Solution peu performante et dépassée.

Les tableaux :

Intégration de tableaux permettant de positionner les informations dans différentes cellules. Solution intermédiaire lourde au niveau codage.

Les calques :

Conteneurs positionnables et dimensionnables à volonté : solution la plus performante pour mettre en page un site.

## Synthèse

Les styles :

Ensemble de caractéristiques affectable à une ou plusieurs balises.

Style appliqué à toutes les balises d'un même type :

`nomBalise {...}`

Style appliqué à une balise possédant un id :

`#valeurId {...}`

Style indépendant, applicable à toute balise avec l'attribut class :

`.nomStyle {...}`

Style indépendant, applicable à toute balise d'un seul type, avec l'attribut class :

`nomBalise.nomStyle {...}`

Modification d'un style prédéfini :

`nomBalise:nomStylePredefini {...}`

La feuille de style :

Créer une feuille de style :

Créer un fichier avec l'extension CSS, regroupant un ensemble de styles.

Attacher une feuille de style à une page :

`<link rel="stylesheet" type="text/css" href="monstyle.css" />`

Les images

Choisir les formats les plus légers (jpg, png...). Réduire la taille non pas dans la page mais avant d'intégrer l'image, pour réduire son poids. Réduire le nombre de chargement des images.

Le travail de l'infographiste

Il intervient essentiellement au niveau du CSS : choix du positionnement des informations et création des images.

## Séquence 5

# JavaScript : langage client

Cette séquence aborde le langage client JavaScript ainsi qu'un premier aspect de l'étendue de ses possibilités. Ce langage permet d'intégrer de la programmation côté client, en insérant par exemple des tests, ou en dynamisant les pages. Vous apprendrez entre autre à créer un bandeau d'images dynamiques et des news qui défilent.

### ► Capacités attendues en début de séquence

Avoir de bonnes notions algorithmiques et connaître les bases du HTML (abordé dans les séquences précédentes).

### ► Contenu

1. Introduction .....	56
2. Positionnement du code .....	56
3. Éléments de syntaxe .....	59
4. Outils prédéfinis .....	64
5. DOM .....	66
6. Priorité au JavaScript .....	69
7. JavaScript non intrusif .....	73

## 1. Introduction

JavaScript est un langage interprété, directement exécuté par le navigateur, donc côté client, comme le html. En quoi un langage client est-il intéressant ? Il apporte la puissance d'un vrai langage de programmation, avec la possibilité de manipuler des variables, de réaliser des traitements conditionnels ou itératifs, bref d'aller beaucoup plus loin dans les traitements possibles d'une page.

Par exemple, avec le JavaScript il va être possible de contrôler la saisie d'un utilisateur et de lui afficher un message d'erreur personnalisé.

## 2. Positionnement du code

Le code JavaScript peut se positionner à différents endroits. Il existe plusieurs possibilités.

### Directement dans une balise

Le JavaScript est normalement sollicité suite à un événement (automatique ou provoqué par l'utilisateur). Par exemple, suite à un clic sur un bouton, on peut insérer du code JavaScript qui va afficher un message :

```
<input type="button" onclick="alert('bonjour')" />
```

Il est aussi possible de mettre du code JavaScript à un endroit qui n'attend à priori pas de JavaScript. Il faut alors préciser en premier que le code qui suit est du JavaScript :

```
<a href="javascript:alert('coucou')">cliquez ici pour  
afficher le message</a>
```

Ces solutions sont les moins bonnes pour 2 raisons : d'abord parce que le code qui est inséré est limité en taille (on peut faire plus long mais cela devient vite illisible), ensuite parce que le code JavaScript se mélange au code html ce qui risque de rendre les choses rapidement difficiles à gérer.

### Dans l'entête de la page

Une seconde solution, un peu plus propre que la première, revient à regrouper le code JavaScript dans des fonctions placées en tête de page, et d'appeler ces fonctions dans les balises.

```
<head>

<script type="text/javascript">

    function hello() {

        alert("coucou") ;

    }

</script>
```

```
</head>
...
<input type="button" onclick="hello()" />
...
```

Même si ceux sont des fonctions qui sont appelées, on a toujours un mélange entre le JavaScript et le html.

### Dans un fichier js

Plutôt que de mettre le code dans l'entête de la page, mettons-le dans un fichier séparé. Si l'extension de ce fichier est "js" alors le fichier sera reconnu comme contenant du JavaScript. Voici comment ensuite lier le fichier à la page :

```
<head>
    <scripttype="text/javascript" src="monCode.js"></script>
</head>
...
<input type="button" onclick="hello()" />
...
```

Cela suppose que dans un fichier monCode.js se trouve la fonction hello().

### Isolement total du code JavaScript

Cette solution est déjà plus efficace, mais n'est toujours pas totalement satisfaisante. Pourquoi ? Parce qu'il y a toujours un petit mélange de code : derrière l'événement "onclick", c'est du code JavaScript qui est écrit puisque c'est l'appel d'une fonction JavaScript. La séparation peut être totale. Observez le contenu du fichier js :

```
window.onload = function() {
    document.getElementById("btnHello").onclick = function()
    {
        alert("coucou") ;
    }
}
```

Puis le contenu du code de la page :

```
<head>
    <scripttype="text/javascript" src="monCode.js"></script>
</head>
```



...

```
<input id="btnHello" type="button" />
```

Il n'est plus nécessaire de marquer l'événement dans la page. Cette fois la séparation est totale. Cependant, vous remarquez que pour affecter un code au bouton, il faut que celui-ci possède un id (identifiant). Dans le fichier js, on fait appel à l'élément par son id et on écrit la fonction événementielle attachée à l'événement onclick. Mais attention, pour que cette fonction événementielle soit chargée dès l'ouverture de la page, on la place sur l'événement chargement (onload) de l'objet window.

A partir de ces connaissances, nous allons faire une première modification sur le site. Le but va être d'afficher un message si, en cliquant sur ok, le login ou le mot de passe n'ont pas été remplis. Créez un dossier js (qui contiendra le fichier JavaScript) et dans ce dossier, un fichier boutique.js. Rattachez-le à votre projet. Dans ce fichier, mettez le code suivant :

```
window.onload = function() {

    document.getElementById("cmdOk").onclick = function() {

        if (document.getElementById("txtLogin").value==" " ||
            document.getElementById("pwdMdp").value=="") {

            alert("les 2 champs doivent être remplis") ;

        }

    }

}
```

Dans index, ajoutez l'attribut id "txtLogin" dans la balise du login, l'attribut id "pwdMdp" dans la balise du mot de passe et l'attribut id "cmdOk" dans la balise du bouton.



Faites plusieurs tests pour contrôler que le code fonctionne correctement :

- remplissez les 2 champs puis ok, normalement il n'y a pas de message
- remplissez seulement le login puis ok, le message apparaît
- remplissez seulement le mot de passe puis ok, le message apparaît
- laissez les 2 champs vides, le message apparaît

Pour être capable de lire le code de pages existantes, il était important que vous découvriez les différentes méthodes de positionnement du JavaScript. Dans la suite de ce tp, c'est bien sûr la dernière méthode qui sera appliquée.

### 3. Éléments de syntaxe

Pour comprendre le positionnement du code, vous avez commencé à écrire un peu de code. Il est temps de voir de façon un peu plus détaillée les différents éléments de syntaxe du JavaScript.

Le JavaScript est un langage dont la syntaxe est basée, comme beaucoup de langages, sur la syntaxe du C, avec bien sûr moins de fonctionnalités et en étant plus permissif. N'oublions pas aussi que la grande différence entre JavaScript et un langage comme le C est la méthode d'exécution : le C est compilé alors que le JavaScript est interprété. Quelle est la différence ? La rapidité d'exécution mais aussi le repérage des erreurs. Dans un langage interprété, l'erreur est mise en évidence que si la ligne est exécutée.

#### Introduction

Voici quelques points importants du langage :

- sensibilité à la casse : comme le C, les majuscules et minuscules ne sont pas interprétées de la même façon.
- Les ";" : comme en C, ils marquent la fin d'une instruction, cependant en JavaScript ils ne sont pas obligatoires car un changement de ligne représente aussi un changement d'instruction (mais autant garder les bonnes habitudes et les mettre).
- Les {} : comme en C, elles permettent d'entourer un bloc d'instructions dans une même structure.
- Les " " et ' ' : contrairement au C, il n'y a pas de distinction entre un caractère et une chaîne. Du coup, les guillemets(") et les côtes(') sont interchangeables pour encadrer les chaînes.

#### Les variables

Normalement la déclaration d'une variable se fait en la précédant du mot clé "var". Dans ce cas la variable est locale à l'environnement où elle se trouve. Donc si elle est déclarée dans un module, elle est locale au module. Si elle est en tête de fichier, extérieure à tout module, elle est globale à tout le fichier.

En l'absence du mot clé "var", la variable sera globale quelque soit sa position. Il est préférable d'éviter d'utiliser cette possibilité.

```
var uneVariable ;    // déclaration explicite locale  
  
uneAutreVariable = 2 ;    //déclaration implicite globale
```

Le typage d'une variable est implicite en JavaScript : la variable est typée lors de la première affectation. Cependant, il est tout à fait possible de typer explicitement une variable, ce qui est nettement moins fréquent.

```
var uneVar = "bonjour" ; // typage implicite  
  
var uneAutreVar = new String("bonjour") ; // typage  
explicite
```

Les types classiques sont accessibles : chaîne, divers types numériques, booléen.

La déclaration d'un tableau se fait à l'aide du mot clé "Array" :

```
var unTableau = new Array(10) ; // tableau de 10 cases
```

La syntaxe d'accès aux cases d'un tableau est similaire au langage C, cependant la grande différence réside encore une fois sur le typage. En effet, il n'est pas interdit d'avoir des types différents dans chaque case.

### La syntaxe du langage

Toutes les bases du langage (affectation, calcul, comparaison, condition, boucle) sont identiques au langage C. Les commentaires fonctionnent aussi comme pour le C.

### Les fonctions

Il n'y a pas de typage explicite des modules en JavaScript, du coup il n'y a pas de distinction entre "procédure" et "fonction", excepté par la présence ou non du mot clé "return" à l'intérieur du code.

```
function nomFonction (paramètres) {
    ...
}
```

La liste des éventuels paramètres ne comporte que les noms des paramètres.

### L'accès aux données de la page

Vous avez commencé à voir un début de syntaxe dans des exemples précédents : l'accès à un objet graphique html de la page va se faire par son id. Donc chaque objet qui doit être manipulé en JavaScript doit être identifié avec un id unique. C'est finalement le même principe que pour l'utilisation d'un objet avec le CSS.

Une fois l'objet identifié, le JavaScript y accède de la façon suivante :

```
document.getElementById("idObjet")
...
document.getElementById(variableContenantUnId)
```

Directement l'id entre guillemets ou une variable contenant l'id

Ensuite, il suffit de préciser la propriété qui nous intéresse. Par exemple, plus haut, on avait utilisé la propriété "value" pour accéder à la valeur de l'objet.

### Exemple de création d'une fonction

Vous avez déjà inséré une fonction dans votre site, mais elle était donnée. Cette fois on va écrire une fonction JavaScript pour remplacer la balise dépréciée "marquee" que l'on avait utilisé en attendant de faire mieux.

#### Principe

Le but va être d'écrire une fonction qui permet de faire défiler un calque dans un autre calque. On les appellera "calque contenu" et "calque conteneur". Dans un premier temps, la fonction va juste se contenter de faire défiler de bas en haut et de recommencer au début lorsque le défilement est terminé. Une fois que cette partie sera opérationnelle, quelques améliorations seront apportées.

Avant tout, dans index.html, enlevez la ligne de la balise marquée et la ligne de la fin de cette balise (n'enlevez pas le contenu de la balise, c'est-à-dire le texte des news).



Faites un test pour contrôler que le texte des news s'affiche mais ne défile plus.

### Signature de la fonction

Dans le fichier boutique.js, en tête de fichier (donc avant la fonction déjà écrite) écrivez la signature de la fonction (son entête) : la fonction va s'appeler `defile` et possède 3 paramètres (`idContenu` qui contiendra l'id du calque à déplacer, `idConteneur` qui contiendra l'id du calque conteneur pour connaître les limites de défilement, `temps` qui contiendra le temps en milliseconde entre chaque étape de défilement pour gérer la vitesse).

### Appel de la fonction

Avant même de remplir la fonction, on va positionner son appel pour le défilement des news. Dans la fonction `window.onload` précédemment écrite, en fin de fonction, appelez la fonction `defile` en envoyant en paramètres `"divTextNews"` pour le contenu, `"divNews"` pour le conteneur et `10` pour le temps.



Avant de commencer le code de la fonction, voyons si la fonction est correctement appelée : dans la fonction `defile`, ajoutez juste un message :

```
alert("test") ;
```

puis faites un test : logiquement dès l'ouverture de la page, la petite fenêtre d'alerte doit s'afficher. Si c'est le cas, la fonction est correctement appelée. Supprimez dans le code la ligne d'alerte.

### Variables nécessaires

La fonction `defile` va contenir plusieurs variables, mais aussi une autre fonction récursive (c'est-à-dire qui s'appelle elle-même) pour gérer le défilement. Commençons par les variables.

Dans la fonction `defile`, il faut avant tout récupérer les 2 calques dans des variables locales : déclarez les variables `leContenu` et `leConteneur` et affectez à ces variables les objets correspondants (en utilisant leur id reçu en paramètre). Attention, pour récupérer les objets correspondants, n'oubliez pas qu'il faut utiliser `document.getElementById`.

Déclarez ensuite la variable `hContenu` qui va contenir la hauteur du calque du contenu :

```
var hContenu = leContenu.offsetHeight ;
```

La propriété `offsetHeight` permet de récupérer la hauteur du calque. On aurait pu utiliser `leContenu.style.height` mais cela retourne une chaîne qu'il faut ensuite convertir. `offsetHeight` permet d'obtenir directement la valeur numérique (valable aussi pour `width`, `top` et `left`).

Avec la même logique, déclarez la variable `min` et affectez-lui la position du haut du calque du contenu (`offsetTop`). Déclarez aussi la variable `max` qui cette fois doit recevoir la hauteur du calque du conteneur. Déclarez enfin la variable `position` et initialisez là avec `max`. La variable `position` permettra de modifier la position du top du calque du contenu, afin de le faire bouger dans le conteneur. Le déplacement se fera de `max` à `min`.

### Fonction réursive

Juste avant les déclarations, mais toujours dans la fonction `defile`, écrivez la fonction `go()` qui ne comporte aucun paramètre (oui, on peut écrire une fonction dans une fonction !). Dans cette fonction, commencez par décrémenter la variable `position` puis faites un test : si le défilement est terminé (donc si la variable `position` + la hauteur du calque du contenu est inférieur à `min`) alors réinitialisez `position` avec `max`, pour que le défilement redémarre d'en bas. Après ce test, dans tous les cas il faut concrètement affecter la nouvelle position au calque : voici la ligne de code nécessaire :

```
leContenu.style.top = position + "px" ;
```

Cette ligne affecte à la propriété `top` du style du calque du contenu, la variable `position` mais remarquez bien le `"px"` ajouté. Vous aviez remarqué ce `"px"` dans la feuille de style, pour préciser l'unité utilisée (ici, c'est le pixel).

Il ne reste plus qu'à faire en sorte que la fonction `go` soit à nouveau appelée à intervalle régulier. Pour cela, nous allons utiliser la fonction JavaScript `setTimeout`, en utilisant le paramètre `temps`, qui va permettre de déterminer au bout de combien de millisecondes la fonction est rappelée. Voici la syntaxe :

```
setTimeout(go, temps) ;
```

La fonction `go` doit aussi être appelée une première fois : en fin de fonction `defile`, après les déclarations, appelez simplement et sans timing la fonction `go` avec la ligne de code suivante :

```
go() ;
```

Il est temps de faire un premier test...



Lancez un test : normalement vous allez voir le calque du texte défiler de bas en haut et reprendre le défilement lorsque celui-ci est terminé. Mais vous avez remarqué deux problèmes : le calque passe "par-dessus" l'image, et le calque conteneur n'a pas le fond blanc, ce qui n'est pas très joli.

Ces 2 problèmes sont indépendants de la fonction JavaScript qui vient d'être écrite. Il faut revoir le code html et le css. Voici comment résoudre ces deux problèmes.

Problème de l'image :

Dans `index.html`, dans le calque `"divNews"`, juste avant le calque `"divTextNews"`, créez un calque `"divImgNews"` vide. Dans la feuille de style, créez un nouveau style pour le calque `"divImgNews"` en lui affectant l'image `"news.jpg"` qu'il faut enlever du calque `"divNews"`. Donnez aussi une taille à ce nouveau calque (la taille de l'image) mais ne mettez pas de `top` et de `left` (pour qu'il se positionne dans le coin haut gauche de son conteneur).

Pourquoi avoir mis l'image dans un nouveau calque ? Parce qu'il est possible de déterminer pour chaque calque une position "dessus/dessous" par rapport aux autres calques. On va placer ce calque de l'image par-dessus tous les autres, en le mettant au niveau 1 avec l'attribut z-index, à ajouter dans les attributs de ce calque, dans le css : faites-le.



Refaites un test : cette fois le défilement passe bien derrière l'image.

Problème de la couleur de fond :

Il suffit d'affecter au style du calque "divNnews" la même couleur de fond que le style "divTextNews".



Refaites un test : normalement le fond est blanc ce qui est plus joli.

### Hauteur des calques

Nous sommes arrivés à remplacer la balise marquée... pas tout à fait. Dans index.html, allongez la partie texte en ajoutant une vingtaine de lignes (mettez, comme les lignes précédentes, des dates et des informations).



Faites un test : remarquez que tout le texte ne défile pas, mais seulement une partie. Le problème vient de la hauteur du calque "divTextNews".

Dans la feuille de style, dans le style du calque "divTextNews", enlevez tout simplement la hauteur du calque pour que la hauteur s'adapte au contenu.



Refaites un test : cette fois tout le texte défile.

### Pause sur chaque page

Il est envisageable d'améliorer la fonction pour, par exemple, ajouter des pauses régulières à chaque "page" (donc hauteur du contenu).

Dans la fonction defile, partie déclaration des variables, ajoutez la variable page et initialisez-là à 0. Cette variable va permettre de savoir à quelle page on se trouve au cours du défilement.

Dans la fonction go, dans le test pour vérifier si le défilement est terminé, remettez à 0 la variable page (car après chaque défilement, on recommence à la page 0). Après avoir affecté la nouvelle position du top au calque contenu, faites un test pour vérifier s'il y a changement de page : pour cela, contrôlez que la position + la hauteur de la page (max-min) multiplié par la page est inférieur ou égal à min. Si c'est le cas, alors il y a changement de page. Dans ce cas, incrémentez la variable page et appelez la fonction go avec setTimeout mais cette fois en mettant temps\*100 pour le timing (pour que le temps soit plus long). Faites un else à ce test et mettez l'autre ligne de setTimeout qui reste.



Faites un test pour contrôler qu'il y a bien une pause à chaque page, puis que le défilement reprend depuis le début.

### Arrêt du défilement avec la souris

Autre amélioration : ce serait bien que le défilement s'arrête lors du survol de la souris, et recommence lorsque la souris sort de la zone du calque.

Pour cela, on va écrire les fonctions événementielles correspondantes, comme cela a déjà été fait pour le bouton ok. Dans la fonction `defile`, en fin de fonction, donc après les déclarations et l'appel de la fonction `go`, écrivez la fonction événementielle qui va se déclencher sur le survol du calque du conteneur (événement `onmouseover` directement sur l'objet `leConteneur`). Dans cette fonction, il faut utiliser la fonction `clearTimeout` qui permet d'arrêter un timer. Mais cette fonction attend en paramètre un nom de timer. Hors nous n'avons pas nommé le timer. Donc dans les variables de la fonction `defile`, déclarez la variable `leTimer` sans l'initialiser. Lors des 2 appels de `setTimeout`, affectez le résultat de l'appel dans la variable `leTimer`. Maintenant vous pouvez mettre `leTimer` en paramètre de la fonction `clearTimeout`.

Ecrivez aussi la fonction événementielle lorsque la souris sort du calque du conteneur (événement `onmouseout`) et, dans cette fonction, appelez simplement la fonction `go()` pour relancer le défilement.



Faites un test pour contrôler qu'en plaçant la souris sur le calque, le défilement s'arrête, puis reprend à l'endroit où il s'était arrêté lorsque la souris sort du calque.

Voilà, la fonction `defile` est terminée. Faites une sauvegarde de votre travail dans un dossier "version 5.3 script", sans oublier les dossiers css et js.

## 4. Outils prédéfinis

A travers les premières lignes de code que vous venez d'écrire en JavaScript, vous avez déjà utilisé des outils prédéfinis (`alert`, `setTimeout`, `clearTimeout`), donc préalablement écrits et mis à votre disposition. Faisons un tour rapide des outils les plus courants.

### window

Dans l'outil prédéfini `window` se trouvent plusieurs outils qu'il est possible d'utiliser sans préciser "window" dans le cas où l'outil s'applique sur la fenêtre actuelle. Voici les syntaxes possibles avec par exemple l'outil `moveTo` qui permet de déplacer une fenêtre :

```
moveTo(10, 20) ; // déplace la fenêtre actuelle
window.moveTo(10, 20) ; // idem (syntaxe complète)
maFenetre = window.open("unepage.html") ;
```

```
maFenetre.moveTo(10, 20) ; // déplace la popup maFenetre
```

Voici quelques autres outils contenus dans window :

**focus()** : met la fenêtre en premier plan (en lui donnant le focus)

**blur()** : met la fenêtre en arrière plan

**open(url)** : ouvre une url dans une popup

**close()** : ferme une popup (attention, close doit être précédé du nom de la variable qui a reçu l'ouverture de la popup, sinon la fermeture porte sur la fenêtre active)

**confirm(message)** : affiche le message dans une boîte de dialogue, avec 2 boutons (ok et annuler) et retourne true si l'utilisateur a cliqué sur ok, false sinon

**prompt(message)** : affiche le message dans une boîte de dialogue, avec une zone de saisie et 2 boutons (ok et annuler) et retourne le texte saisi si l'utilisateur a cliqué sur ok, retourne null sinon

**moveTo(x, y)** : positionne la fenêtre aux coordonnées (x, y) par rapport au coin haut gauche du navigateur

**moveBy(x, y)** : déplace la fenêtre horizontalement de x pixels et verticalement de y pixels par rapport à sa position actuelle

**resizeTo(x, y)** : redimensionne la fenêtre (x pixels de large et y pixels de haut)

**resizeBy(x, y)** : grandit (valeurs positives) ou réduit (valeurs négatives) la fenêtre

**print()** : imprime la page en cours

### Les groupes d'outils de window

Dans window se trouvent aussi plusieurs outils qui contiennent eux même des outils. Voici très rapidement le rôle de certains :

**document** : offre des outils qui permettent d'accéder au contenu de la page (par exemple, vous avez vu document.getElementById)

**history** : offre des outils qui permettent d'accéder à l'historique des pages

**location** : offre des outils qui permettent la navigation

**screen** : offre des outils qui permettent d'avoir des informations sur l'écran (résolution...)

### Les autres outils

En dehors de window qui est l'outil le plus important, JavaScript offre plusieurs autres outils. Voici globalement le rôle de certains d'entre eux :

**string** : offre des outils pour la manipulation des chaînes

**navigator** : offre des outils qui permettent d'accéder aux propriétés du navigateur

**Math** : offre des outils mathématiques

**Image** : offre des outils qui permettent de manipuler les images

**Date** : offre des outils qui permettent de manipuler les dates

**Array** : offre des outils qui permettent de manipuler les tableaux



Vous trouverez facilement sur internet la liste détaillée du contenu de ces outils. Exemple de site qui donne ce genre de détails :

<http://www.toutjavascript.com/reference/>

### Exemple d'utilisation des outils

En utilisant les outils de screen (les propriétés height et width donnent la résolution de l'écran) et en récupérant la taille du calque principal, faites en sorte de positionner la fenêtre du navigateur au centre de l'écran, avec une bordure maximale de 50 pixels tout autour du calque (dans la mesure où la taille de l'écran le permet). Comme le navigateur possède une certaine hauteur d'entête, variable d'un navigateur à l'autre, n'ayant pas trouvé de moyen pour connaître cette hauteur, prévoyez 100 pixels. Une fois la fonction écrite, pensez à l'appeler au chargement du site.



Faites un test pour contrôler que le navigateur est bien redimensionné et centré. Si vous n'y arrivez pas, aidez-vous des explications suivantes.

En cas de blocage, voici quelques éléments pour vous aider :

- L'entête de la fonction doit comporter 2 paramètres : l'id du calque et la marge demandée (ce n'est pas obligatoire mais cela rend la fonction un peu plus réutilisable).
- Récupérez dans des variables locales la hauteur et largeur du calque, idem pour l'écran (screen.availWidth et screen.availHeight).
- Calculez la position x et y de la fenêtre de la façon suivante : enlevez à la taille de l'écran, la taille du calque et 2 fois la marge. Le résultat doit être divisé par 2. Attention, si le résultat est négatif, il faut le remplacer par 0. Dans le cas de la hauteur, enlevez aussi la hauteur de l'entête du navigateur.
- Calculez la hauteur et largeur de la fenêtre de la façon suivante : ajoutez à la taille du calque, 2 fois la marge, + la hauteur de l'entête du navigateur dans le cas du calcul de la hauteur. Si le calcul dépasse la taille de l'écran, prenez alors la taille de l'écran.
- Il ne reste plus qu'à utiliser moveTo et resizeTo pour repositionner et redimensionner la fenêtre.

Remarque : cet exercice est là surtout pour vous faire coder et vous montrer quelques possibilités du JavaScript. Vous en verrez beaucoup d'autres par la suite. Cependant, positionner et dimensionner la fenêtre de navigation d'un utilisateur est généralement quelque chose de mal perçu, donc à éviter. Après avoir testé, mettez juste en commentaire la ligne de code qui appelle la fonction de redimensionnement.

## 5. DOM

Document Object Model est une spécification qui permet de décrire la structure et l'accès aux objets d'un document. Vous avez donc déjà utilisé le DOM en passant par "document", par exemple en écrivant `document.getElementById`.

### Règles principales

Le DOM fixe les règles de structure d'un document. Il peut s'appliquer à n'importe quel type de documents. Le W3C (que l'on abordera plus tard) a standardisé le DOM ce qui évite de nombreuses variantes. Les pages HTML, les documents XML sont des documents qui typiquement doivent respecter les règles du DOM.

Voici quelques points fondamentaux du DOM :

- structure d'arbre : le document doit être structuré sous forme d'arbre, avec des balises imbriquées
- nœuds : un élément du document est appelé un nœud, possédant une valeur et des attributs
- fonctions standards : un ensemble de fonctions standards permettent de manipuler les nœuds (ajout, suppression, modification)

Toutes ces notions seront approfondies dans la séquence qui aborde le XML.

### Méthodes et propriétés classiques

L'outil "document" qui a été étudié dans la partie précédente, respecte les règles du DOM.

Voici rapidement quelques méthodes et propriétés très classiques sur un document :

**`getElementById(idBalise)`** : accède à l'élément dont l'id est passé en paramètre

**`getElementsByTagName(nom)`** : accède à l'élément ou la collection d'éléments dont le nom est passé en paramètre

Une fois un élément trouvé, on peut récupérer, voire modifier un de ses attributs. Voici quelques propriétés qui peuvent s'appliquer :

**`value`** : valeur brute de l'élément

**`innerHTML`** : valeur formatée de l'élément de type balise html

Il y a ensuite toutes les propriétés spécifiques à chaque type d'élément. Par exemple, si l'élément est une balise "a", il est possible d'accéder à sa propriété "href".

Dans le cas d'une collection d'éléments, il est possible d'accéder à un élément de la collection avec un indice, comme pour un tableau. De plus, la propriété `length` permet d'obtenir le nombre d'éléments de la collection. Voici un exemple qui montre comment récupérer la collection de toutes les balises "a" (liens) et d'afficher le contenu du href de chacune des balises :

```
var colA = document.getElementsByTagName("a") ;  
for (var k=0 ; k<colA.length ; k++) {  
    alert("l'attribut href du "+k+"e élément :")  
}
```

```

    "+colA[k].href);
  }

```

Cet exemple donne un petit aperçu de l'étendue des possibilités de manipulation des objets.

### Cas particulier du style

Nous avons déjà utilisé le style, par exemple pour modifier la hauteur d'un calque. Il suffit de faire suivre le nom de l'élément, de la propriété style, puis du style concerné. Voici un exemple de syntaxe :

```

document.getElementById("textNews").style.height =
position+"px"

```

D'où la syntaxe générale :

```

document.getElementById(unId).style.unStyle = uneValeur

```

Mais attention, si le nom du style comporte un underscore (\_) dans le css, il faut en JavaScript l'enlever et remplacer la première lettre qui suivait l'underscore par une majuscule. Voici un exemple :

```

document.getElementById("textNews").style.backgroundColor
= ...

```

### Créer de nouveaux éléments

Il est possible de créer de nouveaux éléments graphiques html à partir du JavaScript. Ce système est très pratique pour faire évoluer la page suivant les besoins, sans faire appel au serveur. Voici la syntaxe :

```

var unObjet = document.createElement(nomBalise) ;

// exemple de création d'une balise img :
var uneImage = document.createElement("img") ;

```

Une fois l'objet graphique créé, il est possible de modifier ses propriétés. Voici la syntaxe et quelques exemples pour mieux comprendre :

```

unObjet.unePropriete = uneValeur ;

// exemples
uneImage.src = "images/fond.jpg" ;
uneImage.style.height = "500px" ;

```

Vous avez compris qu'il est possible de créer n'importe quelle balise et de modifier tous ses paramètres en JavaScript. Une fois une balise créée, il faut l'intégrer dans

l'arbre du DOM, donc l'insérer dans un autre objet graphique de type conteneur (comme par exemple un div, un p, un form...). Voici la syntaxe :

```
unObjetConteneur.appendChild(unObjet) ;
```

Ici l'objet conteneur est présenté comme une variable JavaScript qui a soit été créée en JavaScript, soit récupéré un objet du DOM par son id (par exemple). Voici donc une autre syntaxe que l'on pourrait trouver :

```
document.getElementById(unId).appendChild(unObjet) ;
```

Cette fois on accède directement à l'objet conteneur par son id.

De même que le JavaScript permet de créer un objet, il permet de le détruire. Voici la syntaxe :

```
// suppression de unObjet qui est dans unObjetConteneur
```

```
unObjetConteneur.removeChild(unObjet) ;
```

```
// suppression de unObjet qui est dans un objet repéré par  
l'id
```

```
document.getElementById(unId).removeChild(unObjet) ;
```

Vous allez utiliser ces possibilités lors de la création de la page de génération de t-shirts où il faudra créer à la volée des div contenant de petites images à insérer sur le t-shirt. La séquence correspondante fera alors appel à cette partie du cours.

## 6. Priorité au JavaScript

Le JavaScript est un langage qui a pris en puissance et qui présente de nombreux avantages. Voyons rapidement quelques points forts.

### Pourquoi des traitements côté client ?

Le langage côté serveur (nous allons étudier le PHP dans la séquence suivante) n'a pas encore été abordé. Nous verrons que dans certains cas, il n'est pas possible de se passer d'un langage serveur. Cependant, comme la sollicitation du serveur nécessite du temps, il faut la minimiser au maximum. De plus, compte tenu des multiples possibilités du JavaScript, il ne faut pas hésiter à lui donner la priorité... sauf si l'utilisateur a désactivé le JavaScript dans son navigateur ! Nous aborderons ce point dans la partie suivante.

### Les contrôles de saisie

Une des erreurs classique consiste à faire un contrôle de saisie côté serveur alors que le JavaScript peut très bien s'en charger. D'ailleurs c'est ce que nous avons déjà fait avec la saisie du login et mot de passe. Le JavaScript ne peut pas contrôler si le login et mot de passe sont correctes (car il faut un accès à la base de données qui est côté serveur) mais il peut au moins vérifier si les champs ont bien été remplis.

D'une manière générale, tous les contrôles, voire tous les traitements, qui ne nécessitent pas le serveur doivent se faire en JavaScript.

### Les modifications de contenu

Le JavaScript permet de rendre dynamique la page en modifiant certains contenus. Par exemple, lorsque l'utilisateur aura saisi son login et mot de passe, et après avoir cliqué sur ok, un contrôle sera fait côté serveur pour vérifier que l'utilisateur existe (nous verrons ce contrôle plus loin) et si c'est le cas, la zone d'identification doit se transformer en zone de bienvenue. Comment ? Il faut prévoir un second calque caché, contenant un message de bienvenue. Ce calque apparaîtra lors de la reconnaissance de la personne, alors que le premier calque deviendra invisible. Voici les étapes de réalisation de ce travail :

- Dans index.html, dans le calque "divIdent", se trouve un sous-calque "divTextIdent". A la suite de ce sous-calque et en restant dans le calque "divIdent", faites un copier/coller du calque "divTextIdent" et appelez le "divTextBienvenu". Changez le titre "déjà client" en "bienvenue". Supprimez les 3 lignes du login, mot de passe et le bouton, et remplacez-les par ces 2 lignes :

```
bonjour <label id="lblLogin"></label><br /><br />
<a href="#" id="aDeconnecter">se déconnecter</a>
```

La balise label sera remplie par la suite avec le login de l'utilisateur. Le lien "se déconnecter" n'a pas de page de destination mais permet juste d'exécuter une déconnexion sur le clic du lien.

- Dans la feuille de style, faites un copier/coller du style #divTextIdent et nommez le #divTextBienvenu. Mettez en plus la propriété visibility à hidden.
- Dans le fichier js, dans la fonction événementielle correspondant au clic sur le bouton ok, ajoutez un else au test et, dans le else, rendez invisible le calque "divTextIdent" (avec "hidden") et rendez visible le calque "divTextBienvenu" (avec "visible").
- Encore dans le fichier js, au chargement de la page, ajoutez la fonction événementielle sur le clic sur le lien "deconnecter". Dans cette fonction, rendez invisible le calque "divTextBienvenu" et visible le calque "divTextIdent".

Bien sûr pour le moment aucun contrôle n'est fait : on se contente de basculer d'un calque à l'autre sans faire appel au serveur.



Faites un test en saisissant un login et mot de passe et en cliquant sur ok : normalement le calque a été remplacé par bienvenue. Il contient "bonjour" et le lien "se déconnecter". En cliquant sur le lien, vous revenez au calque initial. Si cette fois vous laissez 1 ou les 2 champs vides, en cliquant sur ok, vous obtenez un message d'erreur mais vous restez sur le même calque.

Vous avez vu que les 2 calques existent depuis le début et qu'il suffit de jouer sur la visibilité de chaque calque pour le faire apparaître ou non.

### Les animations

Beaucoup de personnes pensent que pour insérer des animations sur une page, il faut utiliser la technologie flash. Par exemple, de nombreux modules de diaporama sont proposés en flash. Il est vrai que certains effets ne peuvent pas se faire sans

flash mais cette technologie est gourmande et parfois utilisée alors qu'il est possible de gérer de nombreux effets en JavaScript.

Vous avez déjà réalisé une petite animation en gérant le défilement du calque pour les news. Dans le même esprit, on peut gérer un défilement d'images dans le calque du bandeau. Actuellement il contient une image fixe. On va faire en sorte qu'à intervalle régulier, l'image s'enroule pour découvrir une nouvelle image. C'est un effet très facile à réaliser en JavaScript.

Le principe est le suivant : on va gérer 2 sous-calques dans le calque du bandeau, chacun recevra une image de fond. En jouant sur la superposition des calques (un coup le premier sous-calque sera dessus, un coup il sera dessous) et en réduisant la largeur du calque de dessus, on va obtenir l'effet désiré (défilement vers la droite en laissant découvrir l'image du dessous). Voici plus en détail les étapes de codage :

- Dans index.html, dans le calque du bandeau, créez les 2 sous-calques vides avec les id `divImg0` et `divImg1`.
- Dans la feuille de style, créez les 2 styles pour `divImg0` et `divImg1` en mettant juste le `width` et le `height` à 100% (pour qu'ils remplissent le calque bandeau).
- Dans le fichier js, dans la fonction du chargement de la page, à la position de votre choix, créez la fonction `imageBandeau` sans paramètre. Cette fonction va s'occuper de gérer les informations spécifiques à cette page. A l'extérieur de la fonction du chargement de la page, on écrira les fonctions qui sont réutilisables. Donc, dans la fonction `imageBandeau`, voici les étapes à suivre :
  - déclarez la variable `cheminImages` et initialisez la avec `"images/"` (pour récupérer les images dans le dossier correspondant)
  - déclarez la variable `nblImages` et initialisez-la à 4 (nous allons tourner sur 4 images dans le bandeau)
  - déclarez la variable `tabNomsImages` de type tableau (ce tableau contiendra les noms des fichiers images du bandeau)
  - faites une boucle pour remplir le tableau précédent afin de mettre dans chaque case le nom complet (chemin compris) des images, sachant que les noms des fichiers sont `"bandeau0.jpg"`, `"bandeau1.jpg"`...
  - déclarez la variable `lesImages` et affectez lui l'appel de la fonction (que l'on va écrire juste après) `chargelImages` en envoyant en paramètre de cette fonction, le tableau `tabNomsImages` : la variable `lesImages` recevra en retour un tableau d'images (non pas uniquement le nom de l'image, mais les images elles-mêmes)
  - écrivez à cet endroit la fonction événementielle qui va s'exécuter en fin de chargement de la dernière image (car il ne faudra démarrer l'animation que lorsque toutes les images seront chargées), voici l'entête de la fonction :

```
lesImages[lesImages.length-1].onload = function() {
```

Dans cette fonction événementielle, appelez la fonction (que l'on va aussi écrire juste après) `animlImages` en lui envoyant les 3 paramètres suivants : `"divImg0"`, `"divImg1"` et le tableau `lesImages` (donc les 2

calques qui doivent recevoir les images alternativement, et le tableau qui contient les images à afficher)

- fermez la fonction imageBandeau et juste après, appelez la fonction pour qu'elle soit appelée dès le chargement de la page
- Toujours dans le fichier js, cette fois avant la fonction du chargement de la page, créez la fonction chargelImages qui attend un paramètre : appelez le tabNoms. Cette fonction va s'occuper de charger en mémoire (dans un tableau) les images du bandeau dès l'ouverture de la page. Voici son contenu :
  - déclarez une variable lesImages de type tableau
  - faites une boucle sur la taille du tableau tabNoms (tabNoms.length donne la taille) en utilisant un indice k
  - dans cette boucle, affectez à la kème case de lesImages, une instance de Image() : voici la ligne correspondante

```
lesImages[k] = new Image() ;
```

- ensuite, affectez à la propriété src de la même case, la kème case du tableau tabNoms de la façon suivante :

```
lesImages[k].src = tabNoms[k] ;
```

Cette propriété correspond à la source de l'image, donc son nom.

- après la boucle, retournez le tableau lesImages

Dans le css, enlevez l'image qui est rattachée au calque "divBandeau".

Avant de continuer, faisons un petit contrôle : mettez une "alert" dans la fonction événementielle onload sur la dernière image, juste avant l'appel de la fonction animImages qui n'est pas encore écrite. Mettez en commentaire l'appel de la fonction animImages.



Faites un test : normalement à la fin du chargement des images, le message d'alert devrait s'afficher et vous devriez voir le nom de la dernière image (bandeau3.jpg). En revanche le bandeau n'apparaît plus, c'est normal.

Si le message apparaît, c'est que les images ont bien été chargées et que la fonction événementielle sur le chargement de la dernière image s'est bien déclenchée. Enlevez l'alert et la marque de commentaire que vous aviez ajouté devant l'appel de la fonction animImages.

- Il reste à écrire la fonction animImages : vous allez l'écrire avant la fonction précédente. La fonction animImages reçoit 3 paramètres : idImage1, idImage2 et tabImages. Les 2 premiers paramètres contiennent les id des 2 calques qui se superposent et doivent recevoir les images, le 3<sup>ème</sup> paramètre contient le tableau d'images. Le principe est le suivant : les 2 calques contiennent une image chacun, un des 2 calques voit sa taille diminuer laissant découvrir l'autre calque et vice versa. Il va donc falloir régulièrement inverser l'ordre des calques (dessus/dessous) : pour cela, il faudra modifier la propriété de style z-index des calques (plus le numéro est grand et plus le calque est "dessus", vous pouvez simplement alterner entre les valeurs 1 et 2). Reste ensuite à utiliser le timer comme cela a déjà été fait. Un petit détail risque de vous gêner : les 2 calques



des postits qui chevauchent sur le bandeau vont certainement se retrouver en arrière plan du bandeau : à vous de leur affecter directement dans la feuille de style un z-index pour éviter ce problème (par exemple 10, en tout cas une valeur supérieur à celles que vous avez utilisé pour les 2 calques du bandeau). Maintenant que vous avez toutes ces explications, essayez par vous-même d'écrire cette fonction. Vous avez les connaissances pour le faire.



Testez pour contrôler que le défilement se fait correctement et que vous voyez les 4 images passer les unes après les autres. Ne pensez pas y arriver du premier coup : vous allez passer certainement du temps, et c'est nécessaire pour votre apprentissage. Si toute fois vous arrivez à un point de découragement, alors utilisez la correction, en tentant de bien comprendre le code et en comparant avec le votre.

Dans le même esprit, vous pouvez imaginer qu'il est possible de gérer des menus déroulants et toutes sortes d'animations. Laissez libre cours à votre imagination.

Faites une sauvegarde de votre travail dans un nouveau dossier "version 5.6 animation script", sans oublier les dossiers css et js.

### **Et même les traitements côté serveur**

Après l'étude du PHP, nous verrons que le JavaScript peut aussi solliciter le serveur sans avoir à recharger la page, à l'aide de la technologie Ajax.

## **7. JavaScript non intrusif**

Il se peut qu'un internaute utilise une vieille version d'un navigateur qui n'accepte pas JavaScript, ou du moins la version que vous avez utilisé. Il se peut aussi qu'un internaute ait désactivé le JavaScript sur son navigateur. Dans les 2 cas, vos scripts ne vont pas s'exécuter, pire encore, cela va provoquer des erreurs.

Pour mieux comprendre, faisons un test :



Désactivez JavaScript sur votre navigateur (par exemple sous IE 8 : outils, options internet, onglet sécurité, personnaliser le niveau, script ASP, désactivé) et faites un test. Normalement les images du bandeau n'apparaissent plus, les news ne défilent plus et, pire encore, les 2 calques d'identification apparaissent en même temps l'un sur l'autre.

Il faut donc tenir compte de ce problème et trouver une solution. Voici quelques règles à respecter, dans l'esprit du JavaScript non intrusif :

### **Séparer le JavaScript du html**

Ce point là a déjà été respecté : tout notre code JavaScript se trouve dans un fichier isolé. Dans les pages, il ne doit y avoir qu'une seule ligne qui fait mention au JavaScript : c'est la ligne qui intègre le fichier js.

### **Mettre le JavaScript en conditionnel**

Lorsque, malgré tout, le code JavaScript est directement intégré dans le code de la page, il est possible d'éviter des erreurs, pour les navigateurs ne supportant pas le JavaScript, en mettant ce code entre commentaires html spécifiques, comme ceci :



```
<!--
code JavaScript
//-->
```

Mais vous ne devriez pas avoir à utiliser cette possibilité, si tout votre code est dans un fichier séparé. En effet, l'ordre d'intégration du fichier ne sera tout simplement pas traité et du coup l'accès au code JavaScript ne se fera pas.

### Prévoir une alternative au JavaScript

C'est le point le plus délicat. Il faut que, même sans JavaScript, il se passe des choses à peu près correctes sur le site. Il y a toujours la solution d'afficher un message à l'internaute pour lui demander d'activer le JavaScript (et tant pis pour ceux qui travaillent avec des navigateurs préhistoriques). Puisque ça existe, autant le savoir : il suffit de mettre le message entre balises spécifiques :

```
<noscript>
Activez le JavaScript pour visualiser ce site
</noscript>
```

Mettez ces 3 lignes dans le head de la page.



Contrôlez que le JavaScript est toujours désactivé sur le navigateur, puis faites un test. Vous devinez en haut à gauche le début du message, mais la page s'est tout de même affichée... ce qui n'est pas exactement ce que l'on voulait.

En fait pour utiliser ce système, il faut faire en sorte que le reste de la page s'affiche uniquement dans le cas où JavaScript est activé. Toute votre page étant dans un calque principal, il suffit de rendre invisible ce calque : allez dans la feuille de style et ajoutez ceci au calque "divPrincipal" :

```
visibility:hidden;
```

Le calque va être caché par défaut : il faut donc le rendre visible si le script fonctionne. Pour cela, allez dans le fichier js et au chargement de la page, dès le début, mettez en visible ce même calque. Cette ligne ne s'exécutera que si le JavaScript est autorisé.



Faites un test : normalement cette fois vous ne devriez voir que le message s'afficher. Activez à nouveau le JavaScript et rechargez la page : le message n'apparaît plus et la page s'affiche normalement comme avant. Désactivez à nouveau le JavaScript pour la suite des tests et supprimez les modifications de code qui viennent d'être faites (la visibilité dans le js et le css, et la balise noscript).

Vous avez donc une méthode très facile pour éviter l'exécution du site si le JavaScript est désactivé. Mais est-ce une bonne méthode ? Si l'internaute a volontairement désactivé le JavaScript, il y a des chances pour qu'il n'ait pas envie qu'on lui force la main. Si l'internaute a un trop vieux navigateur, ça ne va pas forcément lui plaire qu'à cause de cela le site ne soit pas accessible. Enfin, si l'internaute n'a tout simplement pas su activer le JavaScript, il y a peu de chance

qu'il sache le faire après la lecture du message. Il n'y a qu'un internaute étourdi (oubli d'activer le JavaScript) ou confiant qui va s'exécuter. Tout ça pour dire que ce serait bien que le site fonctionne tout de même, en version simplifiée, sans JavaScript. Deux possibilités : soit vous faites des modifications dans le même site pour prendre en compte toutes les possibilités, soit vous créez un second site complet et redirigez l'internaute vers ce second site en l'absence de JavaScript. Suivant l'utilisation du JavaScript sur votre site, la seconde solution risque d'être la plus simple. Mais cependant pour voir comment on peut aborder la première solution, voici quelques exemples.

Il faut donc trouver des moyens pour que le site fonctionne, même si c'est de façon un peu différente. Il faut alors travailler au cas par cas. Voyons comment résoudre chacun des 3 problèmes rencontrés sur notre site :

### Les news ne défilent plus

Sans JavaScript, aucune chance de faire défiler les news. On pourrait revenir à la balise marquée, mais vu son incompatibilité, c'est à proscrire. Cela veut dire que dans le cas où le JavaScript ne peut pas s'exécuter, il faut juste ajouter un ascenseur au calque des news.

Dans la feuille de style, dans le style du calque "divNews", ajoutez les ascenseurs automatiques avec la ligne de code suivante :

```
overflow:auto;
```

Comme cette ligne ne concerne que l'exécution sans JavaScript, allez dans le fichier js et, sur le chargement de la page, mettez l'overflow du calque "divNews" à "hidden".



Faites un premier test avec JavaScript désactivé : vous devriez obtenir des ascenseurs sur les news, qui vous permettent de lire toutes les news.

Faites un second test avec JavaScript activé : cette fois il ne doit plus y avoir d'ascenseur et les news défilent.

Avec ce premier exemple, vous avez compris le principe global.

### Le bandeau n'apparaît plus

Deuxième problème : le bandeau avec les images qui défilent n'apparaît plus. On aimerait que, en l'absence de JavaScript, il y ait au moins une photo fixe. Là c'est encore plus simple : dans la feuille de style, dans le style de "divImg0", ajoutez une image de fond (bandeau0.jpg). Si le JavaScript ne fonctionne pas, cette image sera fixe, sinon le JavaScript va prendre la relève pour modifier l'image.



Faites un premier test avec JavaScript désactivé : une image fixe apparaît dans le bandeau.

Faites un second test avec JavaScript activé : les images du bandeau défilent à nouveau.

### Les 2 calques d'identification se superposent

Il reste un dernier problème : les 2 calques de l'identification qui se superposent. C'est un peu plus complexe et à l'heure actuelle nous ne pouvons pas

complètement le résoudre. Vous pouvez tout de même mettre en visibilité cachée le calque de bienvenue dans la feuille de style, pour éviter cette superposition, mais rien de plus ne peut être fait pour le moment.



Avec JavaScript désactivé, seul le calque d'identification apparaît, cependant rien ne marche : si vous cliquez sur le bouton ok, rien ne se passe.

Vous avez compris qu'au fur et à mesure de la création d'un site, beaucoup de questions vont se poser pour que le site marche le mieux possible quelque soit les configurations. Voilà pourquoi, il est plus pratique de créer 2 sites différents, un site avec et un autre sans JavaScript, et de rediriger vers le bon site dès la connexion.

Activez à nouveau JavaScript pour la suite du dossier. Faites une sauvegarde de votre travail dans le dossier "version 5.7 script non intrusif".

## Synthèse

Positionnement du code :

Il est conseillé d'isoler complètement le code JavaScript dans un fichier indépendant.

Création d'un fichier JavaScript avec l'extension js.

Lien entre la page et le fichier avec la balise suivante dans le head :

```
<script type="text/JavaScript" src="monCode.js"></script>
```

Contenu du fichier js :

Les fonctions non événementielles.

Les fonctions événementielles qui doivent être chargées dès le chargement de la page, donc à mettre dans la fonction suivante :

```
window.onload = function() { ... }
```

Syntaxe du JavaScript :

Pour l'essentiel, basée sur la syntaxe du C.

Déclaration :

```
var uneVariable ; // déclaration explicite locale
```

```
uneAuteVariable = 2 ; // déclaration implicite globale
```

```
var uneVar = "bonjour" ; // typage implicite
```

```
var uneAutreVar = new String("bonjour") ; // typage explicite
```

```
var unTableau = new Array(10) ; // tableau de 10 cases
```

Fonctions :

```
function nomFonction(parametres) { ... }
```

Accès aux données de la page :

```
document.getElementById("idObjet")
```

```
document.getElementById(variableContenantUnId)
```

Outils prédéfinis :

Il existe plusieurs outils prédéfinis comme window, string, navigator, Math, Image...

DOM :

Le DOM (Document Object Model) est une spécification pour la structure et l'accès aux objets.

```
document.getElementById(unId).value //valeur brute du contenu de l'objet
```

```
document.getElementById(unId).innerHTML //valeur formatée du contenu de l'objet
```

```
document.getElementById(unId).style.unStyle //élément de style de l'objet
```

```
document.createElement(nomBalise) // crée un objet
```

```
unObjetConteneur.appendChild(unObjet) //insère un objet dans un autre
```

```
unObjetConteneur.removeChild(unObjet) //retire un objet d'un autre
```

## Séquence 6

# PHP : langage serveur

Cette séquence présente le langage serveur PHP, son intérêt, les points importants de sa syntaxe et son utilisation dans un site. Vous apprendrez aussi à intégrer des variables dans les fichiers CSS, ce qui n'est pas possible sans utiliser un langage serveur.

### ► Capacités attendues en début de séquence

Avoir de bonnes connaissances algorithmiques et sur le CSS (abordé dans les séquences précédentes).

### ► Contenu

1. Introduction .....	79
2. Positionnement du code .....	79
3. Éléments de syntaxe .....	82
4. Quelques outils prédéfinis .....	88
5. Optimisation du code .....	88

## 1. Introduction

A ce niveau, les connaissances abordées à travers le html, le CSS et le JavaScript permettent de créer des pages avec un contenu fixe (html), une présentation (css) et une flexibilité d'affichage et de tests (JavaScript).

### Pourquoi un langage serveur ?

Avec tous ces outils, les manipulations sont importantes mais limitées. Qu'est-ce qui ne peut –être fait ?

Le premier gros problème réside dans l'absence de possibilité de remplir la page avec des données variables, donc provenant du serveur. Rappelons que le site se trouve sur le serveur et que l'internaute télécharge la page en utilisant un navigateur. La page côté serveur peut très bien être dans un premier temps remplie par des données variables avant d'être envoyée côté client, à l'internaute. Ces données peuvent provenir de différentes sources : base de données, fichier XML, fichier texte... Ce premier problème rend un langage serveur indispensable si des données variables doivent être manipulées (par exemple des articles avec leurs caractéristiques, leurs prix, etc...).

Le second problème provient de la redondance d'informations dans la page, qui ne peut être évitée qu'en utilisant un langage pour aider à construire la page. Cette fois, le langage serveur n'a pas un caractère obligatoire mais représente une aide.

### Principe d'un langage serveur

Il existe plusieurs langages qui s'exécutent côté serveur. Le PHP est l'un des plus répandus pour son apparente facilité d'apprentissage (à nuancer) et aussi parce qu'il est gratuit. C'est donc le langage qui va être utilisé pour la suite de cet apprentissage. Cependant, globalement, le principe est le même pour les autres langages de type serveur.

Le PHP est un langage interprété, dont la syntaxe est basée sur le langage C (comme de nombreux langages). Le code PHP contenu dans une page est obligatoirement interprété côté serveur : le PHP n'apparaît pas côté client.

## 2. Positionnement du code

Comme pour le JavaScript, il existe plusieurs méthodes pour positionner le code PHP dans un site. Cependant, quelque soit la méthode, il y a 2 règles fondamentales à respecter :

- le code PHP doit obligatoirement être **mis dans un fichier avec l'extension ".PHP"**

Cela suppose donc que les pages qui feront appel à du PHP devront être renommées. Une page avec l'extension PHP est totalement capable de comprendre aussi le HTML ou le JavaScript.

- le code PHP doit être **entouré des balises <?PHP et ?>**

Voici les différentes possibilités de positionnement du code :

### Mixé au code de la page

La première méthode est la plus courante mais cependant la moins propre. Il est possible d'alterner entre le html et le PHP. On va commencer par un exemple simple, pour comprendre le principe. Avant tout, renommez le fichier index.html en index.php. Dans ce fichier, vous avez les 2 lignes de code suivantes pour déclarer les 2 calques des images du bandeau :

```
<div id="divImg0"></div>

<div id="divImg1"></div>
```

Remplacez ces 2 lignes par le code suivant :

```
<?php for ($k=0 ; $k<2 ; $k++) { ?>

    <div id="divImg<?php echo $k ; ?>"></div>

<?PHP } ?>
```

Quel est le principe ? Une boucle "for" est utilisée pour créer les 2 calques. Les syntaxes du PHP seront détaillées plus loin, cependant vous reconnaissez déjà une syntaxe similaire au C. La variable s'appelle \$k : nous verrons que les variables en PHP commencent toujours par le signe \$. \$k prend les valeurs 0 et 1. Après l'ouverture de l'accolade du "for", la balise du PHP se ferme. Pourquoi ? Parce que juste après c'est du code html qui doit être exécuté (la balise div). Mais remarquez que sur cette ligne de code, on retrouve encore un petit bout de PHP pour afficher la variable \$k : la fonction echo en PHP permet l'affichage. Du coup, le id de la balise div prendra respectivement les valeurs "divImg0" et "divImg1". Dernière ligne, les balises PHP sont à nouveau utilisées pour fermer la boucle "for" en mettant l'accolade fermante.



Faites un test pour voir si la page s'affiche toujours correctement, avec les 2 images du bandeau qui défilent. Regardez le code source de la page dans le navigateur (normalement en faisant en milieu de page, un clic droit et "afficher le source"). Là vous pouvez voir le code reçu côté client : il n'y a plus de PHP mais les 2 lignes html qui ont été générées avec le PHP (donc les 2 lignes de div).

Avec cette méthode, largement utilisée, donc que vous devez connaître, le code devient rapidement illisible car il y a trop de mélange et souvent des indentations difficiles à respecter. N'oubliez pas tout de même que vous pouvez à tout moment gérer des retours à la ligne ou des décalages avec des espaces sans que cela pose de problèmes au navigateur.

### Dominant sur le code de la page

Vous venez de voir que la fonction PHP "echo" permet d'afficher des informations. Pourquoi alors ne pas l'utiliser pour afficher aussi le code html plutôt que d'ouvrir et fermer sans arrêt les balises PHP ? C'est ce que font certains. Voici ce que cela peut donner sur notre petit exemple : remplacez le code que vous avez mis précédemment, par le code suivant :

```
<?php

for ($k=0 ; $k<2 ; $k++) {
```

```
    echo ('<div id="divImg'.$k.'"></div>') ;  
}  
?>
```

La fonction echo accepte du texte entre guillemets (") ou entre côtes ('). Ici, on a choisi les côtes car le texte à afficher contient déjà des guillemets. Pour gérer la concaténation entre le texte et la variable \$k, c'est le point qui est utilisé. En PHP, ce n'est pas le signe + mais le point qui permet de réaliser la concaténation de chaînes. Vous remarquez qu'en limitant le mixage des langages, le code est plus lisible, et mieux présenté.



Faites un nouveau test : normalement le résultat doit être exactement le même.

Certains n'aiment pas trop cette méthode car le code html est nettement moins visible, et de plus, dans les éditeurs colorisés, il n'est plus reconnu puisqu'il est considéré comme une simple chaîne de caractères.

### Dans un fichier séparé

Comme cela a été fait en JavaScript, il est possible de mettre le code PHP dans un fichier séparé, et de l'inclure en cas de besoin. En fait, le principe est basé sur le copier/coller : un fichier avec l'extension PHP peut contenir du code puis, pour l'intégrer dans une page, il faut utiliser l'instruction suivante :

```
<?php  
include('nomfic.php') ;  
?>
```

Le code complet contenu dans le fichier sera recopié à cet endroit de la page.

Mettre du code dans un fichier séparé est très intéressant pour éviter des répétitions (par exemple, si les entêtes de vos pages sont toutes identiques) mais aussi pour stocker des fonctions réutilisables dans plusieurs pages. Cette notion sera mise en pratique plus loin. Cependant, cela ne résout toujours pas le problème de la séparation totale du code PHP et du code html (comme on est bien arrivé à le faire en JavaScript).

### Avec des templates

Les templates offrent une alternative pour réellement isoler le code PHP. Elles sont cependant plus rarement utilisées. Vous les étudierez dans la séquence 11.



### 3. Éléments de syntaxe

Il est conseillé de récupérer l'indispensable manuel PHP "php\_manual\_fr.chm" que vous trouverez facilement sur internet (sur le site php.org). Ce manuel français est très complet et vous pourrez facilement retrouver toutes les informations manquantes. Ce cours y fera parfois référence.

Tout ce qui a été vu sur les bases du langage JavaScript au niveau syntaxe est aussi valable pour le PHP (respect de la casse, point-virgule en fin d'instruction, accolades, commentaires...).

#### Les variables

Toutes les variables en PHP doivent commencer par le signe \$. Si vous oubliez ce signe, l'interpréteur PHP pensera qu'il s'agit d'une fonction. Le nom de la variable est ensuite libre et respecte les règles classiques de nommage.

```
$nomVariable
```

Comme le JavaScript, le PHP est un langage non typé : le typage se fait à la volée, au moment de l'utilisation de la variable. Les types gérés par le PHP sont :

**int** ou **integer** (entier), **float** ou **double** (réel), **string** (chaîne), **boolean** ou **bool** (booléen).

Il est en revanche possible de transtyper une variable, avec la même syntaxe qu'en C :

```
(bool)$uneVariable    // transtypage en booléen
(int)$uneVariable      // transtypage en entier
(float)$uneVariable    // transtypage en réel
```

Le transtypage, comme dans tous les langages, peut se faire sur une variable ou une valeur. En PHP, il est implicite dans la plupart des cas, donc rarement utilisé. Par exemple, si vous concaténez à une chaîne une variable numérique, elle sera automatiquement transtypée en chaîne.

Les chaînes de caractères peuvent être entourées de guillemets ("...") ou de côtes ('...').

Il est possible de manipuler aussi des tableaux (**array**) et des objets (la notion d'objet ne sera pas abordée dans ce cours). Les tableaux sont très flexibles en PHP. Voici quelques exemples qui vont vous permettre de mieux comprendre :

```
// exemples de déclarations de tableaux

$tabVide = array() ;

$tabSimpleRempli = ("val1", "val2", "val3") ;

$tabAssociatifRempli = ("cle1" => "val1", "cle2" =>
"val2") ;
```

```
// exemples d'utilisation des tableaux
$tabVide[0] = "val1" ;
echo $tabSimpleRempli[1] ;           // affiche val2
echo $tabAssociatifRempli["cle1"] ; // affiche val1
foreach ($tabAssociatifRempli as $cle => $valeur) {
    echo "la case d'indice ".$cle." contient ".$valeur ;
}
```

Remarquez que, dans le fichier index.php, vous avez 4 calques pour les postits dont le code est très similaire. Essayez de remplacer cette partie de code (les 4 calques) par un bloc de code PHP dans lequel vous allez déclarer un tableau associatif (avec par exemple la clé "t-shirt" et la valeur "t-shirts personnalisés"), puis bouclez sur ce tableau pour afficher les balises nécessaires.



Faites un test pour contrôler que, malgré ces modifications, l'affichage des 4 calques se passe comme avant. Vous pouvez aussi contrôler le code source généré dans le navigateur : vous allez retrouver les 4 calques alors que dans le code PHP, vous avez fortement raccourci le code de la page.

### La portée des variables

Les variables locales sont celles qui sont initialisées dans une fonction : leurs portées se limitent à la fonction.

Les variables globales sont initialisées en dehors des fonctions. Mais attention, elles ne sont pas visibles dans les fonctions, sauf si vous les redéfinissez en global dans la fonction. Voici la démarche :

```
$var1 = "bonjour" ;
$var2 = "au revoir" ;
function test () {
    global $var1 ;
    echo $var1 ; // affiche bonjour
    echo $var2 ; // n'affiche rien
}
```

### Les constantes

Les constantes sont globales, définies par initialisation avec le mot "const" et, contrairement aux variables, ne sont pas préfixées par \$. Ce n'est pas une obligation mais classiquement les constantes sont écrites en majuscules. Voici un exemple de déclaration de constantes :

```
const UNECONSTANTE = "bonjour" ;  
const UNEAUTRECONSTANTE = 12 ;
```

## Les variables super globales

Le PHP offre un ensemble de variables appelées "super globales" qui donnent de nombreuses informations très utiles. Une super globale est toujours écrite en majuscule et comporte un nom de clé (c'est le principe des tableaux associatifs vus plus haut, où chaque case du tableau possède une clé comme indice, et une valeur). Par exemple, la variable `$_SERVER` qui contient des informations sur le serveur, possède entre autre la clé "HTTP\_USER\_AGENT". Pour accéder à la valeur correspondante, voici la syntaxe :

```
$_SERVER["HTTP_USER_AGENT"]
```

Les variables super globales sont accessibles partout et sont manipulables comme n'importe quelle variable excepté que leur contenu ne peut pas être modifié.

Vous pouvez retrouver la liste complète des super globales dans le manuel PHP. Nous verrons en pratique certaines de ces variables au fur et à mesure des besoins.

## La syntaxe du langage

La syntaxe du PHP est similaire à celle du C et des langages dérivés. Les structures conditionnelles (if, switch), itératives (while, do/while, for) et autres éléments de syntaxe comme les opérateurs arithmétiques et logiques sont totalement identiques. Si vous voulez plus de précisions à ce niveau là, référez-vous au manuel PHP que vous avez téléchargé.

## Les inclusions

Le code PHP peut être mis dans des fichiers séparés : par exemple il est possible, voire conseillé, de regrouper les fonctions par thème et de les mettre dans des fichiers séparés. Au moment où les fonctions sont nécessaires, il suffit alors d'inclure le fichier concerné dans le code en cours. Le contenu du fichier sera tout simplement copié à l'emplacement désiré. Voici la syntaxe :

```
include ("nomFic.php") ;
```

Pour éviter de faire plusieurs fois le même include, ce qui peut arriver et qui risque de provoquer des erreurs, il existe un autre ordre qui ne réalise l'inclusion que si elle n'a pas déjà été faite :

```
include_once ("nomFic.php") ;
```

On aura par la suite plusieurs fois l'occasion d'utiliser ces includes. Cependant, pour le moment, il est déjà possible de les utiliser pour éviter des répétitions de code de l'entête et du pied de page qui normalement se retrouvent sur toutes les pages. Cette technique est utilisée sur de nombreux sites.

Créez un fichier `head.php` et copiez dedans tout le début de la page `index.php`, jusqu'au code de la balise `div` du bandeau (non inclus). Le code doit donc contenir tout le head et le début du body avec la `div` du titre et la `div` du menu. Remplacez le code copié dans `index.php` par l'include nécessaire pour intégrer `head.php`.

Faites de même en copiant la fin du code index.php (à partir de la div pied jusqu'à la fin) dans un fichier foot.php et en remplaçant le code par un include.

Renommez toutes les autres pages du site avec l'extension PHP et intégrez dans toutes les autres pages les 2 includes pour l'entête et le pied de page. Dans head.php, pensez aussi à changer les extensions par PHP lors de l'appel des différentes pages dans le menu.



Faites un test : normalement la première page doit s'afficher correctement. Si vous naviguez dans les autres pages, vous devriez obtenir des pages qui comportent au moins l'entête et le pied de page.

Cette petite modification, qui n'a pris que très peu de temps, permet déjà d'avoir un squelette de présentation de l'ensemble du site.

Cependant vous avez peut-être remarqué un problème : quand les autres pages que la page index s'affichent, le navigateur signale des erreurs JavaScript. Cela vient du fait que dans le code JavaScript, on fait appel à des éléments qui n'existent pas sur les autres pages que la page index. Il faut donc faire en sorte que, lors du chargement d'une page, seules les fonctions qui concernent cette page soient chargées. Retournez dans boutique.js, fonction "window.onload" qui se charge dès le chargement de la page. Dès le début de cette fonction, il faut récupérer le nom de la page active. Voici les 2 lignes de code qui permettent de le faire :

```
// récupère l'url active  
  
var url = (document.location.href).split("/") ;  
  
// récupère le nom de la page dans l'url  
  
var mapage = (url[url.length-1].split(".")[0]) ;
```

Explication : url récupère l'adresse complète de la page, en découpant sur le caractère "/". Il suffit alors de récupérer la dernière case de url (qui va contenir par exemple "index.php") et de découper sur le "." pour récupérer la première case (donc dans l'exemple, la partie "index").

Il ne reste plus qu'à mettre tout le reste du code de la fonction, dans un test (si mapage == "index"). Comme il faudra aussi faire les comparaisons avec les autres pages, utilisez plutôt un switch et préparez tous les tests, même s'ils restent vides (excepté le break) pour les autres pages. Si vous ne savez pas utiliser le switch, allez voir dans le manuel PHP.



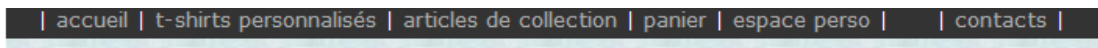
Refaites un test : normalement il n'y a plus d'erreurs JavaScript, quelque soit la page.

Pour faire encore plus propre, créez en dehors de la fonction window.onload une fonction spécifique par page du site. Dans le switch, appelez dans chaque cas la fonction correspondante. Pensez à bien transférer tout le code qui concerne la page index dans la fonction correspondante.



Refaites un test pour voir si tout fonctionne correctement.

Un autre point est un peu gênant : on n'a pas de moyen pour revenir à la page index dans le menu. Dans le menu, ajoutez en premier la rubrique "accueil" pour revenir à l'index et modifiez un peu le reste pour que tout rentre. Le but est d'obtenir un menu dans ce style :



Faites un test et vérifiez qu'en allant sur une autre page, cliquer sur l'accueil ramène sur la page initiale.

### L'accès aux données de la page

Le PHP intégré dans une page, accède aux données suivantes :

- ses propres variables PHP
- les variables reçues d'une page précédente (notion qui sera abordée dans la séquence 10 "transferts entre pages")
- les variables super globales

Vous ne pouvez pas accéder en PHP à des variables JavaScript ou html de la page active. Pourquoi ? Parce que le PHP s'exécute côté serveur, c'est donc un langage qui n'accède qu'aux informations côté serveur. La page se construit côté serveur, à l'aide du PHP : tant que la page est sur le serveur, le PHP peut travailler dessus. Mais ensuite, dès que la page est construite donc traduite en html/JavaScript, elle est envoyée au client. Une fois que la page est du côté du client, le PHP n'existe plus donc il ne peut plus rien faire, et certainement pas traiter ce que fait l'utilisateur sur la page (remplir une zone de texte) ni voir une variable JavaScript.

Nous verrons plus loin comment le PHP arrive tout de même à récupérer des informations, mais celles-ci devront obligatoirement transiter par le serveur.

### Les fonctions

En PHP, comme en JavaScript, tous les modules sont des fonctions. La différence réside dans la présence ou non de "return". Les fonctions ne sont pas explicitement typées : elles prennent le type de la valeur retournée. Une fonction peut avoir des paramètres qui, eux aussi, ne sont pas explicitement typés.

```
function essai ($param1, ..., $paramN) {
    // instructions
    return valeur ;    // optionnel
}
```

Par défaut, le passage de paramètres se fait par valeur. Il est possible de gérer un passage par adresse en ajoutant, comme en C, le signe & devant le nom du paramètre.

Les paramètres peuvent aussi être initialisés directement dans les parenthèses. Le paramètre prendra la valeur d'initialisation dans le cas où aucune valeur ne lui est attribuée lors de l'appel. Voici un exemple :

```
function afficheTexte ($texte, $couleur="black") {  
    echo '<p style="color:'. $couleur.'">'. $texte.'</p>' ;  
}
```

Voici 2 possibilités d'appel de cette fonction :

```
afficheTexte ("bonjour", "red") ; //affiche "bonjour" en  
rouge  
  
afficheTexte ("au revoir") ; // affiche "au revoir" en  
noir
```

Attention, les paramètres préremplis doivent être mis à la fin car ceux ne sont que les paramètres finaux qui peuvent être omis (un paramètre prérempli ne peut pas être suivi d'un paramètre à initialiser).

On va faire un premier essai d'écriture d'une petite fonction outil en PHP. Plus tard dans ce cours, vous apprendrez à organiser le code PHP dans plusieurs fichiers. Mais pour commencer, on va faire simple. Créez un dossier PHP et dans ce dossier, créez le fichier fonctions.php. Dans ce fichier, mettez les balises du PHP. Dans notre site, l'affichage d'espaces ("&nbsp;") a été plusieurs fois utilisé pour positionner certaines informations. Ecrivez la fonction espace qui reçoit le paramètre nbEspaces prérempli à 1. Cette fonction affiche autant d'espaces que demandé dans le paramètre. Dans la page head.php, juste avant la balise html, incluez le fichier fonctions.php. Dans le code de la page, remplacez les espaces forcés ("&nbsp;") par des appels à votre fonction : quand il n'y a qu'un espace à afficher, ne mettez pas de paramètre (puisque le paramètre est prérempli à 1) et quand il est nécessaire d'afficher plusieurs espaces, remplissez le paramètre.



Faites un test pour contrôler que rien n'a changé...

C'était juste un premier exemple, pas forcément d'une utilité révolutionnaire, mais vous avez au moins compris le principe d'un fichier contenant un ensemble de fonctions réutilisables et intégrables dans n'importe quelle page.

## 4. Quelques outils prédéfinis

Le PHP comporte de nombreuses fonctions prédéfinies que vous pouvez découvrir dans le manuel que vous avez téléchargé. Ces fonctions touchent tous les domaines : manipulation de chaînes, dates, fonctions mathématiques etc...

Juste pour tester, nous allons afficher la date du jour en haut à gauche du site, dans la barre de titre. Vous utiliserez le style "petitTexte" et la date doit être dans le format suivant : 01 Sep 2009 (par exemple) donc 2 chiffres pour le jour, 3 lettres pour le mois et 4 chiffres pour l'année. Dans le manuel PHP, allez voir la syntaxe de la fonction date(). Tout est expliqué.



Testez pour voir si vous obtenez bien l'affichage de la date en haut à gauche, dans le bon format, en petite écriture et en noir.

## 5. Optimisation du code

Vous l'avez compris, le PHP est un langage complet qui sert à plusieurs niveaux de la construction d'un site. L'aspect accès aux bases de données n'a pas encore été abordé, mais même sans cet aspect, le langage apparaît rapidement comme indispensable.

### Fonctions classées par thème

Vous avez commencé à utiliser des outils prédéfinis et vous avez créé une première petite fonction. Rapidement, vous aurez envie de créer vos propres outils, adaptés à vos besoins. Pour alléger dans un premier temps le code PHP inséré dans les pages, le but est justement de créer ces outils de façon séparée et de les intégrer par un include en début de chaque page. Vous avez déjà appris à créer un fichier fonctions.php, rangé dans un dossier PHP, pour intégrer une fonction réutilisable sur toutes les pages. On va aller plus loin dans l'organisation du code.

Dans le dossier PHP, créez un fichier init.php. Ce fichier contiendra la liste des includes de tous les fichiers de fonctions PHP que l'on va créer. Il suffira alors, en début de chaque page du site, de faire un include de init.php. Dans le dossier PHP, on va proprement créer un fichier par groupe de fonctions d'un même thème. Par exemple, nous aurons l'occasion de manipuler des fichiers, des images etc... et tous les outils d'un même thème seront alors regroupés dans un même fichier.

Renommez le fichier fonctions.php en chaines.php car il contiendra toutes les fonctions en rapport avec les chaînes.

Dans le fichier init.php, faites un include\_once de chaines.php. A chaque fois que vous aurez à créer un nouveau fichier dans le dossier PHP, il ne faudra pas oublier d'en faire l'include dans init.php.

Dans la page head.php, modifiez l'include en remplaçant PHP/fonctions.php par PHP/init.php.



Faites un test pour contrôler que rien n'a changé malgré les modifications de code.

A partir de là, on fera le maximum pour écrire des fonctions paramétrées qui allègent le code dans les pages.

### Optimisation du CSS

Vous avez remarqué que plusieurs informations sont redondantes dans le CSS. Certaines données sont dupliquées, ou dépendent d'autres données. Par exemple, les 4 mini-calques doivent tous faire la même taille. On a envie de changer les valeurs en dur du CSS par des variables. Normalement, le CSS n'accepte pas les variables. C'est le PHP qui va alors pouvoir résoudre le problème. Voici la démarche à suivre :

#### Transformer le css en PHP

Renommez le fichier principal.css en fichier cssPrincipal.php. Dans le fichier head.php, modifiez aussi le nom du css.

#### Indiquer au PHP qu'il contient du css

Tout au début du fichier cssPrincipal.php, insérez la ligne de code suivante :

```
<?php header('content-type:text/css') ?>
```

Cette ligne va indiquer que le contenu du fichier correspond bien à du css et sera interprété comme tel pour tout ce qui est hors des balises PHP.

#### Utiliser des variables

A partir de là, vous pouvez faire toute ce que vous voulez : vous pouvez dès le début du fichier, déclarer et initialiser des variables pour ensuite les utiliser dans le css, à la place de valeurs en dur, en faisant un echo de la variable entre balises PHP. Voici un exemple.

```
<?php header('content-type:text/css')

//--- tailles ---

$hPrincipal = 500 ;
$wPrincipal = 700 ;

...

?>

...

#divPrincipal{

    left:50% ;

    top:50% ;

    width:<?php echo $wPrincipal ?>px ;

    height:<?php echo $hPrincipal ?>px ;
```



```
margin-left:-<?php echo ($wPrincipal/2) ?>px ;  
margin-top:-<?php echo ($hPrincipal/2) ?>px ;  
...  
}  
...
```

Vous avez compris le principe ? Alors faites un nettoyage de tout votre fichier css pour qu'il n'y ait plus de valeurs en dur et surtout plus de valeurs dépendantes d'autres valeurs sans passer par des dépendances entre variables. Mettez bien toutes vos variables PHP en début de page.



Au cours des modifications, faites des tests réguliers pour voir que rien ne change dans l'aspect visuel. Tentez aussi de changer une variable dont plusieurs dépendent, pour voir les retombées à l'affichage. Remettez ensuite à cette variable la bonne valeur.

Enregistrez votre travail dans un dossier "version 6 PHP" sans oublier d'enregistrer les dossiers css, js et PHP.

## Synthèse

Le code PHP doit être placé entre balises `<?PHP et ?>`

Syntaxe du langage :

Similaire à la syntaxe du C.

Les variables :

Les noms des variables doivent commencer par \$.

Les déclarations sont implicites.

Les variables globales sont juste initialisées à l'extérieur des fonctions et redéclarées en global dans les fonctions où elles sont utilisées (global \$variable ;).

Les variables super globales sont prédéfinies et initialisées par le langage.

Les inclusions :

`include ("nomFic.php") ;` // permet d'inclure le contenu d'un fichier

`include_once("nomFic.php") ;` // idem mais uniquement s'il n'a pas déjà été inclus

Les fonctions :

`function nomFonction ($param1, ..., $paramN) { ...}`

Il est possible d'initialiser certains paramètres directement dans les parenthèses, par simple affectation.

CSS :

Un fichier CSS peut être transformé en fichier PHP et du coup peut intégrer des variables.

## Séquence 7

# XML et flux RSS

Cette séquence présente le format XML qui est de plus en plus répandu pour mémoriser des données. L'intérêt et l'utilisation des fichiers de ce format, les formats associés (DTD et XSL) sont expliqués. Vous apprendrez aussi à intégrer un flux RSS dans le site.

### ► Capacités attendues en début de séquence

Avoir des bonnes notions algorithmiques et HTML.

### ► Contenu

1. Qu'est-ce que le XML ? .....	92
2. Le fichier XML .....	93
3. Les parseurs de XML .....	95
4. Le DTD : fichier de structure .....	97
5. Le XSL : fichier de formatage .....	99
6. Flux RSS .....	101

## 1. Qu'est-ce que le XML ?

Jusque là, nous n'avons pas manipulé des données enregistrées sur le serveur. Toutes les informations sont "en dur" dans la page actuellement créée. Vous avez bien compris en abordant le PHP, que ce langage serveur va surtout permettre de manipuler les données enregistrées sur le serveur pour les intégrer dans les pages.

### Pour stocker quoi ?

Il existe plusieurs méthodes pour stocker des données sur le serveur. Le fichier XML représente l'une de ces méthodes. Un fichier XML est un fichier au format texte qui contient des données organisées dans des balises dont les noms sont choisis par le créateur du fichier.

**Avantages :** le format XML est un format standard sur le net, facile à lire puisqu'il est au format texte, léger pour les mêmes raisons et exploitable par quasiment tous les langages qui intègrent de plus en plus des commandes spécifiques au XML.

**Inconvénients :** le format XML n'accepte pas le SQL et ne gère pas les verrous (blocage en cas d'accès multiples sur la même ligne d'information). Pour ces raisons, il n'est pas adapté pour mémoriser des informations susceptibles d'être modifiées par plusieurs utilisateurs et/ou des informations devant être exploitées avec du SQL par une autre application.

Le format XML est idéal par exemple pour mémoriser des news (qui ne seront gérées que par l'administrateur du site et qui sont exploitées dans leur totalité pour l'affichage).

### Les fichiers qui accompagnent le XML

Sans que ce soit une obligation, un fichier XML peut être accompagné d'autres fichiers :

- **fichier DTD :** il permet de fixer la syntaxe du fichier XML. Si en début de fichier XML, vous précisez le fichier DTD correspondant, le fichier XML générera une erreur si sa syntaxe ne respecte pas le DTD
- **fichier XSL :** il permet d'exploiter le fichier XML pour gérer la présentation des données dans une page. Ce type de fichier est de moins en moins utilisé, essentiellement parce que les langages intègrent de plus en plus de fonctionnalités pour "parser" (exploiter) les fichiers XML

### Les éditeurs de XML

Puisqu'un fichier XML est au format texte, il est possible de le créer dans n'importe quel éditeur. Il existe cependant des éditeurs plus spécifiques, qui permettent de coloriser le code, de gérer les fichiers DTD, XSL, voire de tester l'ensemble des fichiers.

Pour en tester un, nous allons utiliser cooktop qui est gratuit et fonctionne sous Windows. Vous pouvez facilement récupérer cooktop sur internet, directement sur le site <http://www.xmlcooktop.com>. Au moment de la réalisation de ce dossier, j'ai récupéré la version 2.5.0.1204.

## 2. Le fichier XML

Le fichier XML contient des données entourées de balises personnalisées. L'indentation du code n'est pas obligatoire mais conseillée pour mieux visualiser la hiérarchie des informations et pour mieux repérer les éventuelles erreurs au niveau des balises.

### Création d'un fichier XML

Lancez Cooktop et choisissez dans les menus : File/New. Dans la petite fenêtre qui s'ouvre, choisissez "XML Cooktop" puis ok. Une fenêtre vierge s'ouvre, avec plusieurs onglets. Sélectionnez le premier, "source(XML)". Dans cette fenêtre, il va être possible d'écrire le code XML.

Le but est de créer un fichier XML pour notre site, afin d'y stocker les news. Voici le début du fichier, pour vous mettre sur la voie. La première news est donnée en détail : à vous de remplir les news suivantes (mettez-en quelques unes avec le contenu de votre choix). Attention, même si normalement les noms des balises sont libres, respectez les noms donnés ici : vous comprendrez plus tard pourquoi, quand on abordera les flux rss.

```
<news>

  <item>

    <title> </title>

    <description></description>

    <pubDate>2009-08-14</pubDate>

  </item>

  <item>

    ...


  </item>

  ...

</news>
```

Vous avez remarqué que l'éditeur colorise les balises, ce qui permet de mieux les repérer. Une fois le code écrit, pensez à le sauver, dans un dossier XML (que vous allez créer dans votre projet) et sous le nom news.xml.

### Vérification d'un fichier XML

Cooktop ne se contente pas de coloriser : il permet entre autre de contrôler le balisage. Remarquez  le bouton qui se trouve dans la barre du haut et qui porte le nom "Validate". Cliquez dessus : une petite fenêtre s'ouvre et se referme aussitôt. Cela prouve que tout va bien. Ce bouton contrôle qu'à toute balise ouvrante correspond une balise fermante, et que les imbrications entre balises sont correctes. Ajoutez une erreur dans une balise (par exemple en mettant

</ite> à la place d'un </item>) et cliquez à nouveau sur le bouton de contrôle. Cette fois la fenêtre s'ouvre précisant où se trouve l'erreur. Fermez la fenêtre et corrigez l'erreur. N'oubliez pas de sauvegarder.

### 3. Les parseurs de XML

La plupart des langages, dans leurs versions récentes, proposent des outils pour parser (exploiter) les fichiers XML.

#### XMLReader

Le PHP propose plusieurs outils permettant de parser les fichiers XML, dont un depuis la version 5, particulièrement facile et efficace : c'est la classe XMLReader. Allez dans la documentation du PHP et cherchez XMLReader, vous aurez un aperçu de l'étendue des possibilités de cet outil.

Dans notre site, nous allons l'utiliser pour afficher le contenu des news, qui ne seront donc plus "en dur" dans la page mais qui proviendront du fichier XML.

Allez dans la page index.php et effacez tout le contenu du <div> "divTextNews", c'est-à-dire tout le texte des news qui avait été mis en dur dans la page. Insérez les balises PHP pour mettre du code PHP à cet emplacement. A l'aide de XMLReader qui est directement intégré dans le php5, nous allons récupérer les informations contenues dans le fichier news.xml. Pour cela, commencez par écrire les 2 lignes de code suivantes :

```
$lesnews = new XMLReader() ;  
$lesnews->open("xml/news.xml") ;
```

Ces 2 premières lignes de code vont permettre de mieux comprendre comment manipuler un outil préexistant en PHP. La première ligne déclare une variable \$lesnews (le nom est libre) à partir de la classe XMLReader du PHP. A partir de là, il est possible d'utiliser tout le contenu de la classe, que vous avez vu dans la documentation, en passant par la variable \$lesnews. A chaque fois que l'on veut accéder à une propriété (une variable) ou une méthode (une fonction) de XMLReader, il faudra écrire \$lesnews suivi d'une flèche (->) suivi du nom de la propriété ou de la méthode. D'où la seconde ligne de code qui appelle la méthode open sur \$lesnews. Cette méthode attend en paramètre le nom du fichier, avec son chemin, qui contient le XML. Elle permet d'ouvrir ce fichier.

A partir de là, on aimerait récupérer les informations contenues dans le fichier XML et les présenter sous cette forme.



Donc la date doit apparaître en premier, suivi du signe ":" puis du titre en gras. La description doit s'afficher à la ligne et enfin une ligne est sautée avant la news suivante.

On partira du principe que dans le fichier XML, les news sont déjà triées sur la date, de la plus récente à la plus ancienne.

Un parseur XML parcourt un fichier dans l'ordre, donc compte tenu de la constitution de notre fichier XML, il va d'abord lire le titre et la description de la news avant de lire la date. Hors l'affichage de la date doit se faire en premier. Cela suppose que l'on ne va pas afficher au fur et à mesure de la lecture mais uniquement lorsque l'on passe à la news suivante. Pour cela, nous allons utiliser une variable qui va permettre de concaténer la chaîne à afficher. Il faut commencer par l'initialiser à vide :

```
$message = "" ;
```

Il faut ensuite boucler sur la lecture du fichier XML. La méthode `read()` permet de lire un nœud du fichier XML et d'avancer dans le fichier, un peu comme le principe d'un curseur SQL dans une base de données. La méthode `read()` retourne false si la lecture s'est mal passée, donc à priori si la fin du fichier a été atteinte. Du coup il est possible de lire directement dans le test de la boucle :

```
while ($lesnews->read()) {
```

Ainsi la boucle se terminera sur la fin du fichier.

Dans la boucle, il faut commencer par un test pour contrôler si le nœud sur lequel on est positionné est un nœud de début (ouverture de balise). Pour cela, il faut comparer la propriété `nodeType` qui donne le type du nœud, avec la constante `ELEMENT` de la classe `XMLReader` qui contient la valeur correspondante à un nœud de début. Voici la syntaxe (ce qui va vous permettre de voir comment on manipule une constante dans une classe) :

```
if ($lesnews->nodeType==XMLReader::ELEMENT) {
```

Dans ce test, il faut vérifier dans quel nœud on se trouve ("item", "title", "description", "pubDate"). Suivant le nœud, les actions seront différentes. Faites un switch sur la propriété `localName` pour tester le nom du nœud. Dans ce switch, testez chaque cas qui vient d'être cité. Si c'est un "item" c'est que l'on vient de passer à une nouvelle news : affichez `$message` (la news précédente) puis réinitialisez le avec une chaîne vide. Si c'est un "title", concaténez dans `$message` la chaîne suivante :

```
$message.="<strong>".$lesnews->readInnerXML()."</strong><br />";
```

La méthode `readInnerXML()` permet de récupérer le texte contenu dans le nœud concerné. Pour le titre, on a entouré le texte des balises `<strong>` permettant de mettre en gras. La balise `<br />`, vous la connaissez aussi, elle permet d'aller à la ligne.

Si c'est une "description", concaténez dans `$message` le texte du nœud suivi de 2 retours à la ligne. Enfin, si c'est un "pubDate", mettez dans `$message`, en premier, le texte du nœud, suivi de la chaîne " : ", suivi de `$message`.

Fermez le switch, le if et la boucle. La dernière news n'a pas été affichée : affichez la juste après la boucle. Fermez l'accès au fichier XML en utilisant la méthode `close()` sur `$lesnews`.



Faites un test : vous devriez voir défiler les news contenues dans le fichier XML, et avec une présentation quasiment similaire à celle qui a été montrée plus haut dans ce dossier.

### Problèmes d'affichage

Il y a tout de même 2 problèmes à résoudre : les accents (si vous avez mis des accents, vous remarquez qu'ils ne sont pas correctement affichés) et le format de la date (que l'on aimerait bien sous la forme jj/mm/aaaa).

#### Problèmes des accents :

C'est toujours ces fameux problèmes d'encodage. Le problème vient du fait que XMLReader parse obligatoirement au format utf-8. Il faut donc ensuite transformer ce qui a été récupéré pour qu'il soit au format iso-8859-1. Pour cela, entourez chaque `$lesnews->readInnerXML()` (sauf celui de la date) par la fonction `utf8_decode()` qui permet de faire cette transformation d'encodage.

#### Format de la date :

La date récupérée est sous forme de chaîne : il faut d'abord la transformer en date puis la formater pour obtenir le résultat voulu. La fonction `date_create` permet de réaliser la transformation en date, la fonction `date_format` permet de formater une date. Voici le code correspondant :

```
date_format(date_create($lesnews->readInnerXML()),  
            "d/m/Y")
```

Faites la modification dans votre code.



Refaites un test : contrôlez que les dates sont au bon format et que les accents s'affichent correctement.

## 4. Le DTD : fichier de structure

Le fichier DTD est optionnel, mais quand il est présent et surtout signalé en tête du fichier XML, il permet de définir un prototype pour contrôler la structure d'un ou de plusieurs fichiers XML.

### Création d'un fichier DTD

Par exemple, dans notre fichier XML, on veut qu'il y ait obligatoirement une structure ressemblant à ceci :

```
<news>  
  
  <item>  
  
    <title> ... </title>  
  
    <description> ... </description>
```



```

        <pubDate> ... </pubDate>

    </item>

    ...

</news>

```

Comme un fichier XML, un fichier DTD est un simple fichier texte qui peut être écrit dans n'importe quel éditeur. Bien sûr cooktop permet de créer des fichiers DTD. Dans cooktop, faites File/New et sélectionnez "DTD Editor". Dans la fenêtre qui s'ouvre, tapez le code suivant :

```

<!ELEMENT news (item+)>

<!ELEMENT item (title, description, pubDate)>

<!ELEMENT title (#PCDATA)>

<!ELEMENT description (#PCDATA)>

<!ELEMENT pubDate (#PCDATA)>

```

Comment ça marche ? La première ligne nous dit que le fichier XML contient une balise <news> qui va contenir plusieurs balises <item> (c'est le signe + qui symbolise le "plusieurs"). La seconde ligne dit que la balise <item> contient plusieurs balises de même niveau et qui apparaissent une seule fois : <title>, <description> et <pubDate>. Les lignes suivantes décrivent le contenu de ces sous-balises, qui ne contiennent pas d'autres balises mais juste du texte (#PCDATA).

Enregistrez le fichier sous le nom "news.DTD".

### Lien avec le fichier XML

Il faut maintenant que le fichier XML fasse appel au fichier DTD pour être contrôlé. Retournez dans le fichier XML et créez une ligne vierge en tout début de fichier. Allez dans le menu "Code Bits"/DTD/DOCTYPE/SYSTEM. Une ligne de code vient de s'ajouter. Remplacez l'étoile dans "\*.dtd" par "news" pour obtenir "news.dtd". Remplacez aussi "top\_element" par "news" car il faut préciser ici le nom de l'élément racine.

Sauvez et utilisez, comme précédemment, le bouton qui permet de contrôler le fichier XML (Validate). Normalement rien ne doit se passer puisque le fichier XML est correct et en accord avec le DTD. Supprimez une ligne complète dans le fichier XML (par exemple une ligne de description) et refaites le contrôle. Cette fois vous obtenez un message précisant qu'il y a désaccord avec le fichier DTD. Cette erreur serait passée inaperçue sans le DTD. Remettez la ligne supprimée et sauvez.

## 5. Le XSL : fichier de formatage

Les fichiers XSL permettent de formater l'affichage des informations contenues dans un fichier XML, afin de les intégrer dans une page html. L'utilisation de ce type de fichier est moins courante du fait de la création d'outils de plus en plus performants pour parser les fichiers XML, comme XMLReader que nous venons de découvrir.

Mais il y a quelques années, de tels parseurs n'existaient pas et le XSL était du coup très utilisé. Nous n'utiliserons pas de XSL dans le site, cependant pour votre culture générale, et pour que vous sachiez aussi décrypter un fichier XSL si vous devez un jour en lire un, voici le principe.

### Structure d'un fichier XSL

Cooktop permet d'intégrer facilement les lignes de code qui gèrent la structure d'un fichier XSL. Sur le fichier XML, cliquez sur l'onglet "stylesheet (xsl)". Sélectionnez le menu "Code Bits"/"XSL HTML Template". Normalement du code s'est automatiquement intégré dans la page. C'est la structure du fichier XSL. Enregistrez le fichier XSL sous le nom "news.xsl".

Voici quelques explications sur le code qui s'est généré.

Une première ligne, optionnelle, permet juste de préciser la version utilisée pour le XML.

```
<?XML version="1.0"?>
```

La seconde ligne permet de préciser la version xsl ainsi que l'origine du nom de préfixe donné à toutes les variables, pour éviter les confusions.

```
< xsl:stylesheet version="1.0" xmlns:xsl =  
"http://www.w3.org/1999/XSL/Transform ">
```

La ligne suivante précise que le but est de générer du html, et précise la méthode d'encodage (ici, par défaut, c'est l'encodage Windows qui est préconisé).

```
<xsl:output method = "html" encoding="Windows-1252" />
```

Puis, par défaut, le positionnement se fait sur la racine.

```
<xsl:template match="/">
```

On retrouve ensuite les balises classiques du html pour la structure d'une page.

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

Enfin il y a 2 balises de fermeture.

### Affichage dans un fichier XSL

Il est possible d'intégrer du code xsl afin de générer des balises html. Entre les balises body, ajoutez le code suivant :

```

<body>

  <xsl:for-each select="item">

    <xsl:value-of select="pubDate"/> :

    <strong><xsl:value-of select="title"/></strong><br />

    <xsl:value-of select="description"/><br /><br />

  </xsl:for-each>

</body>

```

Ce code va permettre de boucler sur chaque item (donc chaque news) et d'afficher, avec la même présentation déjà adoptée dans la page, la date, le titre et la description de chaque news.

Vous devez aussi modifier le match dans la balise template. Pour le moment il est à "/" et vous devez le positionner sur la toute première balise racine, donc "news".

Enregistrez le fichier et testez-le (toujours avec le même bouton Validate) pour voir si la syntaxe est correcte.

### Visualisation du résultat

Avant de pouvoir visualiser le résultat, il faut lier le fichier XML au fichier XSL en précisant dans le fichier XML qu'il doit utiliser un fichier XSL pour la présentation, un peu comme on a précisé qu'il devait utiliser un fichier DTD pour la syntaxe.

Dans le fichier XML, en deuxième ligne, insérez la ligne suivante :

```
<?xml-stylesheet type="text/xsl" href="news.xsl"?>
```

Enregistrez la modification.

Pour tester le résultat, vous pouvez le faire directement dans cooktop.



Cliquez sur le bouton XSLT. Normalement, dans le dernier onglet, vous devez voir apparaître les news correctement formatées (date suivi du titre en gras et de la description en dessous, avec un saut de ligne entre chaque news).

Mais le plus important est de contrôler qu'un navigateur classique est capable d'interpréter l'affichage.



Avec l'explorateur, allez directement dans le dossier XML et double cliquez sur le fichier news.xml. Un navigateur va s'ouvrir et vous devriez obtenir le même résultat d'affichage.

Le fichier XML a fait appel au fichier xsl pour s'afficher, plutôt que de s'afficher de façon simple.

### Bases de données au format XML


Il est possible de créer non pas l'équivalent d'une table, mais d'une base de données complète au format XML. Dans ce cas, des ajouts d'identifiants en attribut sont nécessaires et, avec le XSL, il est possible de lier les différentes "tables" sur

les identifiants. Cet aspect est cité à titre informatif. En effet, nous n'utiliserons pas cette possibilité qui n'a pas de réel intérêt et qui est très rarement utilisée. Une base de données complète nécessite des systèmes de protection (verrous) que n'offre pas le format XML. De plus, une base de données doit généralement être exploitée avec du SQL, ce qui n'est pas possible en XML. L'utilisation du XML pour enregistrer une base de données complète n'est donc à priori pas une bonne idée.

## 6. Flux RSS

Vous avez peut-être entendu parler des flux rss qui sont accessibles sur certains sites.

### Principe d'un flux RSS

Quand vous allez sur un site qui possède un flux rss, le navigateur le signale en affichant ce signe . Vous pouvez alors vous abonner au flux. Vous allez alors recevoir le contenu du flux sur votre navigateur sans avoir besoin de retourner sur le site concerné. Il suffit d'accéder, dans le navigateur, à la liste des abonnements aux différents flux. Nous allons voir un peu plus loin comment tester ces étapes.

### Format d'un flux RSS

Pour intégrer un flux rss sur un site, il faut avant tout créer un fichier au format rss. Un fichier de ce type est un fichier au format XML, respectant un dtd précis. Voici un exemple de contenu de fichier rss.

```
<rss version="2.0">

  <channel>

    <title>Mon site</title>

    <description>Ceci est un exemple de flux
RSS</description>

    <pubDate>2009-09-20</pubDate>

    <link>http://www.exemple.com</link>

    <item>

      <title>Actualité N°1</title>

      <description>Ceci est ma première
actualité</description>

      <pubDate>2009-09-25</pubDate>

    </item>

    <item>

      ...
```

```

        </item>
    </channel>
</rss>
</body>

```

Certaines balises sont obligatoires, d'autres optionnelles. Voici quelques explications :

Au début de la balise channel, on trouve plusieurs balises uniques. Dans ces balises, 3 sont obligatoires : title, description et link. Comme balises optionnelles, il y a : pubDate, image, language... Puis chaque balise item représente un item du flux rss. Il peut y avoir plusieurs balises item. La balise item doit au moins contenir les balises title et description. Les autres balises sont optionnelles. On peut trouver : link (pour mettre un lien), pubDate... Vous trouverez facilement sur internet toutes les informations qui concernent la structure d'un fichier rss.

Faites une copie de votre fichier news.xml et renommez cette copie en rss.xml. Dans ce fichier, enlevez les 2 premières lignes (concernant le dtd et le xsl) et remplacez-les par la première ligne donnée dans l'exemple concernant la version du rss. Ajoutez ensuite au début du fichier les lignes suivantes :

```

<channel>

    <title>Ma Boutique Perso</title>

    <description>news de maBoutiquePerso</description>

    <link>http://localhost/maBoutiquePerso</link>

    <pubDate>2007-09-15</pubDate>

    <language>fr</language>

    <image>

        <title>monSitePerso rss flux</title>

        <url>http://localhost/maBoutiquePerso/images/img.jpg</url>

        <link>http://localhost/maBoutiquePerso/images/img.jpg</link>

        <width>120</width>

        <height>150</height>

    </image>

```

Puis laissez à la suite vos balises item (qui sont déjà correctement formatées. N'oubliez pas en fin de fichier, de fermer la balise channel et la balise rss.

## Intégration d'un flux RSS dans une page

Pour que le fichier rss soit lié au site, et du coup accessible pour les internautes, voici la ligne de code à ajouter dans la balise d'entête de la page (donc dans le fichier head.php) :

```
<link rel="alternate" type="application/rss+xml"
title="rss de maBoutiquePerso" href="XML/rss.xml" />
```

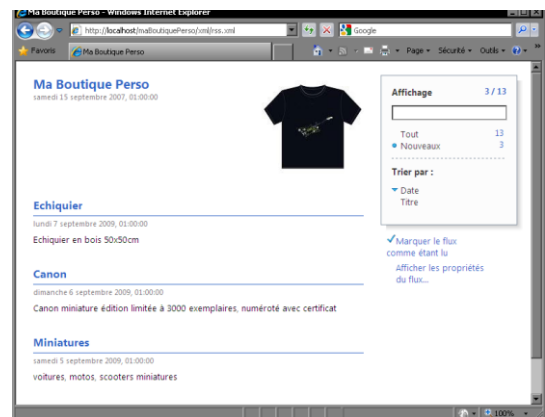
Il ne reste plus qu'à tester.



Lancez le site. Contrôlez que les news défilent toujours correctement. Cette fois, vous remarquerez l'icône orange du flux rss. Cliquez dessus : l'aperçu du flux doit normalement apparaître (avec la liste des news suivant un formatage spécifique aux flux rss) et il vous est proposé de vous abonner. Faites-le.

Normalement vous obtenez une page qui ressemble à ceci :

A tout moment, vous pourrez voir le résultat du flux sans aller sur le site. Faites un test en modifiant une information dans le fichier rss et en réactualisant la page : la modification doit apparaître.



Enregistrez tout votre travail dans un dossier nommé "version 7 XML" sans oublier les dossiers XML, css, js, PHP.

## Synthèse

### Fichier XML :

Le fichier XML est au format texte, contenant des informations balisées.

### Parseurs XML :

Les langages possèdent pour la plupart des outils prédéfinis permettant de gérer les fichiers XML (parcourir, mettre à jour), appelés "parseur XML".

Sous PHP, le parseur est XMLReader.

### Fichier DTD :

Il contient la syntaxe à respecter dans un fichier XML, au niveau du balisage.

### Fichier XSL :

Il permet de formater la présentation d'un fichier XML (souvent remplacé par l'utilisation d'un parseur).

### Flux RSS :

Fichier XML respectant un DTD officiel universel afin que tous les navigateurs et lecteurs de flux RSS soient capables de parser le fichier et en récupérer le contenu.

Pour intégrer un flux RSS dans une page, il faut ajouter la balise suivante dans le head :

```
<link rel="alternate" type="application/rss+xml" title="titre du flux" href="XML/rss.xml" />
```

## Séquence 8

# IMAGES ET OBJETS DYNAMIQUES

Cette séquence présente les possibilités au niveau de manipulation des images, des fichiers ainsi que la gestion évoluée des calques pour apporter de l'interactivité aux pages. Vous apprendrez à créer des calques, contenant des images, que l'utilisateur va pouvoir déplacer et redimensionner.

### ► Capacités attendues en début de séquence

Avoir de bonnes notions algorithmiques ainsi que des connaissances en JavaScript et PHP (abordées dans les séquences précédentes).

### ► Contenu

1. Récupération de fichiers .....	106
2. Calques dynamiques .....	110
3. Autres objets dynamiques .....	117
4. Fichiers d'images .....	118



## 1. Récupération de fichiers

### Introduction

Le PHP permet de manipuler des fichiers stockés côté serveur. Vous avez eu l'occasion de le voir pour des fichiers XML, c'est vrai aussi pour d'autres formats de fichiers.

Par exemple, il existe une batterie d'outils qui permettent de parcourir un dossier, de tester son contenu et de récupérer certains fichiers pour les traiter. Cet aspect est particulièrement intéressant pour les fichiers d'images car on peut ainsi récupérer des images pour les intégrer dans une page. En allant encore plus loin, il est même possible de récupérer des informations mémorisées derrière les images pour les exploiter dans la page.

Cette séquence va tenter de voir l'essentiel de ces différentes possibilités.

### Configuration

Mais avant tout, pour manipuler les images, vous devez correctement paramétrer PHP. Ce qui suppose que quand vous mettez au point un site, il faut se renseigner sur le paramétrage de l'hébergeur. Cependant j'ai cru comprendre que la plupart des hébergeurs étaient configurés pour accepter l'exploitation des fichiers d'images. En local, pour que cela marche, vous allez devoir le paramétrer. Cliquez (clic normal) sur l'icône de wamp. Dans config\_files, sélectionnez php.ini. Le fichier php.ini s'ouvre. Faites une recherche pour trouver "exif". Vous allez tomber sur la ligne suivante :

```
extension=php_exif.dll
```

normalement il y a un ";" en début de ligne. Si c'est le cas, enlevez juste le ";" (il est là pour mettre en commentaire la ligne). Ainsi la ligne sera prise en compte.

Ce n'est pas tout. Quelques lignes plus loin, vous devez trouver la ligne :

```
extension=php_mbstring.dll
```

Normalement par défaut il n'y a pas de ";" (s'il y en a un, enlevez-le) en revanche, le problème qui se pose est que cette ligne doit s'exécuter avant la ligne "extension=php\_exif.dll". Si elle est avant, tant mieux, mais il y a de grande chance pour qu'elle soit après. Dans ce cas, faites un couper/coller de la ligne "extension=php\_mbstring.dll" pour la placer avant "extension=php\_exif.dll".

Sauvez et fermez le fichier. Pour que ces modifications soient prises en compte, cliquez à nouveau sur wamp et faites un "restart all services".

### Création et transfert

Tous les fichiers copiés sur le serveur peuvent être exploités. PHP offre des outils de parcours de dossier pour récupérer les fichiers. Cette fonctionnalité est très pratique. Par exemple, pour la seconde page du site (création de t-shirts personnalisés), il est nécessaire de récupérer les images de la palette de couleurs (pour choisir la couleur du t-shirt), les images des t-shirts correspondants et enfin les images des éléments qui peuvent être ajoutés sur le t-shirt.

Comme les fichiers correspondants doivent être sur le serveur pour être exploitables, ils peuvent être montés en même temps que les pages du site et peuvent être modifiés à tout moment.

Normalement, pour notre site, vous avez déjà un dossier "Images" qui contient des sous dossiers comme "palette", "tshirt" et "ajouts". Allez rapidement regarder dans ces dossiers pour voir leur contenu.

Avant de continuer dans la manipulation des fichiers, il est temps de construire la structure de la deuxième page du site : la page des t-shirts personnalisés. Vous avez pour le moment une page avec au moins l'en-tête et le pied de page. Créez le code nécessaire (calques, css...) pour obtenir une présentation similaire à celle-ci :



Pour le calque du t-shirt (300x300px), intégrez l'image du t-shirt blanc directement dans le css, comme image par défaut. Elle sera changée par la suite, suivant le choix sur la palette. Le calque qui contient la palette de couleurs sera rempli dynamiquement avec les 9 couleurs (taille du calque : 60x60). Le calque qui contient les ajouts (taille du calque : 200x300) sera rempli dynamiquement. Il doit être en "overflow:auto" (pour intégrer un ascenseur si nécessaire). Le calque de droite contiendra le résultat des choix réalisés (couleur et liste des éléments ajoutés). Il sera aussi rempli dynamiquement. Faites en sorte que la souris se transforme en doigt pointé sur le survol du calque de la palette. Au niveau du css, vous pouvez aussi faire les choses plus proprement en créant un fichier par page (cssIndex.php, cssTshirt.php...) et en ne mettant dans ces fichiers que les styles spécifiques à la page. Du coup, dans le fichier cssPrincipal, vous ne mettez que les variables et styles communs, et en fin de fichier, vous faites les include nécessaires pour aller chercher les autres fichiers css.



Faites un test pour contrôler que l'affichage correspond à celui attendu (avec les calques de la palette, des ajouts et des choix vident pour le moment).

## Récupération des fichiers

### Le calque de la palette

Pour remplir le calque de la palette, il faut récupérer les images dans le dossier palette.

Dans la page tshirt.php, div "divPalette", ouvrez les balises PHP. Comment parcourir un dossier et récupérer des fichiers ? Voici quelques fonctions PHP bien utiles :

```
// récupère dans la variable $flux le flux d'accès au
contenu d'un dossier se trouvant à l'emplacement unChemin
:

$flux = dir($unChemin) ;

// récupère le nom du fichier suivant du flux (donc avance
dans le dossier à chaque lecture d'un fichier) :

$fic = $flux->read() ;

// cette fonction retourne false si la lecture a échoué
(donc s'il n'y a plus de fichier à lire dans le dossier)

// ferme le flux

$flux->close() ;
```

Sachant cela, faites une boucle pour récupérer les fichiers qui se trouvent dans "images/palette/" et à chaque fichier récupéré, contrôlez d'abord que le fichier ne porte pas le nom "." ou ".." (ceux sont les 2 fichiers créés par défaut dans tous les dossiers et qui permettent de remonter au dossier parent ou à la racine). Contrôlez aussi que le fichier est bien de type image. Pour cela, utilisez la fonction suivante :

```
// fonction booléenne qui permet de savoir si un fichier
est une image (attention, il faut donner le nom complet,
avec le chemin)

exif_imagetype($nomcomplet)
```

Si le fichier récupéré n'est pas ".", ".." et si c'est bien un fichier d'image, alors récupérez la première partie du nom dans la variable \$couleur. Pour cela, aidez-vous des fonctions suivantes (car il faut récupérer le début du nom, sans le point et l'extension qui font 4 caractères de long) :

```
strlen($uneChaine) // retourne la longueur d'une chaîne

substr($uneChaine, $depart, $longueur) /* retourne une
sous-chaîne de la chaîne, à partir de la position depart
et d'une longueur fixée avec le dernier paramètre*/
```

Dans une variable \$lescouleurs, correctement initialisée avant la boucle, concaténez une chaîne contenant la construction de la balise html "img" avec pour id le nom de la couleur, pour src le nom complet (chemin compris) du fichier et pour alt (message) le nom de la couleur.

Après la boucle, fermez le flux et faites juste un echo de la variable \$lescouleurs.



Faites un test : la palette s'est normalement remplie des 9 couleurs. Quand vous passez la souris dessus, elle change de pointeur et vous voyez apparaître dans la bulle, le nom de la couleur. En revanche il ne se passe pour le moment rien sur le clic d'une couleur.

Les événements vont être gérés avec la même logique que dans la page index, c'est-à-dire directement dans le fichier js, donc sans mélanger les codes.

Puisqu'on est dans la page tshirt et que le code que l'on doit écrire doit s'exécuter dès le chargement de la page, placez-vous dans window.onload et dans le test de la page tshirt (ou dans la fonction dédiée à la page tshirt, si vous avez fait des fonctions séparées).

Là, on se trouve confronté à un petit problème : le but est d'avoir autant de fonction événementielle sur le clic d'une couleur qu'il y a de couleurs. Logiquement, on devrait avoir neuf fonctions événementielles dans ce genre :

```
document.getElementById("blanc").onclick = function()
{...}

document.getElementById("rouge").onclick = function()
{...}

...
```

Bien sûr, il n'est pas question que l'on fasse ainsi. JavaScript accepte que des fonctions soient définies dans des boucles. Du coup tout est réglé : déclarez un tableau couleur contenant les 9 couleurs de la palette. Faites une boucle de 0 à 8 contenant la déclaration de la fonction événementielle sur le clic d'une des couleurs de la palette. Pour pouvoir tester le bon fonctionnement, insérez dans la fonction événementielle une alerte pour afficher l'indice de la boucle (en utilisant la fonction JavaScript alert).



Faites un test. Normalement, sur le clic d'une couleur de la palette, la boîte d'alerte doit apparaître, mais avec toujours la valeur 9.

On s'attendait à avoir le numéro qui correspond à la position de la couleur dans le tableau. Alors pourquoi tout le temps 9 ? Parce que lors de l'appel de la fonction événementielle, toutes les fonctions événementielles ont été construites et l'indice a parcouru toutes les valeurs de 0 à 8. Du coup, au moment de l'appel, l'indice vaut tout le temps 9. Ce petit problème va nous forcer à un peu jongler dans la suite du code.

Enlevez la ligne alert : elle sera remplacée par la suite par du code pour changer la couleur du t-shirt.

Cette capacité à créer des fonctions par programme est absolument énorme : cela va ouvrir des tas de perspectives que l'on va user et abuser.

### Le calque des ajouts

Puisque vous avez su remplir le calque de la palette, en suivant la même logique, remplissez le calque des ajouts dans `tshirt.php`. Petite différence : lors de la construction de la balise `img`, mettez en id la concaténation entre "ajout" et un numéro correspondant au numéro d'ajout (en démarrant à 0). Fixez la hauteur et la largeur à 60 (pour cela, créez un style `imgAjout` et affectez-le à la balise).



Faites un test pour contrôler que dans le calque des ajouts apparaissent bien les images des ajouts. Vérifiez que l'ascenseur s'affiche et que vous pouvez descendre pour voir tous les ajouts.

Commencez aussi à créer, dans le fichier `js`, les fonctions événementielles vides pour les ajouts correspondants. Cette fois, contrairement à la palette où le nombre de couleurs était connu (9) et vous avait poussé à faire une boucle "pour", vous allez devoir faire un "tantque l'objet existe". Sachant que les id des objets d'ajouts sont construits avec "ajout" + un numéro partant de 0, il suffit de faire une boucle comme ceci :

```
k = 0 ;

while (document.getElementById("ajout"+k)) {

    ...

}
```

Vous mettez ensuite dans la boucle les différentes fonctions événementielles. On aura besoin des événements suivants : `onclick`, `onmouseover` (survol de la souris) et `onmouseout` (la souris sort de la zone). Ecrivez donc les fonctions vides correspondantes.

## 2. Calques dynamiques

### Événements possibles sur les calques

De nombreuses possibilités sont offertes avec les calques. Il est possible de modifier son contenu, de le rendre visible ou invisible, de le créer ou au contraire le détruire. Enfin, de l'interactivité peut être ajoutée en permettant le déplacement et le redimensionnement d'un calque par l'utilisateur. Voyons toutes ces possibilités.

### Modification du contenu d'un calque

Lors du clic sur une couleur de la palette, il faut afficher le t-shirt de la bonne couleur dans le calque `tshirt`. Positionnez vous dans la bonne fonction événementielle et ajoutez la ligne de code suivante :

```
document.getElementById("divTshirt").style.backgroundImage
= 'url("images/tshirts/'+this.id+'.jpg")' ;
```

Cette ligne permet de modifier l'image de fond du calque divTshirt. La nouvelle image est construite à partir de l'id de l'ajout sur lequel on a cliqué. Retenez bien cela :

**this = OBJET ACTUEL CONCERNE**

Ceci est bien pratique car autrement, on n'aurait pas pu accéder à l'objet de l'ajout sur lequel on vient de cliquer (car le document.getElementById("ajout"+k) ne donnerait pas le bon ajout : rappelez vous la remarque qui a été faite plus haut sur la valeur du k).



Faites un test pour contrôler que la couleur du t-shirt change en fonction de la couleur sélectionnée dans la palette (et bien sûr que les couleurs correspondent).

### Visualisation d'un calque

Il est possible de rendre visible ou invisible un calque. Pour tester cet effet, on aimerait que sur le survol de la souris sur une des images des ajouts, un calque s'affiche pour montrer l'image en un peu plus grand. Le positionnement du calque se fera légèrement décalé par rapport à l'ajout concerné. Bien sûr, ce calque doit disparaître dès que la souris sort de la zone de l'ajout.



Puisqu'à chaque fois on ne va travailler qu'avec un seul calque de visualisation, on va pouvoir le créer "en dur". Dans la page tshirt, après la fermeture du calque "divAjouts", créez le calque vide "divVisuel". Dans le css, faites en sorte que ce calque ait une largeur et une hauteur de 120, et mettez lui la couleur de fond noir (qui apparaîtra en particulier en fond des images transparentes).

Dans le fichier js, au début de la partie portant sur la page "tshirt", pensez à rendre invisible ce calque. Maintenant, on va remplir la fonction événementielle sur le survol d'un ajout (onmouseover). Dans cette fonction, forcez le pointeur de la souris à être en forme de main (attention, n'oubliez pas d'utiliser this pour accéder à l'élément actuel). Rendez visible le calque "visuel" (avec la propriété de style "visibility" en lui affectant "visible"). Mettez en fond d'image l'image dont l'url est contenue dans la propriété src de l'ajout actuel (attention à la syntaxe du backgroundImage : allez voir si nécessaire dans le css). Affectez au style.top et style.left la valeur "1px". A priori le calque va toujours apparaître au même endroit, c'est-à-dire dans le coin haut gauche.

Maintenant, dans la fonction événementielle onmouseout (lorsque la souris sort de la zone de l'ajout), rendez invisible le calque "visuel" en affectant "hidden" à la propriété visibility. Pensez aussi à mettre la même ligne de code, pour rendre invisible le calque dès le début, juste avant la boucle sur les ajouts.



Faites un test pour voir si, au survol d'un des ajouts, vous voyez apparaître en haut à gauche et en plus grand, la même image, toujours sur fond noir. Regardez si cela marche avec tous les ajouts.

Il reste à apporter une petite modification : la position de l'image n'est pas correcte. Il faut qu'elle apparaisse légèrement plus bas et plus à droite de l'ajout correspondant. Pour cela, il faut connaître la position du calque de l'ajout par rapport au navigateur. Mais ce n'est pas si simple : la position d'un calque n'est pas en fonction du navigateur mais en fonction du calque conteneur, ainsi de suite jusqu'au navigateur. Du coup, il est nécessaire, pour obtenir la position d'un calque, de faire une fonction récursive pour monter à chaque fois dans les calques conteneurs, en additionnant les positions. Ceci est valable aussi bien pour la hauteur que pour la largeur. On vous donne les 2 fonctions récursives à ajouter à la fin de votre fichier js, et surtout en dehors de toute autre fonction (elles ne se déclenchent pas sur un événement) :

```
function getLeft(monObjet) {
    if (monObjet.offsetParent) {
        return
        (monObjet.offsetLeft+getLeft(monObjet.offsetParent));
    }else{
        return (monObjet.offsetLeft) ;
    }
}
```

Avec la même logique, écrivez aussi, à la suite, la fonction getTop.

Revenez dans la fonction événementielle onmouseover sur les ajouts. Remplacez les valeurs du top et du left par l'appel de la fonction getTop ou getLeft (en envoyant l'objet actuel en paramètre, donc this), moins la position du calque principal plus 40 (pour obtenir un petit décalage par rapport à l'image d'origine). N'oubliez pas d'entourer tout votre calcul d'une parenthèse pour concaténer à la fin, la chaîne "px".



Faites un test pour voir si, au survol d'un des ajouts, la zone de visualisation apparaît bien proche de l'ajout (un peu plus bas et un peu plus à droite) comme on peut le voir sur l'exemple.

Si vous avez un problème de superposition de calques (le calque de visualisation qui passe sous le calque de résultat), alors pensez à affecter une valeur assez grande (10, par exemple) à la propriété z-index du style du calque de visualisation.

Il y a encore un problème au niveau de l'ascenseur. Si vous descendez un peu et que vous tentez un survol, vous allez voir que le décalage vertical est plus important. Il faut donc tenir aussi compte du scroll du calque "divAjouts". Vous pouvez récupérer ce décalage avec scrollTop et scrollLeft sur le calque concerné (divAjouts). Donc, dans vos 2 calculs précédents, enlevez cette valeur.



Faites à nouveau un test pour voir si, même en utilisant l'ascenseur, la visualisation se positionne correctement.



### Ajout d'un calque

Allons plus loin. Mieux que de faire apparaître et disparaître un calque, il est possible de le créer et de le supprimer. En fait, ceci est possible pour tous les objets du html, comme vous l'avez appris (sans le pratiquer), dans la séquence 5 "JavaScript : langage client", partie "DOM". Pour la suite, il faut que vous retourniez dans cette séquence pour voir les syntaxes.

Dans le fichier js, positionnez-vous dans la fonction événementielle du clic sur un ajout. Dans cette fonction, plusieurs choses sont à faire. On pourrait se contenter de créer un calque que l'on positionne sur le t-shirt, et de mettre l'image de l'ajout en fond de l'image du calque. Mais dans la perspective de pouvoir redimensionner le calque et donc l'image, cette solution n'est pas viable. En effet, une image de fond de calque n'est pas redimensionnée avec le calque. Il faut donc dans un premier temps créer une image (balise img) pour intégrer l'image, puis créer un calque et intégrer la balise img dans le calque sans oublier de la dimensionner à 100% en hauteur et largeur afin que l'image se redimensionne avec le calque. Voici en détail les étapes à suivre :

#### Création de l'image

Créez un objet `nouvImage` à partir d'une balise "img" (voir la syntaxe dans la séquence JavaScript, partie DOM). Affectez-lui l'image correspondant à l'ajout. Mettez sa taille (`style.width` et `style.height`) à "100%".

#### Création du calque

Créez un objet `nouvCalque` à partir d'une balise "div". Définissez sa hauteur et sa largeur à 60. Positionnez le (`top` et `left`) à 120 pour le centrer sur le t-shirt (attention, n'oubliez pas qu'à chaque fois que vous affectez une valeur, et non un pourcentage, au `top`, `left`, `width` ou `height`, il faut concaténer "px" à la fin, sinon vous allez avoir des surprises sous certains navigateurs, en particulier firefox). Modifiez l'aspect du curseur (pour avoir le doigt pointé). Donnez-lui comme id la concaténation de "el" (pour "element") et de l'id de l'ajout actuel.

#### Intégration des différents objets

Par le code, intégrez l'image `nouvImage` dans le calque `nouvCalque` puis, intégrez ce calque dans le calque portant l'id "tshirt" (encore une fois, la syntaxe est dans la séquence JavaScript).



Faites un test pour voir si, sur le clic d'un ajout, celui-ci vient bien s'ajouter au milieu du t-shirt avec la bonne image. Choisissez ainsi plusieurs ajouts pour voir s'ils s'ajoutent correctement. Bien sûr pour le moment ils sont superposés.

### Suppression d'un calque

Avant d'attaquer la partie la plus complexe, le déplacement et redimensionnement d'un calque, voici comment supprimer facilement un calque à la demande de l'utilisateur. Vous avez remarqué dans le petit texte explicatif qui s'affiche en haut de la page de la création de t-shirts, qu'il est mentionné qu'un double clic permettra de supprimer un ajout sur le t-shirt. Il ne reste plus qu'à le programmer.

Cette fois, c'est un événement directement sur le calque ajouté sur le tshirt. Où écrire le code de cet événement, donc la procédure événementielle ? Et bien, juste après avoir créé le calque ! Donc après la création du calque `nouvCalque` et son



paramétrage, écrivez la fonction événementielle suivante (qui se déclenchera sur le double clic sur le calque de l'élément du t-shirt) :

```
nouvCalque.ondblclick = function() {
    ...
}
```

Dans cette fonction, supprimez l'objet actuel (this) du calque dont l'id est "divTshirt".



Faites un test pour voir si vous pouvez ajouter des éléments au t-shirt, et les supprimer en double cliquant directement sur l'élément du t-shirt.

### Déplacement d'un calque

Il faut pouvoir positionner l'élément où l'on veut sur le t-shirt. Comment cela va-t-il fonctionner ? La personne doit cliquer sur l'élément pour le sélectionner, garder le clic enfoncé pour déplacer l'élément puis lâcher le bouton pour poser l'élément au nouvel emplacement.

On a donc besoin de plusieurs événements : `onmousedown`, `onmousemove` et `onmouseup`. L'événement `onmousedown` est un événement directement sur l'objet pour savoir lequel est sélectionné. Vous pouvez donc tout de suite écrire cette fonction événementielle vide, sur l'objet `nouvCalque`, à la suite de la fonction `nouvCalque.ondblclick`. En ce qui concerne les 2 autres événements, ils concernent une zone plus large, sinon, dès que la souris va sortir de l'image suite à un déplacement un peu rapide, le calque va s'arrêter de bouger. Donc les 2 événements seront mis plutôt sur le calque principal. Donc, en dehors de la grande boucle `while` sur les ajouts, vous pouvez tout de suite écrire les 2 fonctions événementielles, pour le moment vides, concernant le `onmousemove` et `onmouseup` sur le calque principal.

Plus tard, lors de vos tests, vous remarquerez que le glisser/déplacer se passe très bien sous certains navigateurs (opera) et de façon un peu plus surprenante sous d'autres (le bouton doit être lâché pour que le déplacement fonctionne sous IE et Firefox, cependant ça marche).

### Les variables nécessaires dans les 2 fonctions

Un premier problème se pose : il va falloir mémoriser à chaque fois l'ancienne position de la souris pour savoir de combien a été le décalage pour repositionner le calque. Cette information est nécessaire dans plusieurs procédures événementielles, donc les variables correspondantes doivent être déclarées un niveau au dessus. Comme un seul calque peut être modifié à la fois, inutile de rattacher ces variables à chaque calque. Du coup, dans le code du fichier js, positionnez-vous juste avant la boucle `while` sur les calques d'ajout. A cet endroit, déclarez (avec `var`) les variables `sourisX` et `sourisY`. Il faut aussi mémoriser l'objet sélectionné : déclarez une variable `selection` que vous initialisez à `null` (qui signifie "aucun objet").

### `onmousedown` : sélection d'un calque

Avant de commencer à écrire le code de cette fonction, on a un problème à résoudre. Les événements de la souris (qui vont devoir être gérés pour récupérer sa position) ne se gèrent pas de la même façon sur chaque navigateur. Ça aurait été trop simple... Par exemple, sous IE il existe l'objet event qui est déjà configuré pour capturer ces événements, alors que ce n'est pas le cas sous firefox. Voilà comment résoudre le problème à chaque fois que vous devez capturer un événement souris (ou clavier) dans une fonction événementielle :

- ajoutez le paramètre event dans la fonction (pour qu'un navigateur comme firefox puisse récupérer l'événement)
- commencez par tester si window.event existe (c'est le cas sous IE), si c'est le cas alors vous l'affectez dans event

Voilà ce que cela doit donner au niveau code :

```
nomObjet.nomEvenement = function(event) {
    if (window.event) {event=window.event ;}
    ...
}
```

Pensez à utiliser ce code pour toutes vos fonctions événementielles qui ont besoin de gérer event.

Revenons donc à la fonction de l'événement onmousedown. Faites la modification précédente. Ensuite, il faut récupérer l'objet sélectionné : affectez l'objet actuel (this) dans la variable selection.

Il faut aussi récupérer les coordonnées de la souris : affectez à sourisX et sourisY les coordonnées obtenues avec event.clientX et event.clientY.

### **onmousemove : déplacement**

Dans cette fonction, n'oubliez pas de commencer par la gestion de event (comme expliqué plus haut).

Il faut ensuite vérifier que le calque a bien été sélectionné : contrôlez que selection ne contient pas null.

Dans ce cas, modifiez la propriété style.top du calque actuel (this) en lui affectant son top actuel (récupérable avec offsetTop) + le déplacement de la souris (position actuelle – ancienne position qui se trouve dans sourisY). Avec la même logique, modifiez la propriété left.

Pensez ensuite à réinitialiser les variables sourisX et sourisY avec la nouvelle position de la souris.

### **onmouseup : l'objet est posé**

Dans cette fonction, il suffit d'affecter null à la variable selection.



Faites un test pour voir si vous arrivez à sélectionner un élément sur le tshirt et à la déplacer (pour cela, si le drag and drop classique ne marche pas, cliquez sur l'élément sans garder le clic enfoncé, bougez la souris, normalement l'élément va se déplacer, enfin cliquez à nouveau pour le poser).

Par précaution, pensez à remettre selection à null lors d'un double clic sur un élément.

### Redimensionnement d'un calque

Les choses se compliquent pour le redimensionnement. D'abord, il faut repérer si la personne a cliqué dans une petite zone du coin bas droit du calque. Pourquoi ? Car généralement c'est ainsi que l'on redimensionne : on pourrait d'ailleurs changer l'aspect de la souris quand celle-ci rentre dans cette zone.

#### Variable nécessaire

On a besoin d'une autre variable accessible dans les 2 fonctions. Cette variable booléenne permettra de savoir si on a cliqué dans le coin bas droit. Juste après la déclaration de sourisX et sourisY, déclarez redim et initialisez le à false.

#### Modification de onmousedown

Revenez dans l'événement clic et, dans le "si", après avoir récupéré les coordonnées de la souris, on va remplir redim en fonction de la position de la souris. Le calcul est un peu complexe car il faut prendre en compte la position de la souris (qui est en fonction du navigateur) par rapport à la position du calque (qui lui est en fonction du calque parent). Du coup, les fonctions getTop et getLeft qu'on a déjà eu l'occasion d'écrire vont nous resservir. Je vous donne directement le code à ajouter, pour éviter de perdre trop de temps à chercher. Cependant, essayez de le comprendre :

```
var leftbas=this.offsetWidth ;
var topbas=this.offsetHeight ;
var margeX = sourisX - getLeft(this) ;
var margeY = sourisY - getTop(this) ;

redim=(margeX<=leftbas+20 && margeX>=leftbas-20 &&
margeY<=topbas+20 && margeY>=topbas-20);
```

On constitue donc un carré de 20 pixels dans le coin bas droit du calque. La variable redim prendra la valeur vraie si la souris a cliqué dans ce carré.

#### Modification de onmousemove

Dans cette fonction, cette fois il faut gérer les 2 possibilités : le déplacement de la souris provoque soit un déplacement, soit un redimensionnement. A l'intérieur du premier test, faites un second test pour contrôler si redim est à false. Si c'est le cas, alors placez les 2 lignes de code que vous aviez déjà écrites pour modifier le top et le left. Dans le cas contraire, avec la même logique modifiez le height et le width. Après la fermeture de ce test, laissez les 2 lignes de code qui remettent à jour la position de la souris.

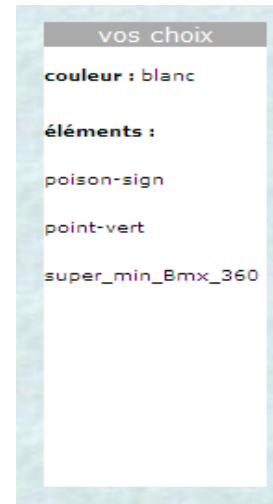


Faites un test pour contrôler le déplacement et le redimensionnement. Faites le test pour plusieurs éléments après en avoir ajouté plusieurs sur le t-shirt.

### 3. Autres objets dynamiques

Les calques ne sont pas les seuls objets qui peuvent être manipulés. Seuls les conteneurs sont déplaçables mais il est possible de modifier, créer et détruire de façon dynamique toutes sortes d'objets html.

Par exemple, dans le site, il faudrait s'occuper de remplir le calque de droite contenant les choix de la personne. Il faudrait y mentionner la couleur du t-shirt mais aussi la liste des ajouts. En fait, on voudrait obtenir un résultat similaire à ceci :



#### Modification d'un objet

Il n'est possible de choisir qu'une seule couleur pour le t-shirt. Donc autant prévoir déjà dans le calque l'objet graphique nécessaire pour recevoir cette information.

Dans la page tshirt, calque "divResultat", en dessous du div que vous avez du mettre pour le titre "votre choix", mettez un div en lui affectant le style petitTexte. A l'intérieur de ce div, mettez en dur le texte "couleur :" suivi d'une balise vide "label" avec comme id "lblCouleur". Cette balise sera remplie dynamiquement lors du clic sur la palette, cependant ce serait bien qu'elle soit remplie avec la couleur par défaut (blanc) dès l'ouverture de la page. On pourrait mettre le mot blanc en dur dans la balise mais il y a mieux : allez dans le fichier js, juste après avoir déclaré et rempli le tableau couleur. Affectez à la propriété innerHTML (le contenu html) de l'objet "lblCouleur" la valeur contenue dans la première case du tableau (ou la case qui contient "blanc").

Revenez dans la page tshirt, juste après la balise lblCouleur et mettez plusieurs balises de retour à la ligne puis mettez en dur le mot "éléments :". Ajoutez une balise "p" (paragraphe) avec comme id "pElements". Laissez cette balise vide. Elle sera remplie dynamiquement.

A quel moment le label lblCouleur doit-il être modifié ? A chaque fois qu'une couleur est choisie. Dans le fichier js, allez dans la fonction événementielle sur le clic de la palette. Dans cette fonction, ajoutez une ligne de code pour modifier la propriété innerHTML de lblCouleur, en lui affectant l'id de l'objet courant (this). En effet, l'id contient la couleur.



Faites un test pour contrôler que la couleur se marque bien dans le calque du choix, et qu'elle change lorsqu'une autre couleur est sélectionnée.

#### Ajout d'un objet

Il faut maintenant gérer l'ajout de nouveaux éléments sur le t-shirt. Lors de chaque ajout, une nouvelle ligne doit s'ajouter dans la liste de droite, donc dans la balise "p".

Dans le fichier js, fonction événementielle correspondant au clic sur un ajout, il faut s'occuper d'ajouter un élément dans la liste de droite : créez un élément nouvMessage en créant un élément de type "p" (on va imbriquer des balises "p" dans la balise "p" principale du calque). Ce nouvel élément, donnez lui comme id la concaténation de "pel" et de l'id de l'objet actuel. Modifier le contenu (innerHTML)

de ce nouvel élément en lui affectant la propriété alt de l'objet actuel. En effet, le alt contient bien le nom simple de l'ajout. Il ne reste plus qu'à intégrer nouvMessage dans le paragraphe "pElements".



Faites un test pour contrôler que chaque élément ajouté apparaît bien dans la liste de droite.

### Suppression d'un objet

Il faut aussi gérer la suppression des éléments. Dans le fichier js, fonction événementielle onclick sur le calque ajouté, supprimez le paragraphe d'indice "p" suivi de l'id de l'objet actuel, qui se trouve dans le paragraphe "pElements".



Faites un test pour contrôler que, lors de la suppression d'un élément, il disparaît bien de la liste. Essayez d'ajouter plusieurs éléments et de supprimer un élément intermédiaire pour voir s'il ne reste pas d'espace vide dans la liste.

### Limitation des ajouts

Pour finir, on voudrait juste limiter le nombre d'ajouts à 5 sur un t-shirt. Il faut donc une variable qui va compter le nombre d'éléments ajoutés. Dans le js, juste avant la boucle sur les ajouts, déclarez la variable totElements et initialisez là à 0.

Sur l'événement clic sur un ajout, contrôlez que totElements est inférieur à 5. Si c'est le cas, incrémentez la variable et faites tout le reste du code déjà écrit. Si ce n'est pas le cas, affichez un message d'alerte pour avertir qu'on ne peut pas insérer plus de 5 éléments. Pensez à décrémenter totElements lors de la suppression d'un élément.



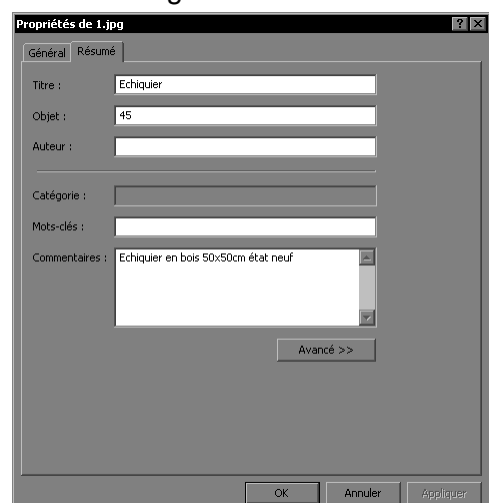
Faites un test pour contrôler que vous ne pouvez pas faire plus de 5 ajouts. Vérifiez que si vous en enlevez un, vous pouvez alors en ajouter un autre.

Enregistrez cette version sous "version 8.3 fichiers" sans oublier les dossiers PHP, css, js, XML.

## 4. Fichiers d'images

Vous avez déjà manipulé des fichiers d'images, mais vous n'avez pas encore exploré toutes les possibilités. Il est possible d'enregistrer des informations directement liées aux fichiers, et de les récupérer dans les pages avec le PHP. Attention, ceci n'est possible que pour les fichiers au format jpeg et tiff. On va utiliser cette possibilité pour la page de la boutique.

### Comment stocker des informations derrière un fichier ?



En utilisant l'explorer, allez dans le dossier articles qui se trouve dans le dossier images. Là se trouvent toutes les photos des articles qui seront proposés à la vente. Faites un clic droit sur l'un des fichiers et sélectionnez "propriétés". Une fenêtre s'ouvre. Vous avez différentes zones qui peuvent être remplies. Ces informations pourront ensuite être récupérées dans le programme.

Pour voir concrètement comment ça marche, on va utiliser cette possibilité pour les articles à vendre. Du coup, derrière chaque image des articles, il y aura le nom précis de l'article, le prix et une description détaillée.

Ce n'est pas forcément une bonne idée de mémoriser les articles de cette façon. Pourquoi ? Parce que les informations enregistrées derrière les images sont récupérables par programme mais non modifiables par programme. Imaginons que l'on veuille modifier le prix, il faudrait modifier l'information derrière l'image puis remonter l'image sur le serveur. Ce n'est pas très pratique. Donc c'est essentiellement pour l'aspect apprentissage de cette nouvelle notion que l'on va gérer les articles de cette façon.

Mais dans quel cas l'utilisation des informations, derrière des images, peut être intéressante ? Typiquement dans les galeries photos.

### Comment récupérer les informations avec le PHP ?

Voici en PHP les commandes qui permettent de récupérer les informations stockées derrière une image :

```
/* récupération, dans un tableau associatif, des
informations d'une image au format jpg ou tiff dont le nom
complet, chemin compris, se trouve dans le premier
paramètre */

$infos = exif_read_data($ficComplet, 0, true) ;

// récupération de certains éléments du tableau

$titre = $infos["IFD0"]["Title] ;

$sujet = $infos["IFD0"]["Subject] ;

$commentaire = $infos["IFD0"]["Comments] ;
```

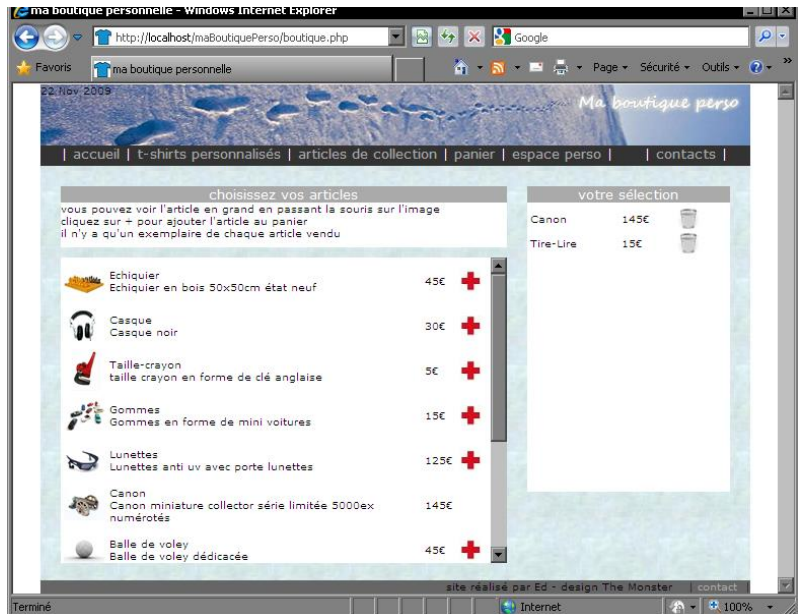
En réalité, il y a beaucoup plus d'informations qui sont récupérables. Allez faire un tour dans le manuel du PHP, commande `exif_read_data`, pour vous en persuader.

On va donc utiliser ces possibilités de récupération dans la page de la boutique qu'il va falloir maintenant construire.

### Création de la structure de la page

Comme pour la page tshirt, vous allez commencer par créer la structure de la page pour quelle ressemble à ceci :

Le calque central ("divArticles") va contenir la liste des articles. Il sera rempli dynamiquement en PHP. Le calque de droite ("divSelection") va contenir la liste des articles sélectionnés. Il sera rempli dynamiquement en JavaScript. Le calque du haut ("divExplications") fonctionne comme celui de la page tshirt et permet juste d'afficher des informations fixes.



Faites un test pour contrôler que vous obtenez bien un affichage correct.

### Calque des articles

Pour remplir le calque des articles, vous allez utiliser le même principe que pour la page tshirt, en récupérant cette fois les fichiers qui se trouvent dans le dossier "images/articles". Cependant il y a tout de même 2 différences. La première est que vous devez récupérer, pour chaque fichier, le titre, l'objet et le commentaire (le titre contient le nom de l'article, l'objet contient le prix et le commentaire contient une description plus détaillée). Utilisez `exif_read_data` pour récupérer ces informations et mettez les dans des variables. La seconde différence réside dans la construction du contenu du calque : le but est d'aligner les informations et pourquoi pas, en faisant un tableau. Cette solution reste assez pratique pour ce genre d'affichage.

Cela suppose qu'avant la boucle, il faudra initialiser une variable chaîne non pas à vide mais avec la balise `<table>` en lui donnant un id ("tabArticles"). Dans la boucle, concaténez dans cette variable une chaîne permettant de construire une ligne du tableau (une ligne par article) avec pour chaque ligne, plusieurs cellules :

- une cellule pour afficher l'image récupérée, au format 40x40 (affectez à l'attribut `alt`, le titre récupéré avec `exif_read_data`)
- une cellule pour afficher le titre suivi (après un retour à la ligne) du commentaire
- une cellule pour afficher le prix (mettez le prix dans une balise `<label>`)
- une cellule pour afficher l'image du "+" (`plus.jpg` qui se trouve dans `images`)

Il est conseillé d'affecter des styles (avec `class`) à chaque case et de gérer dans le css (faites un `cssBoutique.php` que vous intégrerez dans `cssPrincipal.php`) les



caractéristiques du tableau et des cellules. Par exemple, pour le tableau complet, mettez les bordures à 0, demandez un centrage vertical des cellules (valign:center) et mettez à largeur à 95% (pour laisser de la place à l'ascenseur). Pour chaque cellule, gérez l'alignement judicieusement ainsi que la largeur (prix à droite...).

Pour votre boucle sur les fichiers, utilisez un indice de boucle, par exemple \$k, et utilisez cet indice pour construire les id de l'image de l'objet ("img".\$k), du label ("lbl".\$k) et de l'image du plus ("ajout".\$k). Le fait de les repérer ainsi va permettre de les gérer en JavaScript. Après la boucle, n'oubliez pas de concaténer la balise pour fermer le tableau et d'afficher le contenu de la variable chaîne qui a été construite.



Faites un test pour contrôler que les articles s'affichent correctement.

Normalement sous IE, tout doit bien se passer. En revanche sous Firefox, vous devriez avoir des surprises...

Le problème vient du fait que les informations récupérées derrière les images ont un format un peu particulier : chaque lettre est suivie d'un caractère spécial que certains navigateurs arrivent à occulter mais pas d'autres. C'est probablement un problème d'encodage mais j'avoue ne pas avoir trouvé de fonction adaptée pour le résoudre. En l'absence de solution, j'ai créé une petite fonction qui tout simplement récupère un caractère sur 2. Du coup ça marche avec tous les navigateurs. Voici la fonction, à placer dans le fichier chaînes.php :

```
//--- Permet d'enlever un caractère sur 2 d'une chaîne ---  
function enleveUnSurDeux ($texte) {  
    $result = "" ;  
    $longueur = strlen($texte)-2 ;  
    for ($k=0; $k<$longueur; $k+=2) {  
        $result .= substr($texte, $k, 1) ;  
    }  
    return $result ;  
}
```

Utilisez cette fonction pour nettoyer vos 3 variables qui récupèrent le titre, le sujet et le commentaire.



Faites un test pour contrôler que cette fois les articles s'affichent correctement sous tous les navigateurs.

### Visualisation en grand de l'article

Pour visualiser un article en plus grand, sur le survol de son image en petit, le principe est le même que pour les ajouts sur le t-shirt. Vous allez donc suivre la même logique : il faut créer un calque de visualisation dans la page boutique et



gérer ce calque en JavaScript. Le code JavaScript sera écrit cette fois dans le "case" qui concerne la page boutique (ou dans la fonction correspondante, si vous en avez créée une). Pensez à rendre invisible le calque de visualisation puis faites une boucle tant qu'il existe des objets dont l'id est "img"+k (k étant un indice à initialiser avant la boucle). Dans cette boucle, comme vous l'avez fait pour la page tshirt, gérez les événements onmouseover et onmouseout.



Faites un test pour contrôler que les images apparaissent en plus grand quand on les survole.

### Ajout d'un article

Le but est d'ajouter dans le calque de droite, l'article sélectionné en cliquant sur l'image du plus. Avant tout, dans la page boutique, div de droite, partie texte, insérez une balise <table> vide avec comme id "tSelection". En fait, ne laissez pas cette balise totalement vide car elle ne passera pas la validation du w3c : ajoutez une balise <tr> qui contient une balise<td> qui est vide. Au niveau de la feuille de style, donnez-lui les mêmes caractéristiques que le tableau des articles. On va remplir dynamiquement en JavaScript ce tableau pour obtenir une présentation alignée. Dans la boucle générale que vous venez d'écrire en JavaScript, écrivez le code de la fonction événementielle sur le clic d'un des plus. Dans cette fonction, vous allez construire une ligne de tableau pour l'insérer dans le tableau "tSelection" suivant ces étapes :

- dans une variable thisId, récupérez juste le numéro qui se trouve dans this.id (qui est concaténé à "ajout", donc utilisez les fonctions substr et length pour récupérer juste le numéro : essayez de trouver la syntaxe sur internet).
- Récupérez dans une variable article, l'élément dont l'id est la concaténation entre "img" et thisId (vous récupérez ainsi la balise image de l'article).
- Récupérez dans une variable prix, l'élément dont l'id est la concaténation entre "lbl" et thisId (vous récupérez ainsi le label).
- Créez une variable corps qui construit une balise "tbody" (indispensable pour créer une nouvelle ligne dans le tableau : on placera la balise "tr" dans cette balise).
- Donnez à cette balise l'id "b"+thisId.
- Créez une variable ligne qui construit une balise "tr".
- Il faut maintenant créer les 4 cases qui vont aller dans la ligne. Créez la variable case1 qui construit une balise "td". Pour le contenu de la case, on va faire les choses proprement : créez une variable txtCase1 qui construit un textNode de la façon suivante :

```
var txtCase1 = document.createTextNode(article.alt) ;
```

Lors de la création, on affecte au nœud de texte le contenu de l'attribut alt de l'image (donc le titre de l'image).

- Intégrez txtCase1 dans case1 (avec appendChild) et intégrez case1 dans ligne.

- Faites de même pour créer les 3 autres cases. La deuxième case doit récupérer le prix (qui se trouve dans l'attribut innerHTML de prix) : pensez aussi à concaténer le signe €. La troisième case doit contenir l'image de la corbeille : créez d'abord une variable `imgCorbeille` qui construit une balise "img", affectez lui "corb"+thisId comme id, le chemin vers l'image de la corbeille en src et "pointer" pour l'attribut cursor (pour avoir le doigt pointé).
- Intégrez la variable ligne dans corps. Enfin, intégrez corps dans la balise d'id "tSelection" (donc le tableau qui est dans la page).



Faites un test pour contrôler que sur le clic d'un plus, une ligne s'ajoute dans le calque de droite. Faites plusieurs ajouts.

### Suppression d'un article

Il ne reste plus qu'à gérer la suppression d'un article sélectionné. En JavaScript, juste après avoir géré la création de la ligne (donc à la suite du code précédent), c'est là qu'il faut insérer cet événement. L'événement est sur le clic d'une corbeille. Hors vous venez de créer un objet `imgCorbeille`. Donc écrivez le code de la fonction événementielle onclick sur cet objet. Dans cette fonction, il suffit de supprimer (avec `removeChild`) dans l'objet d'id "tSelection", la ligne sélectionnée. Mais comment récupérer la bonne ligne ? C'est celle dont l'id est la concaténation entre "b" et le numéro qui se trouve dans l'id `this.id` (qui contient "corb" suivi du numéro). A vous d'écrire le code correspondant.



Faites un test pour contrôler que les suppressions fonctionnent correctement (vous ne devez pas avoir de ligne vide dans le tableau).

### Interdiction d'ajouter plusieurs fois le même article

Puisque la vente se fait sur des articles en un seul exemplaire, on veut interdire de cliquer sur un plus si l'article est déjà dans le panier. Le plus simple est de rendre invisible l'image du plus lorsque l'article a été ajouté, et de le rendre visible à nouveau si l'article est retiré du panier. Essayez de gérer ces modifications tout seul.



Faites un test pour contrôler que tout fonctionne correctement à ce niveau là.

Enregistrez cette nouvelle version dans le dossier "version 8.4 images" sans oublier les dossiers js, XML, css, PHP.

## Synthèse

## Manipulation des images :

Pour manipuler des images, il faut activer le module exif en allant dans le fichier php.ini et en enlevant le ; qui commente la ligne extension=php\_exif.dll. Déplacez aussi la ligne extension=php\_mbstring.dll avant la ligne précédente.

## Parcours de dossiers :

Il est possible de parcourir un dossier et de récupérer les fichiers.

```
$flux = dir($unChemin) // permet de récupérer le flux d'accès au dossier
```

```
$fic = $flux->read() // permet de récupérer dans l'ordre les fichiers du flux
```

```
$flux->close() // ferme le flux
```

```
exif_imagetype($nomcomplet) // fonction booléenne pour savoir si le fichier est une image
```

## Calques dynamiques :

Il est possible, en utilisant les événements de la souris, de modifier la position ou la taille d'un calque (en modifiant les propriétés de style top, left, width, height).

## Fichiers d'images :

Il est possible de stocker des informations derrière les fichiers d'images et les récupérer avec des commandes PHP :

```
$infos = exif_read_data($ficComplet, 0, true) // récupération des informations
```

```
$infos["IFD0"]["Title"] // récupération de l'information titre
```

```
$infos["IFD0"]["Subject"] // récupération de l'information sujet
```

```
$infos["IFD0"]["Comments"] // récupération de l'information commentaire
```

## Séquence 9

# BASES DE DONNEES

Cette séquence présente comment construire et exploiter une base de données au format MySQL. Vous apprendrez à créer une base de données et à l'exploiter à partir du PHP.

### ► Capacités attendues en début de séquence

Savoir ce qu'est une base de données et avoir de bonnes notions algorithmiques et en PHP (abordé dans les séquences précédentes).

### ► Contenu

1. Pourquoi une base de données ? .....	126
2. Quel type de base de données pour un site ? .....	126
3. Création de la base de données .....	126
4. Utilisation de phpMyAdmin .....	127
5. Exploitation des données dans une page .....	128

## 1. Pourquoi une base de données ?

Pour le moment, le PHP a laissé entrevoir des possibilités d'optimisation de code et d'utilisation de fonctions bien pratiques. Il a permis aussi d'exploiter des informations contenues dans un fichier XML, ainsi que des fichiers d'images. Mais le plus important n'a pas encore été abordé. Le rôle principal d'un langage serveur est de pouvoir exploiter des données stockées côté serveur dans une base de données.

L'intérêt d'une base de données est qu'elle offre la possibilité d'utiliser le SQL pour accéder aux données enregistrées. De plus, elle offre la gestion automatique des verrous (si plusieurs personnes tentent de modifier les mêmes informations) ce qui n'est pas le cas pour les fichiers texte ou XML. La base de données permet aussi de gérer des utilisateurs et des droits (là encore, ce n'est pas le cas avec les autres méthodes).

En revanche, une base de données force l'utilisation d'un SGBDR (système de gestion de bases de données) installé côté serveur.

## 2. Quel type de base de données pour un site ?

Tous les types de bases de données peuvent être utilisés pour créer un site. Le PHP, et tous les autres langages côté serveur sont capables d'attaquer n'importe quel type de bases de données : SQL serveur, postgresQL, oracle, Access, MySQL...

Cependant, s'il n'existe pas déjà une base de données créée, il est conseillé d'utiliser MySQL avec le PHP : les 2 discutent très facilement ensemble. De plus, MySQL est gratuit et d'une puissance tout à fait correcte. Dernier argument de poids : quasiment tous les hébergeurs ont un serveur de données qui gère MySQL.

## 3. Création de la base de données

Il existe différentes techniques pour créer une base de données :

- créer la base directement dans le SGBDR avec les outils proposés,
- écrire un script SQL qui permet de générer automatiquement la base de données sous le SGBDR,
- créer le schéma conceptuel de données sous un logiciel spécifique (comme win'design) et utiliser ce logiciel pour générer automatiquement le script SQL qui permettra la création de la base de données.

La dernière méthode est la meilleure car elle permet de partir de la conception et tout le reste se génère automatiquement.

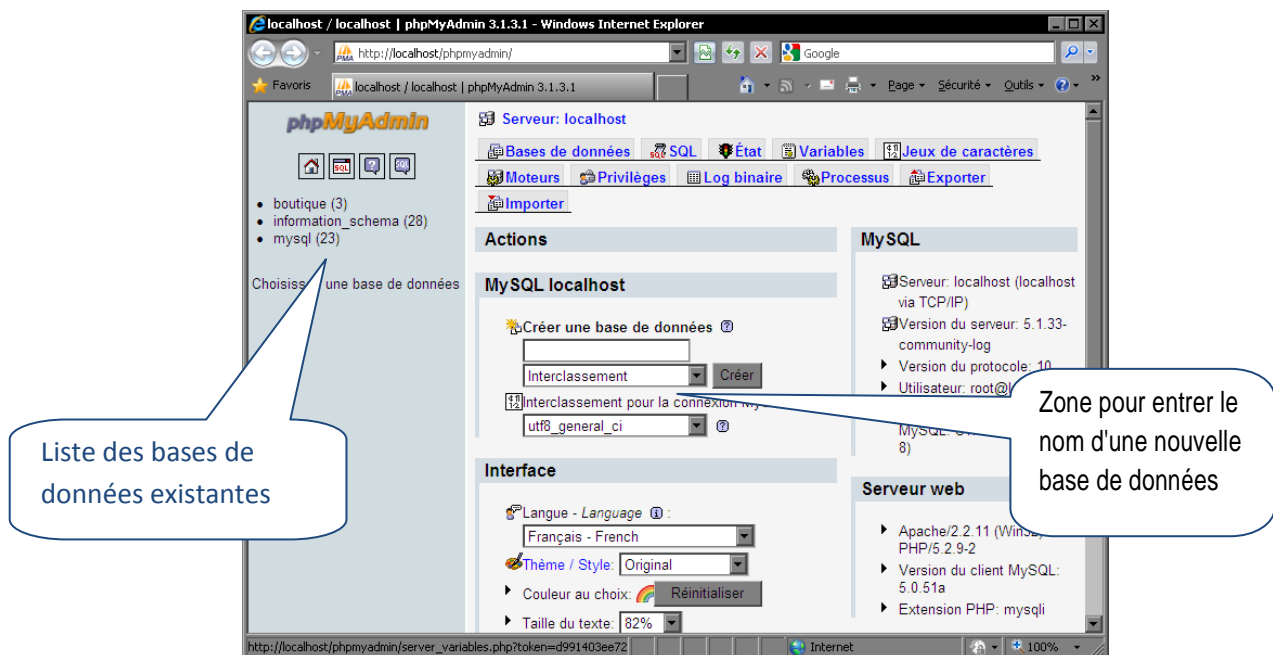
Cependant, pour découvrir phpMyAdmin, le logiciel qui permet de gérer les bases MySQL, vous allez créer la base de données en utilisant les outils intégrés.

#### 4. Utilisation de phpMyAdmin

phpMyAdmin est écrit en PHP et automatiquement installé avec WampServer. Il est aussi possible de l'installer de façon indépendante. Pour y accéder, voici la méthode : faites un clic gauche sur l'icône de WampServer et sélectionnez "phpMyAdmin".

##### Création d'une base de données

A l'ouverture de phpMyAdmin, l'écran ressemble à ceci :



Créez une nouvelle base de données en donnant le nom "boutique" et en cliquant sur "créer".

##### Création des tables

A l'écran suivant, il vous est proposé de créer les tables en précisant le nombre de champs. Vous vous doutez bien qu'à tout moment, vous pourrez ajouter ou supprimer un champ, voire modifier ses caractéristiques. Pour chaque table, vous aurez ensuite à préciser les champs (nom, type...). Il ne faudra pas oublier de préciser les clés primaires, et de sélectionner auto\_increment pour les clés primaires de numérotation automatique. Essayez de comprendre le fonctionnement de phpMyAdmin en créant les tables suivantes :

- client :

	Champ	Type	Interclassement	Attributs	Null	Défaut	Extra
<input type="checkbox"/>	numclient	int(5)			Non	aucune	auto_increment
<input type="checkbox"/>	nom	varchar(30)	latin1_general_ci		Non	aucune	
<input type="checkbox"/>	prenom	varchar(30)	latin1_general_ci		Oui	NULL	
<input type="checkbox"/>	adr1	varchar(50)	latin1_general_ci		Oui	NULL	
<input type="checkbox"/>	adr2	varchar(50)	latin1_general_ci		Oui	NULL	
<input type="checkbox"/>	cp	varchar(5)	latin1_general_ci		Non	aucune	
<input type="checkbox"/>	ville	varchar(30)	latin1_general_ci		Non	aucune	
<input type="checkbox"/>	infoslivraison	text	latin1_general_ci		Oui	NULL	
<input type="checkbox"/>	tel	varchar(10)	latin1_general_ci		Oui	NULL	
<input type="checkbox"/>	mail	varchar(50)	latin1_general_ci		Oui	NULL	
<input type="checkbox"/>	login	varchar(20)	latin1_general_ci		Non	aucune	
<input type="checkbox"/>	mdp	varchar(20)	latin1_general_ci		Non	aucune	

- commande :

	Champ	Type	Interclassement	Attributs	Null	Défaut	Extra
<input type="checkbox"/>	<u>numcommande</u>	int(5)			Non	aucune	auto_increment
<input type="checkbox"/>	datecommande	date			Non	aucune	
<input type="checkbox"/>	livraisonok	tinyint(1)			Non	aucune	
<input type="checkbox"/>	numclient	int(5)			Non	aucune	

- lignecom :

	Champ	Type	Interclassement	Attributs	Null	Défaut	Extra
<input type="checkbox"/>	<u>numcommande</u>	int(5)			Non	aucune	
<input type="checkbox"/>	<u>nomarticle</u>	varchar(50)	latin1_general_ci		Non	aucune	
<input type="checkbox"/>	qte	int(3)			Non	aucune	

### Création des enregistrements

Il est possible de commencer à remplir les tables pour qu'elles contiennent déjà quelques enregistrements, pour les tests. Mettez 2 clients en utilisant dans le menu, le lien "insérer". Si vous cliquez ensuite sur "afficher", vous devriez voir les informations de vos 2 clients.

### Exportation du script SQL

A tout moment, vous pouvez demander de générer le script SQL de votre base de données (structure des tables, mais aussi éventuellement le contenu à travers des ordres "insert").

Dans phpMyAdmin, pour exporter une base complète et non juste une table, sélectionnez d'abord la base de données "boutique". Dans le menu, cliquez sur le lien "exporter". Vous remarquez qu'il existe plein d'options possibles pour exporter la base de données. Gardez les options par défaut excepté en bas de page, cochez la case "transmettre" pour que le script soit copié dans un fichier, puis cliquez sur "Executer". Une fenêtre s'ouvre : cliquez sur "Enregistrer" et choisissez l'emplacement de sauvegarde (par exemple dans un dossier mysql que vous allez créer dans votre projet). Le fichier doit naturellement avoir été nommé "boutique.sql". Ouvrez le fichier (je vous conseille de l'ouvrir avec wordpad car juste le bloc note n'est pas adapté) pour voir le contenu. Vous trouverez la liste des ordres SQL pour créer et remplir la base de données. Vous avez compris que c'est un très bon moyen pour sauver une base de données.

## 5. Exploitation des données dans une page

Une fois la base de données créée sous MySQL, il est possible de l'exploiter avec le PHP. Vous allez voir les bases des manipulations, sans pour autant agir directement sur le site. Pourquoi ? Parce qu'il faut d'autres connaissances qui seront abordées à la séquence suivante pour correctement exploiter la base de données dans les différentes pages.

### Connexion à la base de données

Pour accéder à une base de données à partir d'une page, il faut utiliser des instructions dans un langage serveur car la base est obligatoirement du côté serveur. En PHP, vous pouvez par exemple vous connecter à la base de la façon suivante :

```
// connexion au serveur de données
mysql_connect($bdServeur, $bdUser, $bdMdp)
    or die("Erreur de connexion au serveur");

// connexion à la base de données
mysql_select_db($bdBase)
    or die("Erreur sur le nom de la base de donnée");
```

L'instruction "or die" n'est pas obligatoire mais permet de capturer l'erreur et d'afficher un message.

**\$bdServeur** : adresse du serveur (pour vos tests en local, ce sera "localhost")

**\$bdUser** : nom de l'utilisateur de la base de données (par défaut, si vous n'avez pas créé d'utilisateur, c'est "root")

**\$bdMdp** : mot de passe de l'utilisateur (par défaut, si vous n'avez pas créé ou modifié le mot de passe, c'est une chaîne vide "")

**\$bdBase** : nom de la base de données (car il peut y avoir plusieurs bases de données implantées sur un serveur de données).

On se doute que lors de la création d'une application web, plusieurs pages risquent d'avoir besoin de se connecter à la base de données. Plutôt que de réécrire plusieurs fois les mêmes choses, ce serait plus pratique de tout rassembler et d'intégrer ces informations au chargement de chaque page.

Dans le dossier PHP, créez le fichier curseurs.php. Dans ce fichier, mettez les balises PHP et écrivez la fonction Connexion() qui déclare en global les 4 variables précitées et qui contient les lignes d'instructions précitées, permettant la connexion au serveur et à la base de données. Il faut maintenant s'occuper de remplir les 4 variables. Dans le fichier init.php, dès le début du fichier, avant les includes, vous allez placer des variables globales. Initialisez les 4 variables précitées avec les bonnes valeurs. Pensez aussi à ajouter, en fin de fichier, l'include pour récupérer curseurs.php.

A partir de là, toutes les pages peuvent accéder aux informations de la base de données "boutique".

### Utilisation d'un curseur

Vous allez faire un petit test. Dans la page perso.php, intégrez un calque qui va juste servir de test d'affichage (aucune importance en ce qui concerne la position et la taille). Dans ce calque, mettez les balises PHP.



Voici les ordres PHP à connaître pour créer et parcourir un curseur d'accès à une base de données :

```
// connexion à la base de données
```

```
Connexion() ;
```

C'est votre fonction de connexion

```
// création du curseur
```

```
$curseur = mysql_query($requete) ;
```

```
// récupération d'une ligne du curseur
```

```
$ligne = mysql_fetch_assoc($curseur) ;
```

Cette fonction retourne null (assimilable à false) lorsque la fin de curseur est atteinte

```
// récupération d'un champ de la ligne
```

```
$contenuChamps = $ligne[$nomChamp] ;
```

```
// fermeture de l'accès à la base de données
```

```
mysql_close() ;
```

Puisque `mysql_fetch_assoc` retourne false en fin de curseur, si vous avez besoin de boucler sur un curseur, vous pouvez directement écrire une entête de boucle de cette façon :

```
while ($ligne = mysql_fetch_assoc($curseur)) {  
    ...  
}
```

Sachant tout cela, essayez d'écrire le code nécessaire pour afficher les noms et prénoms des personnes se trouvant dans la table client.



Faites un test pour contrôler que vous obtenez bien l'affichage des 2 clients que vous avez inséré dans la base de données.

Il existe plusieurs autres instructions pour manipuler une base de données. En voici quelques unes :

```
// exécution d'une requête d'administration (insert, update...)
```

```
mysql_query($requete)
```

```
        or die ("erreur dans la requete : ".$requete) ;

// accès à un champs précis d'un index (ligne) précis
$contentuChamps      =      mysql_result($curseur,      $index,
$contentuChamps) ;

// valeur de la clé du dernier élément inséré
$derniereCle = mysql_insert_id() ;

// nombre de lignes du curseur
$nbLignes = mysql_num_rows($curseur) ;
```

Il existe d'autres instructions que vous trouverez dans le manuel du PHP.

### Les autres méthodes de connexion

Il existe plusieurs méthodes pour se connecter à un serveur de données. Les différentes méthodes utilisent des fichiers de connexions qui vont posséder des performances variables. A l'heure actuelle, l'extension PDO (PHP Data Objects) qui ne se base plus sur le standard MySQL mais sur le standard SQLite, est la méthode considérée comme la plus performante pour l'accès aux données en PHP.

Si vous le désirez, allez faire un tour dans le manuel du PHP, pour voir les différentes fonctions qui commencent par "PDO". Cela vous donnera une idée.

Cependant, quelque soit la méthode utilisée, pour le développeur les grandes lignes sont toujours les mêmes. Juste quelques éléments de syntaxe varient.

Dans la suite de ce cours, nous utiliserons la méthode vue plus haut (standard MySQL) qui reste la plus connue et la plus utilisée. Le passage à une nouvelle méthode ne pose pas de problème particulier.

Sauvez votre travail dans le dossier "version 9 mysql", sans oublier le dossier mysql qui contient votre script, et le dossier PHP.

Après la sauvegarde, supprimez le calque et son contenu, que vous avez ajouté dans la page perso pour faire le test.

## Synthèse

phpMyAdmin :

Ecrit en PHP, phpMyAdmin est un logiciel qui offre une interface permettant de manipuler les bases de données au format MySQL.

Création d'une base de données MySQL :

- directement sous phpMyAdmin
- en important un script SQL sous phpMyAdmin

Connexion à la base de données à partir de PHP (exemple) :

```
mysql_connect($bdServeur, $bdUser, $bdMdp) or die ("erreur de connexion")
```

```
mysql_select_db($bdBase) or die ("erreur sur la nom de la base")
```

Gestion d'un curseur

```
$curseur = mysql_query($requete) // création d'un curseur
```

```
$ligne = mysql_fetch_assoc($curseur) // récupération d'une ligne du curseur
```

```
$ligne[$nomChamp] // récupération d'un champ
```

```
mysql_close() // fermeture de l'accès à la base de données
```

## Séquence 10

# TRANSFERTS ENTRE PAGES

Cette séquence présente les différentes possibilités de transferts d'informations entre les pages du site. La technologie Ajax est aussi abordée en détail.

### ► Capacités attendues en début de séquence

Avoir de bonnes notions algorithmiques, PHP, JavaScript et HTML (abordés dans les séquences précédentes).

### ► Contenu

1. Différences avec une application classique .....	134
2. Le formulaire .....	134
3. Les variables de sessions .....	137
4. AJAX .....	140
5. Transfert de variables simples .....	145
6. Les cookies .....	145

## 1. Différences avec une application classique

Dans une application classique, vous avez eu l'habitude de manipuler différentes données, que l'on peut classer de la façon suivante :

- variables locales : utilisables uniquement dans le bloc de code où elles ont été définies
- variables globales : utilisables partout dans l'application

Dans le cadre d'une application web, ce n'est pas si simple. Vous avez remarqué que vous avez placé et initialisé des variables dans `init.php`, et ces variables ont été décrites comme "globales". En réalité, elles ne le sont pas : `init.php` est intégré dans chaque page, d'où l'aspect "global" de ces variables, qui sont en fait redéclarées dans chaque page.

On peut donc parler de variables globales à une page, et non à plusieurs pages. Les variables déclarées dans une fonction seront, elles, locales à la fonction. Cet aspect a déjà été abordé dans la séquence sur le PHP.

Mais alors comment faire pour transférer des informations d'une page à une autre ? Cette séquence va présenter les différentes possibilités.

## 2. Le formulaire

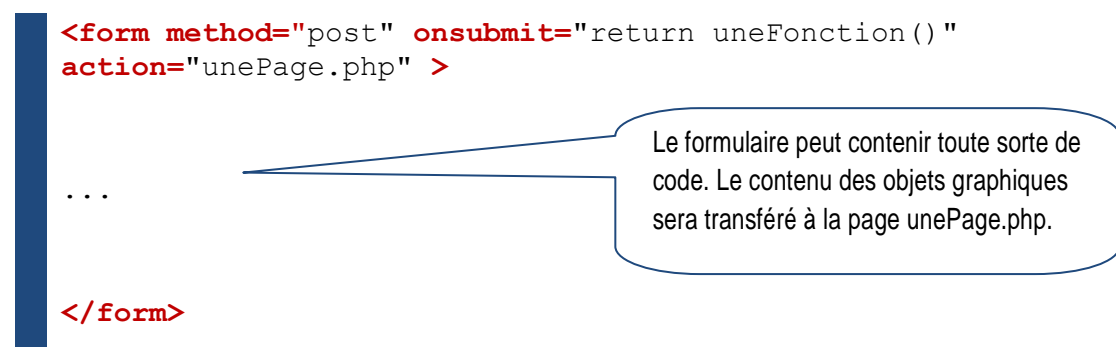
Une des méthodes les plus répandues pour transférer des informations d'une page à une autre est le formulaire. Nous verrons plus loin, en abordant Ajax, que son utilité devient discutable. Cependant, il faut savoir le lire et le manipuler car vous le trouverez souvent, et ponctuellement il peut être pratique à utiliser.

### A quoi sert un formulaire ?

Un formulaire est un regroupement d'objets graphiques (zones de texte, listes...) sur lesquels l'utilisateur peut interagir. Dans chaque formulaire se trouve obligatoirement un bouton de type "submit" qui permet d'envoyer les informations de l'utilisateur (ce qu'il a saisi ou sélectionné dans les objets graphiques) vers une autre page. Ainsi, la nouvelle page va pouvoir traiter les informations reçues.

### Caractéristiques d'un formulaire

Pour créer un formulaire, il faut insérer une balise html spéciale formulaire et lui donner les caractéristiques nécessaires. Voici à quoi ressemble cette balise :



Explication des attributs de la balise form :

- `method` : détermine la méthode de codage des informations transférées ("post" pour un transfert caché, ou "get" pour un transfert dans l'URL : "post" est classiquement utilisé avec les forms)
- `action` : précise la page qui va être appelée lors du clic sur le bouton submit et qui va recevoir les informations de la page.
- `onsubmit` : permet d'exécuter une fonction JavaScript juste avant le transfert des données. Si la fonction retourne false, alors les données ne sont pas transférées. Cet attribut est bien pratique pour faire des tests côté client afin de ne pas solliciter le serveur pour rien. Par exemple, vous pouvez contrôler côté client que les champs obligatoires ont bien été remplis. Comme cet attribut appelle une fonction JavaScript, vous vous doutez qu'il est possible de l'enlever et de gérer cet événement comme on l'a fait jusqu'à maintenant, directement dans le fichier js, après avoir donné un identifiant au formulaire.

La plupart des attributs sont optionnels (excepté `action`) et il existe d'autres attributs. Ceux présentés ici sont les plus classiques.

### Contenu d'un formulaire

Dans le formulaire, vous pouvez mettre tout ce que vous voulez. Tout s'affichera de façon classique. Tous les objets graphiques, contenus dans le formulaire et avec lesquels l'utilisateur peut interagir, verront leur contenu transféré vers la nouvelle page précisée dans l'attribut `"action"` du formulaire. Pour que la page de réception puisse récupérer les données, il faut donner un nom (attribut `"name"`) à chaque balise concernée par le transfert.

Le formulaire doit aussi obligatoirement contenir une balise submit qui est un bouton spécial, déclenchant le transfert des informations sur la page destination. Voici la syntaxe de cette balise :

```
<input type="submit" value="valider" />
```

L'attribut `"value"` est optionnel : en son absence, le message "submit" sera écrit dans le bouton.

Un autre bouton est spécifique aux formulaires, c'est le bouton de type `"reset"` qui permet de réinitialiser le formulaire. Ce bouton est optionnel. Voici sa syntaxe :

```
<input type="reset" value="réinitialiser" />
```

### Récupération des données transférées

La page appelée par le formulaire peut récupérer les données transférées. Suivant le mode de transfert, les données ne sont pas récupérées de la même façon.

Transfert en mode GET

```
$uneVariable = $_GET["lenom"] ;
```

C'est avec le nom donné à la balise du formulaire que l'on peut récupérer son contenu. Dans des anciennes versions de PHP, il suffisait de marquer `$lenom` pour récupérer l'information. Mais cette facilité d'écriture est à proscrire car il peut y avoir confusion avec une variable locale. A l'heure actuelle, cette écriture ne fonctionnera pas, sauf si vous apportez une modification dans le fichier de configuration du PHP,

ce qui est, à ce niveau là, fortement déconseillé car cette modification ne sera certainement pas prise en compte par le serveur qui hébergera votre site.

Transfert en mode POST

```
$uneVariable = $_POST["lenom"] ;
```

Même principe avec juste une petite variante de syntaxe.

Maintenant que vous avez les bases de connaissances nécessaires pour créer et exploiter un formulaire, il est temps de mettre en pratique sur le site.

Vous allez apprendre à gérer un premier formulaire sur la saisie du login et du mot de passe sur la page index. Dans cette page, entourez les 2 balises qui permettent de saisir le login et le mot de passe ainsi que le bouton, par une balise "form" qui enverra les données en mode post vers une page serveur.php (pour le moment pas de onsubmit). Changez le type du bouton pour en faire un "submit". Ajoutez l'attribut name aux balises du login et mdp en mettant comme valeur la même que pour l'id. En racine du projet, créez cette nouvelle page qui sera entièrement en PHP (donc insérez les balises concernées). Remplissez la page serveur.php en mettant juste un affichage des deux variables passées en post.



Faites un test : saisissez un login et un mot de passe puis cliquez sur ok.

Normalement une nouvelle page s'ouvre qui doit juste contenir l'affichage du login et du mot de passe. Refaites un test en revenant sur la page index et cette fois en cliquant sur ok sans remplir les zones du login et du mot de passe. Normalement vous obtenez votre ancien message d'erreur, mais juste après la page serveur s'affiche tout de même, avec cette fois aucune information puisque les 2 zones récupérées sont vides.

Le but est d'améliorer le code de sorte que la page serveur ne soit sollicitée que si les 2 zones ont été remplies. On pourrait utiliser le "onsubmit" dans la balise form. Mais il y a plus rapide compte tenu du code déjà écrit. Dans le fichier de script js, partie concernant la page index, il existe déjà une fonction événementielle sur le clic du bouton ok. Cette fonction contient déjà le test qui nous intéresse. Il suffit de rajouter "return false" à la fin du "alors" (après l'affichage du message d'erreur) et de rajouter "return true" à la fin du "sinon". Le bouton "submit" va donc appeler cette fonction qui va retourner false ou true, si elle retourne true le formulaire fera appel à serveur.php, sinon il y aura juste le message d'erreur que s'affichera.



Faites un test pour contrôler que, si les zones ne sont pas toutes remplies, le message d'erreur s'affiche puis rien ne se passe. En revanche, si les 2 zones sont remplies, alors la page serveur est appelée.

Maintenant que les données sont récupérées côté serveur, il est possible de faire un contrôle de leur validité dans la base de données. Dans la page serveur.php, supprimez le code actuel et, en utilisant un curseur pour accéder à la base de données, contrôlez que vous trouvez bien une ligne correspondant à ce login et ce mot de passe. Bien évidemment, faites le test directement dans la requête SQL. Si vous trouvez une ligne, alors la personne existe sinon le login et/ou le mot de passe ne sont pas corrects. Dans un premier temps, contentez vous juste d'afficher un message pour dire "correct" ou "incorrect". N'oubliez pas d'inclure init.php au début

du fichier serveur.php, sinon vous n'aurez pas accès aux variables de connexion et à la fonction Connexion()).



Faites un test : saisissez un login et mot de passe incorrect puis cliquez sur ok : la page serveur doit s'afficher avec le message "incorrect". Testez un login correct et un mot de passe incorrect pour voir si vous obtenez toujours le message "incorrect". Enfin, saisissez un login et mot de passe correct.

Tout fonctionne bien, mais bien sûr ce n'est pas cet affichage que l'on veut. Le but est de revenir à la page index et de gérer correctement l'affichage sur cette page (donc le passage au calque de bienvenue). Le problème est que de retour sur la page index, on ne se souviendra plus du fait que l'on a trouvé, dans la page serveur, le bon login et mot de passe. Il faudrait un moyen pour transférer une information d'une page à une autre, mais cette fois sans passer par un formulaire puisque les informations que l'on veut transférer ne sont pas saisies par l'utilisateur. Il existe en fait plusieurs méthodes que vous allez découvrir.

### 3. Les variables de sessions

Il est finalement possible de déclarer de vraies variables globales. Voici comment.

#### Principe de la variable de session

Une variable de session est une variable enregistrée côté serveur, liée à une session (donc à un utilisateur qui a accès au site) et qui disparaît lorsque la session se termine (lorsque l'utilisateur quitte le site).

Ce type de variable à plusieurs gros avantages :

- elle est stockée côté serveur, donc le contenu n'est pas accessible de façon frauduleuse côté client : du coup il est possible d'y stocker des informations sensibles, comme un numéro ou un login de client qui s'est identifié, pour le reconnaître sur toutes les pages.
- Elle est accessible sur toutes les pages : elle est donc réellement globale.
- Elle est spécifique à chaque session, donc malgré le fait qu'elle soit enregistrée côté serveur, il y a autant d'occurrences d'une variable de session que d'utilisateurs connectés. Chaque variable est spécifique à son utilisateur.
- Elle disparaît automatiquement en fin de session : inutile de s'occuper de libérer la mémoire.

A ces avantages s'opposent deux inconvénients qui sont liés :

- Comme c'est une variable côté serveur, elle ne peut être remplie quand la page est déjà côté client, donc il faut qu'elle soit gérée dès le début de la page, avant tout code html.
- Le serveur est obligatoirement sollicité pour gérer une variable de session, donc cela suppose de les gérer entre des changements de page ou le rechargement de la page. Cet inconvénient pourra être contourné plus loin, avec Ajax.

#### Comment créer une variable de session ?



Pour qu'une page puisse créer et/ou manipuler des variables de sessions, elle doit contenir avant tout l'ordre PHP suivant :

```
session_start() ;
```

Comme on risque d'avoir besoin des variables de sessions dans toutes les pages, autant intégrer cet ordre directement en début du fichier init.php. Faites-le.

Pour créer une variable de session, il faut utiliser la variable super-globale `$_SESSION` qui est un tableau associatif. Comme index de tableau, mettez le nom que vous voulez pour désigner votre variable. Il suffit ensuite de lui affecter la valeur désirée. Voici un exemple d'utilisation :

```
$_SESSION["login"] = $login ;
```

La variable `$_SESSION["login"]` est ainsi créée et reçoit le contenu de la variable locale `$login`.

Dans la page serveur.php, remplacez l'écho "correct" par la ligne de code qui permet de remplir la variable de session `$_SESSION["login"]` avec le login de la personne et `$_SESSION["id"]` avec l'identifiant de la personne. Pourquoi 2 variables de session pour la personne ? Juste pour que ce soit plus pratique : la première permettra d'avoir en permanence le login sans refaire une recherche dans la base de données (pour afficher le login dans le cadre de bienvenue, par exemple) et la seconde contiendra l'identifiant qui va permettre d'accéder à la personne dans la base de données. Enlevez l'autre affichage ("incorrect"). En fin de page serveur, écrivez le code qui permet d'être redirigé directement vers la page index.php. La redirection vers une page se fait de la manière suivante :

```
header("location: index.php") ;
```

### Comment utiliser une variable de session ?

Une variable de session s'utilise comme une variable classique, excepté qu'elle est accessible partout. Leur écriture étant un peu lourde, on les transfère souvent dans des variables locales.

Les 2 variables de sessions risquent de nous servir dans plusieurs pages. On va donc les récupérer directement dans init.php. Dans ce fichier, après avoir démarré la session, il faut contrôler si on a bien au moins une des deux variables de session. Faites un test sur l'existence de la variable de session "login" : utilisez pour cela la fonction `isset` qui reçoit en paramètre la variable à tester et qui retourne vrai si cette variable existe, faux sinon. Si la variable de session existe, alors stockez là dans la variable locale `$login` et stockez l'autre variable de session dans la variable locale `$id`, sinon transférez une chaîne vide dans les deux variables locales. Ces deux variables "locales" seront en réalité redéfinies à chaque page.

Le but est maintenant d'afficher le login dans le label correspondant. Directement dans la page index.php, remplissez directement le label "lblLogin" avec l'affichage de la variable `$login`.

Il reste à gérer l'affichage du bon calque (identification ou login) au chargement de la page. Soit le label "lblLogin" est rempli, ce qui signifie que la personne est reconnue, il faut alors afficher le calque de bienvenue, soit le label est vide c'est alors le calque d'identification qu'il faut afficher. Dans le fichier js, au début de la

gestion de la page index, faites un test sur le contenu du label "lblLogin" (testez la propriété "innerHTML"). Si le label est vide, alors rendez visible le calque d'identification et invisible le calque de bienvenue. Dans le cas contraire, inversez le code de visibilité. Juste avant ou après le test, pensez aussi à vider les 2 zones de texte contenues dans le calque d'identification.

Du coup, derrière l'événement clic sur le bouton ok, il n'est plus nécessaire de gérer les visibilités des calques : supprimez juste les 2 lignes correspondantes.



Faites un test en saisissant un mauvais login ou mot de passe : rien ne doit changer. Ensuite, saisissez un login et mdp correct : normalement le calque de bienvenue apparaît avec le bon login. Cliquez sur "deconnecter" : vous voyez à nouveau apparaître le calque d'identification. Mais ensuite, rechargez la page et vous remarquerez que le calque de bienvenue réapparaît...

Que c'est-il passé ? Au rechargement de la page, la variable de session étant toujours là, elle a rempli le label et, le label étant rempli, le calque de bienvenue est devenu prioritaire.

### **Est-il possible de manipuler une variable de session en JavaScript ?**

La réponse est tout simplement non. Le JavaScript est un langage client : il ne peut donc pas manipuler des variables qui sont du côté du serveur. Mais vous avez remarqué que l'on peut utiliser des astuces : il suffit d'utiliser la variable de session pour modifier le contenu d'une balise html pour qu'ensuite le JavaScript puisse y accéder. C'est ce qu'on a fait avec le label qui a reçu la variable de session du login.

En revanche, dans la situation actuelle, on aimerait pouvoir supprimer les variables de session pour "déconnecter" une personne. L'idéal serait de gérer cette suppression sur le clic du lien "deconnecter". Cet événement est géré dans le fichier js, donc en JavaScript. JavaScript ne pouvant pas manipuler une variable de session, cette solution ne va pas être possible.

### **Comment supprimer une variable de session ?**

Il faut donc que le clic sur le lien "deconnecter" fasse appel au serveur. Dans le lien de "deconnecter", remplacez le "#" du href par "serveur.php".

Donc la page serveur.php peut être appelée actuellement dans 2 cas : soit lors de l'identification, soit lors d'une déconnexion. Dans la page serveur.php, après l'include, faites un test pour savoir si la variable \$\_POST["txtLogin"] existe. Si c'est le cas, mettez tout le code que vous avez écrit (sauf la redirection vers la page index), sinon cela signifie que l'appel provient de la déconnexion. Il faut alors supprimer la variable de session, de la façon suivante :

```
session_unregister("login") ;
```

Faites de même pour la seconde variable de session.

Du coup, dans le fichier js, la fonction événementielle sur le clic du lien "deconnecter" n'a plus de raison d'être : mettez là en commentaire (elle va nous resservir plus loin).



Faites à nouveau plusieurs tests pour contrôler que les connexions et déconnexions se passent bien.

Tout fonctionne très bien mais un aspect est un peu frustrant : au début, un calque disparaissait pour laisser apparaître l'autre sans recharger la page. Depuis que l'on fait appel à la page serveur.php pour accéder à la base de données ou pour manipuler des variables de session, la page index.php est obligée de se recharger. Est-il possible d'éviter ce rechargement tout en faisant appel au serveur ? Oui : il est temps d'aborder Ajax.

Avant de continuer, faites une sauvegarde de tout votre travail dans "version 10.3 transferts".

## 4. AJAX

AJAX est l'acronyme de Asynchronous JavaScript And XML.

"Asynchronous", qui signifie "asynchrone" et qui laisse supposer qu'Ajax fonctionne toujours en mode asynchrone (les communications ne sont alors pas bloquantes) n'est pas tout à fait exact : on peut demander à la communication d'être synchrone donc de forcer l'attente de la réponse avant de continuer à exécuter le script. Ceci dit, cette possibilité est déconseillée et peut toujours être évitée.

"XML" qui laisse supposer que les échanges se font uniquement en XML, n'est pas tout à fait exact. En réalité, il est possible de récupérer un contenu au format XML ou au format texte. D'ailleurs dans notre site, on va utiliser exclusivement la récupération au format texte.

"JavaScript" qui donne l'impression qu'Ajax doit être écrit en JavaScript, est en revanche tout à fait exact. Ajax représente la mise en pratique de plusieurs technologies mais repose surtout sur l'utilisation d'un objet JavaScript (XMLHttpRequest) permettant la connexion au serveur en arrière plan.

### Créer les fonctions Ajax

Le but va être d'apprendre à créer des fonctions qui vont mettre en œuvre Ajax et qui seront réutilisables à volonté. Pour cela, on va créer 3 fonctions : une pour créer l'objet de connexion, une pour recevoir des informations du serveur (et éventuellement lui en envoyer juste avant) et une juste pour envoyer des informations au serveur sans attendre de retour. Pour écrire ces fonctions, placez-vous à la fin du fichier js et intégrez un important commentaire pour signaler que la suite va concerner Ajax.

#### Création de l'objet de connexion

Ecrivez la fonction Ajax() qui ne reçoit aucun paramètre. Dans cette fonction, initialisez la variable xhr avec null. Cette variable va représenter la connexion au serveur. A priori, elle doit recevoir un objet de connexion du type XMLHttpRequest, mais suivant les versions de navigateurs, cet objet n'est pas accessible et il faut en utiliser d'autres. Il se peut même que le navigateur ne soit tout simplement pas

capable de supporter Ajax (s'il est trop ancien). Il faut donc faire plusieurs tests pour remplir correctement la variable xhr :

Faites un premier test pour voir si window.XMLHttpRequest existe .Il suffit de faire un test en mettant juste l'objet dans les parenthèses, comme ceci :

```
if (window.XMLHttpRequest) {
```

Si le test est bon, alors faites l'affectation suivante dans xhr pour créer l'objet de connexion :

```
xhr = new XMLHttpRequest() ;
```

Dans le cas contraire, faites un nouveau test pour cette fois vérifier l'existence de window.ActiveXObject. Si l'objet existe, faites l'affectation suivante :

```
xhr = new ActiveXObject("Microsoft.XMLHTTP") ;
```

Dans le cas contraire (aucun des 2 objets n'existe), le navigateur n'est pas apte à gérer Ajax : affichez (avec alert) un message en conséquence.

Au final, après la fermeture de tous les tests, retournez xhr.

### Fonction d'envoi

Il est parfois nécessaire d'envoyer une information au serveur, sans pour autant attendre une réponse. Par exemple, lorsque la personne se déconnecte, il suffit d'avertir le serveur pour que celui-ci supprime les variables de session. Aucun retour n'est attendu.

Ecrivez la fonction AjaxEnvoi qui reçoit 2 paramètres : nomfic (qui contiendra la nom du fichier côté serveur qui devra être sollicité) et message (qui contiendra le message envoyé au serveur).

Dans la fonction, affectez à la variable xhr (on va garder le même nom même si ce n'est pas obligatoire) l'appel de la fonction Ajax(). On récupère ainsi l'objet de connexion.

Ensuite, faites un test pour contrôler que xhr n'est pas null (il suffit de tester s'il existe). Si xhr est correct, alors il faut ouvrir la connexion, de la façon suivante :

```
xhr.open("POST", nomfic, true) ;
```

- "POST" permet de préciser le mode de transfert des informations (ce sera "POST" ou "GET"). La méthode "POST" est classiquement utilisé pour un envoi simple au serveur.
- nomfic représente le paramètre qui contient le nom du fichier à solliciter sur le serveur.
- true représente le mode de transfert : synchrone (false) ou asynchrone (true). Un transfert synchrone bloque l'exécution du script tant que le serveur n'a pas répondu. Le transfert asynchrone permet de gérer l'envoi dans un thread indépendant, donc pendant que la connexion au serveur se fait, d'autres traitements peuvent continuer à s'exécuter, ce qui rend encore plus transparente la connexion au serveur. Il faut éviter d'utiliser le mode synchrone : on peut toujours l'éviter. Pour l'envoi, la question ne se pose même pas.

Ensuite, pour que le transfert en mode POST soit correctement interprété par le serveur, il faut réaliser un encodage, envoyé dans l'entête de la requête. Voici la ligne à écrire :

```
xhr.setRequestHeader("Content-type", "application/x-www-form-urlencoded") ;
```

Il ne reste plus qu'à envoyer le message reçu en paramètre, en utilisant l'objet xhr. Voici la ligne de code :

```
xhr.send(message) ;
```

La fonction d'envoi est terminée.

### Fonction de réception

C'est la fonction la plus complexe car elle permet éventuellement d'envoyer des informations au serveur, mais surtout elle attend une réponse de ce dernier.

Ecrivez la fonction AjaxReception qui reçoit 3 paramètres : nomfic (qui contiendra le nom du fichier sollicité côté serveur suivi éventuellement des paramètres transférés en mode get), typefic (qui contiendra "XML" ou "text" pour préciser le type des informations à récupérer du serveur) et enfin uneFonction (surprise : on verra plus loin à quoi sert ce paramètre).

Dans la fonction, comme vous l'avez fait pour AjaxEnvoi, remplissez la variable xhr avec l'appel de la fonction Ajax(), puis faites un test d'existence de xhr. Tout le reste du code va être mis dans ce test.

L'attente de la réponse du serveur va se faire par une fonction événementielle. Voici l'entête de la fonction :

```
xhr.onreadystatechange = function () {
```

Cette fonction se déclenche à chaque fois que le serveur réagit (pas uniquement quand il envoie des données, mais aussi quand il change d'état, par exemple quand il dit qu'il est prêt à recevoir des informations).

Dans la fonction, il faut donc contrôler si la fonction a été appelée parce que l'état est passé à "réponse envoyée" et si la page serveur a bien été trouvée. Voici le test à écrire :

```
if (xhr.readyState == 4 && xhr.status == 200) {
```

- readyState passe par les valeurs 1 à 4 : la valeur 4 signifie que la réponse est bien parvenue.
- status prend la valeur 200 si la page a bien été trouvée et s'il n'y a pas eu d'erreur (status peut prendre par exemple la valeur 404 en cas de page non trouvée).

Dans ce test, il faut réaliser un nouveau test, cette fois sur typefic car la réception ne se passe pas de la même façon si les informations sont envoyées au format XML ou au format texte. Donc contrôlez si typefic contient "XML". Si c'est le cas, les informations envoyées par le serveur sont dans xhr.responseXML. C'est là qu'une question se pose : comment exploiter ces informations reçues ? La plupart des sites (pour ne pas dire tous) exploitent ces informations en affichant ou en affectant

directement dans une balise du site. Mais cette solution n'est pas du tout satisfaisante car cela suppose qu'il faut réécrire cette fonction complètement à chaque fois que l'on veut recevoir des informations du serveur. On aurait pu se dire : il suffit de faire un "return" de l'information reçue. Mais le return n'est pas possible dans une fonction événementielle... Voici au final une solution qui va permettre de réutiliser à volonté la fonction `AjaxReception`. Rappelez-vous, vous avez donné en 3eme paramètre, `uneFonction`. C'est ici que va servir ce paramètre. Le but est de rediriger vers une fonction extérieure, l'information reçue du serveur. Il suffira alors, lors de l'appel de `AjaxReception`, d'envoyer en 3eme paramètre le nom d'une fonction qui recevra et traitera l'information du serveur, cette fonction pouvant être différente pour chaque appel. Du coup, `AjaxReception` devient totalement réutilisable sans modifier son contenu. Magnifique ☺

Voici donc la ligne à écrire :

```
uneFonction(xhr.responseXML) ;
```

Avec la même logique, écrivez le "else" pour la réception de type texte qui arrivera dans `xhr.responseText`.

Après la fonction événementielle, il faut ouvrir la connexion au serveur, comme vous l'avez fait précédemment dans `AjaxEnvoi`, mais cette fois avec la méthode GET. Comme c'est en GET, il n'y a pas de modification d'encodage. Puis, faites le send (comme dans `AjaxEnvoi`) mais en envoyant null en paramètre.

La fonction de réception est terminée.

### Utiliser Ajax

Le transfert du login et mot de passe ne vont plus se faire par le formulaire mais par Ajax. Dans la page `index.php`, commencez par enlever la balise `form`. Du coup, le bouton n'est plus de type "submit" mais de type "button" : faites la modification.

Dans le fichier js, sur l'événement correspondant au clic du bouton ok, remplacez le "return true" par l'appel de la fonction `AjaxReception` en mettant en premier paramètre une chaîne qui va faire appel au serveur en lui envoyant des paramètres en mode GET. Voici à quoi doit ressembler la chaîne :

```
"serveur.php?txtLogin="+lelogin+"&pwdMdp"+lemdp
```

où les variables `lelogin` et `lemdp` doivent contenir (ou sont à remplacer par) le login et le mot de passe récupérés dans les balises html correspondantes. Vous découvrirez par la même occasion, la syntaxe propre au mode GET : le nom de la page appelée doit être suivi d'un point d'interrogation puis des différents paramètres (nom=valeur) séparés par le signe &. Vous avez certainement remarqué ce genre de syntaxe dans certaines url.

Le second paramètre de la fonction doit être "text" puisque vous voulez recevoir du texte de la part du serveur. Le troisième paramètre doit être le nom d'une fonction qui va permettre de traiter l'information reçue. Mettez comme nom de fonction : `majLblLogin`.

Juste après la fonction événementielle sur le clic du bouton ok, écrivez la fonction `majLblLogin` qui reçoit un paramètre (appelez-le `unLogin`). Dans cette fonction, mettez à jour la balise `lblLogin` avec le paramètre.

Vous aviez mis en commentaire la fonction événementielle sur le clic du lien déconnecter : enlevez le commentaire et remplacez le contenu de la fonction par juste l'appel de la fonction AjaxEnvoi avec comme premier paramètre "serveur.php" et en second paramètre une chaîne vide. Après cet appel, transférez une chaîne vide dans la balise lblLogin.

Dans la page serveur.php, il faut réaliser quelques modifications :

- Les informations vont arriver en mode get, donc remplacez tous les POST par GET.
- Après avoir créé les 2 variables de session, il faut afficher (avec echo) le login pour qu'il soit récupéré avec Ajax (cela prouvera que la personne a été trouvée, puisque JavaScript ne peut pas tester les variables de sessions).
- Dans la même logique ajoutez un "else" au test de reconnaissance et mettez un echo d'une chaîne vide.
- Supprimez la redirection vers la page index (puisque la page serveur va être appelée par Ajax).

Dans la page index.php, lien déconnecter, enlevez l'appel à la page serveur.php pour remettre "#". La page serveur est appelée par Ajax.

Les modifications ne sont pas terminées : dans le fichier js, au début de la partie concernant la page index, vous aviez ajouté un test pour contrôler si le label de bienvenue était rempli. Ce test est toujours d'actualité lorsque la page se charge, mais n'est pas appelé lorsque Ajax a fait appel au serveur, donc sans recharger la page. Pour palier à ce problème, entourez le test d'une fonction du nom de majIdent(). Juste après la fonction, appelez là directement pour que le test soit appelé dès le chargement de la page (comme avant). Dans la fonction majLblLogin, juste après l'affectation dans la balise lblLogin, appelez la fonction majIdent(). Dans la fonction événementielle sur le clic du lien "deconnecter", après l'appel d'Ajax et l'affectation d'une chaîne vide dans la balise lblLogin, appelez la fonction majIdent().



Faites plusieurs tests : remarquez que lorsque vous cliquez sur ok, la page ne se recharge pas et pourtant la partie bienvenue apparaît. Remarquez aussi qu'il n'y a pas de blocage en attendant la réponse du serveur (les images défilent toujours avec la même fluidité). Faites plusieurs tests de connexion, déconnexion pour contrôler que tout fonctionne correctement.

Vous avez tout compris à Ajax. Vous vous doutez bien qu'à partir de maintenant, on va avoir du mal à s'en passer. Donc, plus de formulaire... et plus de rechargement de page pour modifier juste une petite partie de page.

Faites une sauvegarde sous "version 10.4 Ajax" avec tous les dossiers nécessaires.



## 5. Transfert de variables simples

Continuons dans l'exploration des différentes méthodes pour transférer les informations entre les pages.

### Transfert en mode GET

Le mode GET a été utilisé dans la partie Ajax pour envoyer des informations directement avec le nom de la page sollicitée. On a aussi vu que ce mode pouvait être utilisé dans un formulaire même si classiquement on utilise plutôt le POST.

Il est enfin possible d'utiliser ce mode dans d'autres circonstances. A chaque fois que vous appelez explicitement une page (à travers un lien, un clic sur un bouton...) vous pouvez coller au nom de la page, une succession de paramètres. Vous avez déjà vu la syntaxe dans la partie Ajax. C'est bien sûr la même syntaxe, qui ressemble à ceci :

```
nomPage.php?nom1=valeur1&nom2=valeur2&...&nomN=valeurN
```

Cette méthode de transfert peut s'avérer parfois bien pratique pour envoyer des variables qui ne sont pas des objets graphiques d'une page à une autre. Mais attention, aucune information sensible ne doit être transférée de cette façon car tout va apparaître dans l'url. Il n'y a qu'avec Ajax que les informations sont de toute façon cachées.

### Transfert en champ caché

Une autre méthode de transfert consiste à créer un champ caché dans un formulaire et de remplir ce champ de la valeur à transférer. La page côté serveur qui reçoit ces informations va pouvoir récupérer le contenu du champ caché. Voici la syntaxe d'un champ caché :

```
<input type="hidden" name="unNom" value="uneValeur" />
```

Cette solution n'a de raison d'être que dans un formulaire.

## 6. Les cookies

Un cookie est un petit fichier enregistré directement sur l'ordinateur du client et qui peut contenir toutes sortes d'informations.

### Principe du cookie

Le principal avantage du cookie est sa durée de vie. Contrairement à une variable de session qui ne vit que le temps d'une session, le cookie peut être permanent. Au moment de sa création, sa durée de vie est fixée. Du coup, les cookies sont souvent utilisés pour mémoriser des informations qu'il est pratique de retrouver lors d'une future connexion. Par exemple, quand vous allez sur un site et que vous êtes automatiquement reconnu, sans même avoir tapé votre pseudo et mot de passe, cela signifie qu'un cookie a été enregistré sur votre ordinateur lors d'une précédente visite et que le site, à la lecture du cookie, a pu vous identifier. Il n'y a qu'avec un cookie que l'on peut faire ce genre de chose. D'une manière générale, les cookies peuvent servir, à tout moment de la navigation, à enregistrer des informations qui peuvent être récupérées par n'importe quelle page du site.



Le principal inconvénient du cookie provient de sa localisation de sauvegarde : en étant enregistré sur le disque de l'utilisateur, ce dernier peut facilement y accéder, le supprimer ou même modifier son contenu. Le contenu d'un cookie est au format texte. Donc il ne faut pas enregistrer d'information sensible dans un cookie. Par exemple, si vous enregistrez l'identifiant de la personne et que l'utilisateur modifie le cookie en mettant un autre identifiant, il pourra alors peut-être aller sur le site en étant reconnu comme un autre utilisateur... Pour sécuriser les informations contenues dans un cookie, vous pouvez utiliser un logiciel de cryptage ou simplement réaliser un petit calcul personnel. Le but est de faire en sorte qu'une personne mal intentionnée ne puisse pas accéder à des données sensibles qui lui permettraient de faire des manipulations néfastes.

Un autre problème peut se poser : la suppression des cookies de la part de l'utilisateur ne doit pas nuire au fonctionnement du site. Car effectivement cette suppression peut intervenir à tout moment. Dans le cas de l'exemple ci-dessus, d'un cookie permettant de reconnaître automatiquement l'utilisateur lors d'une prochaine visite, la suppression du cookie n'a que peu d'incidence : l'utilisateur devra ressaisir son pseudo et mot de passe tout simplement.

Enfin, il est possible dans le navigateur de bloquer l'utilisation des cookies : ce point nous pousse encore plus vers une solution en minimisant les cookies. Nous allons tout de même voir comment ça marche, justement sur le principe de la reconnaissance automatique lors d'une visite ultérieure.

### Comment créer un cookie ?

La création d'un cookie se fait en PHP, par l'envoi d'une requête http au client puisque le cookie est enregistré côté client. Cela suppose donc qu'aucun affichage html ne doit précéder la création du cookie : il doit toujours se situer en entête de page ou dans une page totalement réservée au PHP (page qui n'affiche rien).

Voici la description de la fonction PHP qui permet de créer un cookie :

```
setcookie ($nom, $valeur, $delai, $chemin, $domaine, $securite)
```

Cette fonction retourne un booléen (pour signaler si la création s'est bien passée). Seul le premier paramètre est obligatoire. Voici la description des paramètres :

- \$nom : nom du cookie (donc de la variable pour y accéder)
- \$valeur : contenu du cookie (valeur affectée à la variable)
- \$delai : (optionnel) temps de vie du cookie. 0 ou aucune valeur pour une expiration en fin de session. time()+delaiEnSecondes pour un délai au-delà (time() retourne l'heure actuelle, il suffit d'ajouter une valeur en secondes).
- \$chemin : (optionnel) chemin sur le serveur indiquant le dossier d'accessibilité du cookie, pour savoir quel dossier peut manipuler le cookie. Par défaut, c'est le site complet qui a accès au cookie.
- \$domaine : (optionnel) domaine où le cookie est disponible.
- \$securite : le transfert du cookie ne peut se faire qu'avec un protocole sécurisé.

Vous allez essentiellement utiliser les 3 premiers paramètres.

### Comment utiliser un cookie ?

Le cookie créé peut alors être récupéré dans la variable super globale `$_COOKIE` qui est encore une fois un tableau associatif. L'index est le nom du cookie et la case accédée contient la valeur du cookie. Voici un exemple :

```
setcookie ("login", "Ed", time() + 31536000) ; // cookie
d'un an

...

echo $_COOKIE["login"] ; // affiche le login
```

### Comment supprimer un cookie ?

Il n'y a pas de fonction pour supprimer un cookie. La méthode détournée (mais non moins officielle) consiste à le recréer en mettant un délai négatif par rapport au temps actuel. Voici la syntaxe couramment utilisée :

```
setcookie ("login", "", time() - 3600) ;
```

Comme la création, la suppression est envoyée avec l'entête http, donc il ne peut y avoir d'affichage qui précède la suppression d'un cookie.

### Manipulation des cookies en JavaScript

Les cookies étant enregistrés côté client, il est normal qu'ils soient aussi manipulables en JavaScript. Voici comment créer un cookie en JavaScript :

```
// création d'un cookie

document.cookie = lenom + "=" + escape(lecontenu) ;

// création d'un cookie avec une durée de vie

var expireDate = new Date() ;

expireDate.setTime(expireDate.getTime() + duree) ;

document.cookie = nom + "=" + escape(contenu)
+ ";expires=" +
expireDate.toGMTString() ;
```

La fonction JavaScript `escape` permet d'encoder correctement la chaîne pour son enregistrement dans le cookie.

Pour récupérer un cookie, il faut, par rapport à son nom, extraire la sous chaîne du contenu. Voici une petite fonction bien pratique qui gère cette récupération :

```
// fonction de récupération d'un cookie

function lit_cook(nom) {
    var deb,fin ;

    deb = document.cookie.indexOf(nom + "=") ;
```

```

    if (deb >= 0) {
        deb += nom.length + 1 ;

        fin = document.cookie.indexOf(";", deb) ;

        if (fin < 0) {

            fin = document.cookie.length ;

        }
        return unescape(document.cookie.substring(deb, fin)) ;
    }
    return "" ;
}

```

Si vous analysez bien cette fonction, vous comprenez que tous les cookies sont stockés les uns à la suite des autres dans une même variable (`document.cookie`). Le but est de repérer le nom du cookie puis de positionner la variable `deb` sur le premier caractère de la valeur correspondant à ce nom. Il reste à repérer la fin qui se trouve soit au niveau du ";" (caractère de séparation des cookies), soit au niveau de la fin de la variable `document.cookie`. Au final, si le nom a été trouvé, la fonction retourne la sous chaîne qui se trouve entre les positions `deb` et `fin`.

La suppression d'un cookie se fait avec la même logique qu'en PHP : il suffit de le recréer en lui affectant une chaîne vide et en lui mettant un temps négatif.

Il faut tout de même éviter d'enregistrer un cookie côté client car tout ce qui est côté client est accessible et une éventuelle formule de calcul pourrait alors être décryptée.

### Mise en pratique

Avec ces connaissances, faites en sorte de mémoriser dans un cookie une information pour reconnaître la personne qui se connecte. Vous pouvez par exemple mémoriser son identifiant après lui avoir fait subir un calcul (du genre  $*353 - 27$ ). Donnez au cookie un nom non révélateur, du genre `login` (alors que c'est l'identifiant qui est mémorisé). Mettez une durée de 1 an. L'enregistrement du cookie se fera au moment de l'identification de la personne (si la personne est reconnue). Lorsque la personne arrive sur le site, contrôlez si le cookie existe et si c'est le cas, récupérez son contenu pour créer ensuite la variable de session correspondante. Pensez à supprimer le cookie quand la personne se déconnecte. Attention, à partir de l'identifiant récupéré dans le cookie, trouvez le `login` correspondant pour remplir la seconde variable de session.



Faites un test en vous identifiant puis fermez le navigateur pour que la variable de session soit supprimée. Relancez le navigateur et faites un nouveau test : vous devriez être reconnu.

Faites une sauvegarde sous "version 10.6 cookies", avec tous les dossiers nécessaires.

### C'est à vous

Le site est bien entamé mais pas terminé. Dans les explications guidées, nous n'irons pas plus loin car vous en êtes au point où vous avez les connaissances pour finir la programmation du site par vous même. Il est temps de vous laisser un peu d'autonomie pour que vous puissiez réellement tester vos acquis. Le but est de réaliser les travaux suivants :

- enregistrement temporaire des articles sélectionnés et des caractéristiques du t-shirt (vous pouvez enregistrer ces informations dans des variables de sessions ou des cookies)
- page du panier : qui récapitule tout ce qui a été commandé (donc en récupérer le contenu des variables de sessions) et qui permet de passer à l'étape de l'enregistrement de la personne ou directement du paiement. Vous ne pouvez pas gérer le paiement sécurisé car ce n'est possible qu'en passant par un site tiers de paiement, comme une banque. Cependant, vous pouvez enregistrer la commande dans la base de données.
- page des informations personnelles : où la personne va enregistrer ou modifier ses informations.

Vous trouverez dans la dernière version de la correction officielle, le site au complet pour vous donner une idée de ce qui peut être fait.

Cependant, avant de passer à la pratique lisez et testez la séquence suivante, sur les templates, qui va utiliser la page perso non terminée.

## Synthèse

Le formulaire :

Il permet de regrouper plusieurs objets graphiques dans le but d'un transfert d'informations entre pages.

La page qui est destinataire d'un formulaire, récupère les données soit dans `$_GET["nomVariable"]` pour un passage en mode GET, soit `$_POST["nomVariable"]` pour un passage en mode POST.

Les variables de sessions :

Ces variables sont accessibles à partir de toutes les pages, pendant la durée de la session.

`session_start()` // démarrage d'une session, indispensable pour manipuler des variables de sessions

`$_SESSION["nomVariable"] = valeur` // transfert d'une information dans une variable de session

`session_unregister("nomVariable")` // suppression d'une variable de session

AJAX :

Ensemble de technologies JavaScript qui permet de se connecter au serveur de façon transparente, sans avoir à recharger la page.

Transfert de variables par l'URL :

Le transfert en mode GET (par l'URL) se fait en construisant une chaîne au moment de l'appel d'une page, de la façon suivante :

`nomPage.php?nom1=valeur1&nom2=valeur2&...&nomN=valeurN`

Les cookies :

Le cookie est un fichier enregistré côté client, pouvant stocker des informations.

`setcookie ($nom, $valeur, $delai, $chemin, $domaine, $securite)` // création

`$_COOKIE["nomCookie"]` // variable de type cookie

## Séquence 11

# TEMPLATES ET FRAMEWORKS

Cette séquence aborde le fonctionnement des templates qui permettent d'isoler le code PHP dans une page séparée, comme cela a été fait pour le JavaScript. La théorie des frameworks est aussi rapidement présentée, sans être développée car elle fait en majorité appel à des notions de programmation qui ne seront abordées qu'en deuxième année.

### ► Capacités attendues en début de séquence

Avoir de bonnes notions algorithmiques, html et PHP (abordés dans les séquences précédentes).

### ► Contenu

1. Templates .....	152
2. Frameworks .....	155

# 1. Templates

## Principe

Vous avez appris à totalement séparer le code JavaScript de la page html. C'est aussi possible avec le PHP en utilisant un moteur de template. Le terme "template" signifie "gabarit" (ou modèle). Le but est de créer une page PHP qui va s'occuper de gérer les données (par exemple en les récupérant dans une base de données) et de créer une page html qui ne s'occupe que de l'affichage, sans se préoccuper de l'origine des données, donc sans intégrer de code PHP. C'est le moteur de template qui va s'occuper de faire le lien entre les 2 pages. La fusion se fait bien sûr côté serveur : une page correctement formatée et contenant les bonnes informations est alors envoyée au client.

Vous allez donc travailler avec :

- le fichier **template.php** qui vous est fourni et qui contient tous les outils nécessaires pour faire la fusion. Commencez par copier ce fichier dans le dossier template que vous allez créer en racine de votre projet.
- un **fichier php** qui gère tout ce qu'il veut au niveau php (voire même des includes d'autres fichiers qui contiennent des présentations, comme head.php et foot.php) et qui récupère toutes les données variables nécessaires pour l'affichage. Ce fichier s'occupe aussi de faire le lien avec template.php et demande l'affichage de la partie de page écrite exclusivement en html.
- Un **fichier à l'extension tpl** qui ne contient que du html et qui manipule des variables créées dans le fichier précédent.

## Premier exemple simple

Voyons le principe de construction à travers un premier exemple simple. Dans la page perso, vous aviez fait un petit essai pour afficher les nom et prénom des personnes enregistrées dans la base de données. A travers ce premier exemple, on va juste afficher le nom et prénom d'une seule personne dont on connaît le login.

### Fichier perso.php

Commencez par renommer le fichier perso.php en perso.tpl.

Créez un nouveau fichier persp.php et remplissez-le en suivant les indications :

- Ce fichier sera entièrement php, donc mettez les balises php en début et fin de fichier.
- Faites un include de head.php en début de fichier et de foot.php en fin de fichier.
- Entre les 2, gérez la connexion à la base de données et exécutez une requête qui va vous permettre de récupérer le nom et le prénom d'une seule personne, connaissant son login (donc en faisant une requête du genre "select nom, prenom from client where login ='Ed'").
- Récupérez dans la variable \$ligne le résultat (la ligne) de la requête. Pour faire simple, on va partir du principe que la requête marche (donc inutile de faire une boucle).

- Faites un include du fichier template.php (attention, il est dans le dossier template, donc pensez à donner le chemin). Vous auriez pu faire cet include dès le début du fichier, mais ici ça marche aussi.
- Créez une variable \$template de la façon suivante :

```
$template = new Template('./') ;
```

Le chemin mis entre parenthèses indique où trouver le fichier tpl, donc ici, c'est au même niveau.

- Attribuez un nom au fichier tpl qui sera utilisé par ce template : cela se fait en attribuant un tableau associatif pour les noms de fichiers. Voici la ligne de code correspondante :

```
$template->set_filename(array('perso' => 'perso.tpl')) ;
```

- Créez les variables templates qui seront utilisées dans le tpl, en leur assignant des variables php. Ici, on veut affecter le nom et prénom récupérés avec \$ligne :

```
$template->assign_vars(array(  
    'NOM' => $ligne['nom'],  
    'PRENOM' => $ligne['prenom']  
)) ;
```

- Il ne reste plus qu'à demander l'affichage du tpl :

```
$template->pparse('perso') ;
```

Le fichier est terminé.

### Fichier perso.tpl

Le fichier tpl contient pour le moment le code de l'ancien fichier perso.php. Il ne doit plus du tout contenir de code php (c'est le but !). Donc, supprimez les 2 include de head et foot. Il reste un bloc php dans le div divPerso. Supprimez tout le bloc et remplacez le juste par :

```
{NOM} : {PRENOM}
```

Vous reconnaissez les 2 variables que vous avez déclarées dans le fichier précédent, et qui ont reçu le nom et prénom provenant de la base de données. Donc, quand on veut utiliser dans un fichier tpl une variable créée à l'aide d'un template, il suffit de mettre son nom entre accolades.



Faites un test : vous devriez avoir l'affichage de la page perso qui doit s'afficher comme avant, avec juste un nom et prénom qui apparaît dans la zone de gauche.

Voilà les bases du principe. Cela ouvre pas mal de perspectives...

### Passage aux boucles



C'est bien de récupérer une ligne d'informations, mais c'est mieux d'en récupérer plusieurs. Là on ne voit pas bien comment faire pour boucler dans le tpl. C'est bien sûr possible.

Pour cela, il faut faire appel aux "blocs" dans le tpl. Un bloc est une portion de code html, comprise entre 2 lignes de commentaires spécifiques, et qui va pouvoir se répéter autant de fois que nécessaire. On a notre boucle !

### Fichier perso.php

Modifiez la requête en enlevant la partie where, afin de récupérer tout le monde.

Enlevez la ligne de code qui récupérerait dans \$ligne juste une ligne de résultat.

Supprimez la partie de code qui affectait des valeurs aux variables NOM et PRENOM dans un tableau associatif.

Il faut mémoriser les nom et prénoms. Mais il y en a plusieurs. Faites une boucle pour parcourir le curseur et récupérez chaque ligne dans la variable \$ligne. Dans la boucle, vous allez assigner non pas des variables classiques mais des variables de type bloc, de la façon suivante :

```
$template->assign_block_vars('client', array(
    'NOM' => $ligne['nom'],
    'PRENOM' => $ligne['prenom']
));
```

Le reste ne change pas.

### Fichier perso.tpl

C'est encore plus facile. Supprimez la ligne qui contenait les 2 variables et mettez à la place les lignes suivantes :

```
<!-- BEGIN client -->
{client.NOM}:{client.PRENOM} <br />
<!-- END client -->
```

Explication : Pour gérer un bloc qui va donc boucler sur plusieurs variables, vous devez placer ces 2 lignes de commentaires, avec BEGIN et END suivi du nom que vous avez donné au bloc dans le fichier php. Attention, il ne doit rien y avoir d'autre sur ces lignes de commentaires. Les variables qui sont manipulées dans le bloc doivent être préfixées du nom du bloc. La balise <br /> a juste été rajoutée pour aller à la ligne entre chaque client, et montre par la même occasion que l'on peut mettre du code html dans un bloc.



Faites un test : vous devriez voir cette fois tous les nom et prénom des clients s'afficher.

### Les blocs imbriqués

On peut aller plus loin en imbriquant des blocs, comme on imbrique des boucles. Voici ce qu'il faut savoir à ce niveau là :

- le nom du bloc imbriqué doit porter le nom du bloc parent en préfixe dans la déclaration du bloc (dans le fichier php)
- les variables de bloc, dans le fichier tpl, doivent avoir en préfixe la hiérarchie des blocs imbriqués (donc si la variable TOTAL se trouve dans le bloc commande qui lui-même se trouve dans le bloc client, la variable devra s'écrire client.commande.TOTAL)

En suivant ces règles, modifiez votre code pour faire en sorte d'afficher les nom et prénom de tout le monde, et d'afficher le mail uniquement des personnes qui en ont (faites en sorte que dans la table, certains aient un mail et d'autres non).



Faites un test : vous devriez voir tous les noms et prénoms et en plus les mails uniquement pour ceux qui en ont.

## 2. Frameworks

### Qu'est ce qu'un Framework ?

Un framework est un ensemble de bibliothèques, d'outils et de règles pour le développement dans un langage (ou groupe de langages).

Développer à l'aide d'un framework oblige une rigueur qui permet d'aboutir à une application très structurée et finalisée. Les nombreux outils offerts par les frameworks facilitent énormément le développement.

Côté inconvénients, il faut en noter trois qui sont assez importants :

D'abord, l'apprentissage d'un framework est rarement aisé. Il faut non seulement comprendre et assimiler l'ensemble des règles, mais aussi se familiariser avec les batteries d'outils qui sont généralement très nombreuses.

Ensuite, et ce n'est pas le moindre, l'utilisation d'un framework donne généralement naissance à des applications lourdes car les outils mis en œuvre sont souvent nettement plus complexes (et donc copieux en code) que l'utilisation que l'on en fait.

Enfin, et là c'est l'aspect pédagogique que je vois, utiliser un framework ne permet pas devoir concrètement le fonctionnement détaillé de certains outils car vous allez utiliser d'autres outils qui vont le faire à votre place.

### Pourquoi utiliser un framework ?

On peut se poser la question : pourquoi utiliser un framework ? Son utilité est évidente dans des entreprises où l'on recherche une rigueur et une homogénéité de

développement. Tous les développeurs utilisant le même framework vont devoir respecter les mêmes règles.

C'est encore plus vrai pour des gros projets où plusieurs développeurs interviennent en même temps. Dans le cadre d'un gros projet, le temps de production va être optimisé car de nombreux outils vont faciliter le développement. La lourdeur du code généré est alors nettement moins pénalisante.

En revanche, dans le cadre d'un petit développement personnel, ou par exemple pour réaliser un site de petite ampleur, l'utilisation d'un framework est alors discutable.

### **Quels sont les Frameworks pour le php ?**

Il en existe une cinquantaine... Vous pouvez aller faire un tour sur Internet pour en trouver la liste. Un nom qui revient souvent depuis quelques temps : Zend Framework.

### **Pourquoi ne pas aborder les frameworks dans ce cours ?**

Pour les étudiants frustrés qui auraient bien aimé toucher à un framework, sachez que la notion de framework est directement liée à la notion de programmation objet. Même si, malgré tous mes efforts, je n'ai pas pu éviter de manipuler certaines notions d'objet dans ce cours, j'ai toujours essayé de le faire de façon très simplifiée. Attaquer la notion de framework suppose de très bonnes connaissances en objet, que vous aurez l'an prochain.

### **Pourquoi ne pas créer son propre framework ?**

Ceci n'est pas une blague. Pour citer Rasmus Lerdorf, le créateur du PHP qui pense que seuls les créateurs d'un framework sont capables de l'utiliser correctement :

"Le mieux serait encore de bien identifier son besoin et de personnaliser un framework, voire de se créer son propre framework bien spécifique."

Rien ne vous empêche, au fur et à mesure de l'évolution de vos connaissances, de créer vos propres boîtes à outils que vous allez pouvoir réutiliser dans plusieurs projets. Quand vous attaquerez la programmation objet, cela vous paraîtra encore plus évident.

## Synthèse

Principe des templates :

Mettre le code php dans un fichier, le code html dans un autre et demander au fichier template.php de faire la fusion des 2.

Dans le fichier php :

Contient tout le code php nécessaire à la page.

```
$template = new Template('./') ; // crée la variable template
```

```
$template->set_filename(array('fic' => 'nomfic.tpl')) ; // renomme le tpl
```

```
$template->assign_vars(array('VAR1' => $variable1, ...)) ; // renomme les variables
```

```
$template->assign_block_vars('nomBloc', array(...)) ; // blocs de variables
```

```
$template->pparse('fic') ; // inclus le tpl à cet endroit
```

Dans le fichier tpl (au format html) :

```
{VAR1} // affiche une variable simple
```

```
<!--BEGIN nomBloc --> // début de bloc (boucle)
```

```
{nomBloc.VARIABLE} // affiche une variable de bloc
```

```
<!--END nomBloc --> // fin de bloc (boucle)
```

## Séquence 12

# BACK OFFICE

Cette séquence présente comment mettre à jour les données utilisées dans un site.

### ► Capacités attendues en début de séquence

Savoir créer un site.

### ► Contenu

1. Qu'est-ce qu'un back office ? .....	159
2. Fonctionnalités du back office .....	159
3. Sécurité du back office .....	160

## 1. Qu'est-ce qu'un back office ?

Lorsque vous créez un site, vous pensez en priorité aux internautes qui vont utiliser votre site. Mais dans le cadre d'un site dynamique, il faut aussi penser à celui ou ceux qui vont devoir d'occuper de la gestion du site.

Dans un site dynamique, voici les travaux qui risquent d'être nécessaires pour que le site côté internautes (le front office) fonctionne correctement :

- actualiser les informations variables qui s'affichent (par exemple, informations sur les articles)
- modérer les informations provenant des internautes (forum, livre d'or...)
- récupérer des bilans, statistiques et tout autre type de regroupement d'informations provenant des données récupérées
- valider les inscriptions des internautes
- etc...

Qui doit réaliser ces travaux ?

Là où les personnes désignées pour gérer le site. Ces personnes ne sont pas forcément des informaticiens. Ce qui signifie qu'il faut prévoir des moyens pour leur faciliter ces travaux de maintenance.

Comment faciliter cette gestion ?

La solution la plus couramment utilisée est la création d'un "back office". En opposition au "front office" qui représente le site accessible par les internautes, le "back office" est aussi un site contenant des pages uniquement accessibles par le ou les responsables du site.

## 2. Fonctionnalités du back office

### Aspect du back office

Un back office se construit de la même façon qu'un site classique, à quelques exceptions près :

- l'aspect visuel est généralement nettement moins travaillé, plus épuré
- la priorité est donnée à la facilité et rapidité de manipulation (même si ce point est aussi vrai pour un "front office", mais pas tout à fait dans le même esprit)
- il n'y a pas d'intégration de publicité

### Fonctionnalités

Le back office doit offrir les fonctionnalités suivantes :

- ajout/modification/suppression de toutes les données variables manipulées dans le site (base de données, les fichiers XML, les autres fichiers comme les images...)
- validation/modération d'informations provenant des internautes
- obtention d'informations provenant de bilans, statistiques...

- diverses recherches

En résumé, le back office, au-delà de la gestion des données, doit offrir toutes les fonctionnalités attendues par les responsables du site et leur permettant de leur faciliter la vie.

Cela suppose que, lors de la création d'un site, il faut poser toutes les questions nécessaires aux futurs responsables du site, pour savoir ce qu'ils attendent du back office.

### 3. Sécurité du back office

#### Comment accéder au back office ?

Il existe plusieurs méthodes pour mettre à disposition un back office :

- intégrer un lien dans le front office (déconseillé car les internautes ne comprennent pas la présence de ce lien, sauf si le lien n'apparaît qu'après identification et uniquement pour les personnes concernées)
- mettre le back office dans un dossier spécifique (très souvent nommé "admin") du site, ce qui fait qu'on y accède avec la même adresse, mais juste en ajoutant /admin.
- Mettre le back office à une adresse différente (cette solution est plus rare)

#### Comment sécuriser son accès ?

Le back office ne doit être accessible que par les personnes autorisées. Comment gérer cette limitation d'accès ? Voici différentes solutions :

- demander la saisie d'un login et mot de passe et diriger vers la page principale du back office que si les informations sont correctes, sans pour autant sécuriser la page : cette solution n'est pas du tout fiable car on peut contourner la saisie en tapant directement dans l'url l'adresse du back office
- utiliser une protection du dossier du back office avec le fichier .htaccess. C'est la solution la plus courante.

#### Comment fonctionne la protection avec .htaccess ?

Vous avez plusieurs étapes à suivre pour créer les 2 fichiers nécessaires à la protection du dossier et à son accès sécurisé.

##### Création du fichier .htaccess :

Dans le dossier du site, créez un dossier admin. Ce dossier va contenir les pages du back office. Pour sécuriser l'accès à ce dossier, on va utiliser un fichier htaccess. Ce type de fichier permet de paramétrer plein de choses, et entre autre de limiter l'accès à un dossier. Dans le dossier admin, créez un fichier que vous appellerez dans un premier temps "htaccess.txt". Dans ce fichier, vous allez intégrer les commandes suivantes :

```
AuthUserFile "C:/wamp/www/maBoutiquePerso/admin/.htpasswd"
AuthGroupFile /dev/null
```

```
AuthName "Acces Restreint"

AuthType Basic

<Limit GET POST>

    require valid-user

</Limit>
```

**Explications :**

La première ligne précise où se trouve le fichier qui va contenir le mot de passe pour l'accès sécurisé au dossier.

La seconde ligne précise le fichier contenant la liste des utilisateurs : dans notre cas il n'y en a pas.

La troisième ligne restreint l'accès au dossier, du coup une authentification va être demandée.

La quatrième ligne précise qu'il faut utiliser AuthUserFile pour l'authentification.

Enfin la balise précise pour quel(s) type(s) de méthode(s) la restriction s'application (GET et/ou POST).

**ATTENTION :** dans la première ligne, le chemin doit être le chemin absolu. En local, c'est le chemin complet. Quand le site est en ligne, le seul moyen d'avoir le chemin absolu est de placer sur la page d'entrée dans votre site un :

```
echo realpath ("admin/index.php") ;
```

(en supposant que dans le dossier admin, votre page principale s'appelle index.php)

La fonction PHP realpath donne le chemin réel de la page passée en paramètre. Vous verrez, vous obtenez parfois des chemins très exotiques, qui sont spécifiques à votre hébergeur.

Une fois le fichier htaccess.txt terminé, sauvez-le, fermez-le et renommez le en ".htaccess" (n'oubliez pas le "." devant).

**Fichier .htpasswd :**

Il va falloir un fichier .htpasswd qui contiendra le login et mot de passe de l'administrateur. Ce fichier est créé automatiquement car le mot de passe va être crypté. Voici la procédure pour créer ce fichier : ouvrez une fenêtre dos (d'invite de commande) et allez dans le dossier C:\wamp\bin\apache\apache2.2.11\bin (ou dossier équivalent suivant votre version d'apache). Dans ce dossier se trouve le fichier htpasswd.exe qui va permettre de générer le fichier qui nous intéresse. Tapez la commande :

```
htpasswd -c "c:/wamp/www/maBoutiquePerso/admin/.htpasswd"
Ed
```

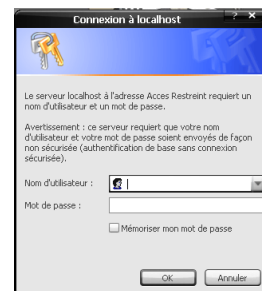
**ATTENTION :** remplacez Ed par le login que vous voulez donner à l'administrateur. Vérifiez aussi que le chemin est bien correct.



Validez cette commande : on va vous demander de saisir 2 fois le mot de passe. Ce mot de passe est alors crypté et l'utilisateur est créé dans le fichier .htpasswd. Par curiosité, allez voir le contenu du fichier .htpasswd. Vous allez y trouver votre login, suivi de ":" et d'une suite de caractères qui représentent votre mot de passe en crypté.

### Page d'admin :

Dans le dossier admin, créez la page index.php. Pour tester, mettez dedans l'affichage d'un message (c'est juste pour voir si ça marche). Testez en lançant le site par wamp, et en ajoutant à la fin de l'url "/admin" puis en validant. Vous devriez obtenir une fenêtre d'authentification qui demande un login et mot de passe. Faites la saisie demandée et cliquez sur ok : vous devriez alors obtenir la page index que vous venez de créer.



## Synthèse

### Back office :

Site d'accès limité (uniquement le ou les responsables du site) permettant de mettre à jour les données utilisées sur le site "front office" accessible par tous.

### Fonctionnalités :

Le back office doit permettre toutes les manipulations possibles sur les données (provenant de la base de données, des fichiers XML et toute autre forme de données).

### Sécurité :

Les pages du back office doivent être protégées. On utilise classiquement un fichier .htaccess pour ajouter un login et mot de passe pour accéder au dossier du back office.

## Séquence 13

# MISE EN LIGNE

Cette séquence présente les points indispensables qu'il faut connaître pour mettre en ligne un site.

### ► Capacités attendues en début de séquence

Savoir créer un site.

### ► Contenu

1. Intégration de publicité .....	164
2. Référencement .....	164
3. URL rewriting .....	164
4. Informations légales et CNIL .....	166
5. Choix du provider et nom de domaine .....	168
6. Mise en ligne et contrôle de la configuration .....	169
7. Tests sous différents navigateurs et norme W3C .....	169
8. Gestion et maintenance .....	171

## 1. Intégration de publicité

Vous avez la possibilité d'intégrer des publicités sur votre site. De nombreux sites proposent l'intégration de publicités.

Globalement, les publicités sont déconseillées car généralement les internautes n'apprécient pas trop. Si toute fois vous choisissez cette option, faites en sorte d'intégrer des publicités qui ne soient pas trop gênantes sur la page et qui soient en concordance avec les sujets que vous abordez dans votre site.

Un des programmes les plus classiques de publicité est celui proposé par Google : Google adsense. Vous devez par internet, sur le site [www.google.com/adsense](http://www.google.com/adsense), faire une demande d'intégration de publicité. Si le site est approuvé, vous recevrez le code à insérer où vous le désirez sur votre site. Les publicités intégrées sont en relation avec le contenu textuel de votre page qui est analysé en permanence.

## 2. Référencement

Si le but du site est qu'il attire le plus de monde possible, ce qui est par exemple le cas pour un site marchand, il faut penser à son référencement.

Comment faire en sorte que le site soit le mieux référencé possible dans les moteurs ? Il est possible d'agir à plusieurs niveaux :

- il faut correctement remplir les balises META du site
- l'ensemble des balises étant testé, le but est de limiter les textes explicites uniquement sous forme d'image (par exemple, si vous réalisez un joli titre sous Photoshop, dites vous que son contenu ne sera pas pris en compte, donc sans pour autant enlever votre titre, faites en sorte que l'information apparaisse aussi à un autre endroit, dans une balise, pour être prise en compte)
- les liens externes qui mènent vers votre site augmentent ses chances de référencement (si par exemple on parle de votre site dans des forums)
- il faut favoriser l'utilisation de l'URL rewriting

## 3. URL rewriting

*(merci à Yannick qui est à l'origine de la rédaction de cette partie)*

La technique de l'URL rewriting permet de changer les noms des pages PHP en nom de pages html. Cette transformation a le triple avantage de sécuriser les pages (en masquant les éventuelles variables passées en paramètre), de faciliter la mémorisation de noms (simplifiés par rapport aux noms d'origine) et de faciliter le référencement (les robots d'indexation de la plus part des moteurs de recherche dont Google ne lisent pas les URL qui ont plus de deux paramètres, de plus la découpe des mots clés d'une url favorise le référencement).

Comment ca marche ?

Dans un premier temps, des modules d'apache doivent être activés. Pour le cas de WampServer, faites un clic gauche sur l'icône de Wamp, allez dans le menu Apache / Modules Apache, puis cochez l'option `rewrite_module`. Le serveur devrait alors se relancer tout seul. Si ca ne marche pas, ou pour les autres serveurs, il faut modifier le fichier `httpd.conf` pour dé-commenter les lignes suivantes :

```
LoadModule rewrite_module libexec/mod_rewrite.so
```

```
AddModule mod_rewrite.c
```

Si ces lignes ne sont pas présentes, ajoutez-les puis sauvegardez et redémarrez le serveur. Une fois ces actions effectuées, on peut attaquer la réécriture.

Tout se passe dans le fichier `.htaccess` qui se trouve à la racine du site. Il faut le créer s'il n'existe pas. Commençons notre premier test. Créez une page `test.php` ou l'on met de ce qu'il y a de plus basic :

```
<html><head><title>Ceci est un  
test</title></head><body>Bonjour ! Ceci est un test  
!</body></html>
```

Puis dans le fichier `.htaccess` (n'oubliez pas le point devant le nom), écrivez ces lignes :

```
Options +FollowSymlinks  
  
RewriteEngine on  
  
RewriteBase /
```

Explications : Les deux premières lignes sont nécessaires pour activer et dire au serveur que l'on souhaite utiliser l'url rewriting pour notre site. La troisième ligne sert à définir le répertoire de base de notre site. La plus part du temps c'est `/` mais dans le cas de WampServer la racine du site est `http://localhost/` et non `http://localhost/monSite/`. Il faut donc la modifier avec ce paramètre : `/monSite/`. Par la suite, les règles de réécriture peuvent être mises en place.

Pour notre exemple, voici la ligne qui correspond :

```
RewriteRule ^test.html$ test.php [L]
```

Petite explication : `RewriteRule` indique que l'on souhaite commencer l'écriture d'une URL. Le `^` indique que l'on prend toutes les url de notre site avant le nom de la page puis nous retrouvons la page que l'on souhaite appeler à la place de `test.php`.

Enregistrez le fichier puis rendez vous sur l'url suivante :

```
http://localhost/monSite/test.html
```

Si le navigateur affiche une erreur 404 c'est que l'url rewriting n'est pas bien mise en place sinon félicitation ca fonctionne ! Vous pouvez maintenant modifier le `test.html` en `ceQueVousVoulez.html` elle appellera toujours la page `test.php`.

Vous savez utiliser de façon basique l'url rewriting. Allons plus loin en ajoutant des règles d'écriture. Les règles d'écriture permettent d'utiliser l'URL Rewriting de façon dynamique. Un site internet est susceptible d'avoir des adresses du type :

article.php?id=34&titre=Coucou

Si on écrit manuellement les règles de re-nommage comme précédemment on risque de ne plus s'en sortir. Pour cela on peut utiliser ce genre d'écriture :

```
RewriteRule ^article-([0-9]+)-(.*)\.html$ article.php?id=$1&titre=$2 [L]
```

Petite explication : la première partie, comme on la déjà vu, correspond à la page que l'on va demander. Ici, article est donc notre page fictive. Ce qui suit l'article est une valeur qui ne peut être qu'un chiffre ou plusieurs chiffres et rien d'autre. Ensuite le (.\*?) permet d'indiquer que l'on peut avoir différents caractères que ce soit des chiffres, des lettres ou des caractères spéciaux. Il est vivement conseillé d'utiliser les - au lieu des \_ car les moteurs de recherche découpent les différents mots clés avec le caractère -. Pour récupérer les valeurs, l'utilisation des variables \$1 et \$2 sont nécessaires. \$1 prend la valeur du premier paramètre qui est ici le ([0-9]+) et \$2 prend la valeur du deuxième paramètre qui est ici le (.\*). Si une troisième valeur était présente on aurait mis \$3 ainsi de suite... L'ordre n'est pas au hasard, c'est en correspondance c'est-à-dire que le premier paramètre dans l'url en .html correspond à la variable \$1. Ainsi on peut écrire les unes à la suite des autres les différentes règles d'écriture.

Voici les différentes restrictions des URL que l'on peut trouver :

(.\*) : Tous caractères que cela soit des chiffres, lettres caractère spéciaux...

([0-9]+) : Que des chiffres (le + permet de spécifier que l'on peut trouver plusieurs chiffres)

([a-z]+) : Que des lettres (Le + permet de spécifier que l'on peut avoir une ou plusieurs lettres)

([a-z0-9]+) : Un ou plusieurs chiffres/lettres (Le + permet d'indiquer que l'on peut avoir plusieurs chiffres/lettres)

Voici une explication succincte de l'utilisation des URL rewriting.

Une petite règle qui permet de renommer toutes les pages en .html en .PHP :

```
RewriteRule ^(.*)\.html$ $1.php [L]
```

Vous avez compris que ces modifications peuvent avoir de lourdes retombées sur le code. Il est donc à priori conseillé de penser à cette solution dès le début, pour éviter d'importantes modifications.

## 4. Informations légales et CNIL

### Mentions légales

Le droit dicte un certain nombre d'informations légales qui doivent figurer sur un site. Les règles sont bien sûr plus strictes pour un site commercial (plus largement, les sites industriels, commerciaux ou de service) que pour un site personnel.

Sites personnels

- informations sur l'hébergeur : nom, adresse, téléphone

#### Sites commerciaux

- informations sur l'éditeur du site : nom ou raison sociale, adresse, téléphone, numéro d'inscription au répertoire des métiers ou registre du commerce, capital (pour les sociétés)
- informations sur le responsable du contenu du site : nom du directeur ou co-directeur de publication
- informations sur l'hébergeur : nom, adresse, téléphone

#### Données personnelles

Un site, personnel ou commercial, manipule parfois des données personnelles relatives aux internautes (à travers, par exemple, une fiche d'inscription).

Si le site manipule ce genre de données, il est obligatoire de faire une déclaration à la CNIL (Commission Nationale de l'Informatique et des Libertés) et de faire mentionner sur le site un certain nombre d'informations :

- le numéro de déclaration fourni par la CNIL
- le droit d'accès des internautes à leurs données collectées, ainsi que leur droit de modification et suppression
- dans le cas d'utilisation de cookies, une mention précisant que ces cookies ne sont pas destinés à collecter des informations personnelles
- une mention précisant que les données collectées ne seront pas revendues ni communiquées à des tiers

#### Propriété intellectuelle

Il n'est pas obligatoire de mentionner, sur le site, les droits des internautes en ce qui concerne la réutilisation partielle ou totale du contenu du site (texte, images...). En l'absence d'informations, la loi de la propriété intellectuelle s'applique par défaut :

"Toute représentation intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droits ou ayants causes est illicite. Il en va de même pour la traduction, l'adaptation ou la transformation, l'arrangement ou la reproduction par un art ou un procédé quelconque."

Bien sûr ceci s'applique dans l'autre sens : quand vous créez un site, faites attention à son contenu, en particulier lorsque vous récupérez des textes ou images sur d'autres sites ou d'autres supports. Vous devez contrôler que ce qui est récupéré est libre de droit, ou demander une autorisation au près des auteurs.

#### Liens hypertextes

L'intégration de liens hypertextes pointant vers d'autres sites est possible, mais sous certaines conditions :

- il faut demander l'autorisation au près du webmaster du site destination
- notre responsabilité est engagée en ce qui concerne le contenu du site destination (si par exemple le contenu est illicite)

#### Spécificités des sites marchands

Certaines règles viennent s'ajouter au niveau des sites marchands :

- si des photos sont utilisées pour présenter les articles, une mention doit préciser si elles sont non-contractuelles et qu'elles n'engagent pas le vendeur
- les prix doivent être clairement affichés, avec obligatoirement la mention HT ou TTC
- la disponibilité du produit doit être précisée (et les délais d'obtention)
- les frais et délais de livraison
- la durée de validité du prix et de l'offre
- les conditions générales de vente (informations sur l'offrant – identité, adresse, téléphone -, informations sur la prestation – étapes de réalisation du contrat -, informations relatives à la contre-prestation)

## 5. Choix du provider et nom de domaine

Pour que le site soit accessible sur internet, il doit être hébergé chez un FAI (Fournisseur d'Accès à Internet, ou Provider). Il existe plusieurs hébergeurs, français ou même étrangers. Vous trouverez sur internet tous les hébergeurs possibles, et aussi des sites qui permettent de comparer les différents hébergeurs.

Lors du choix d'un hébergeur, vous devez prendre en compte un certain nombre de critères :

- le langage serveur et la base de données acceptés (le provider doit être capable d'héberger votre site, donc doit posséder l'interpréteur de code serveur nécessaire et aussi le sgbdr correspondant). Si vous avez utilisé le couple PHP/MySQL, vous avez accès à la majorité des providers.
- la place réservée au site (pages mais aussi documents, photos, vidéos...)
- la place réservée à l'éventuelle base de données (en prévoyant son expansion)
- la bande passante (suivant la fréquentation prévue et la quantité de téléchargement)
- éventuellement le nombre de connexions simultanées
- les tarifs

Si vous désirez que l'adresse de votre site soit spécifique, vous devez aussi prendre (acheter) un nom de domaine. De nombreux sites proposent la possibilité d'acquérir un nom de domaine (par exemple, le site officiel de l'AFNIC pour les noms de domaine .fr).

Un nom de domaine doit être bien sûr unique, donc un contrôle sera fait au préalable avant de vous permettre d'acquérir le nom demandé. Un nom de domaine est payant, et il est affecté pour une durée limitée.

## 6. Mise en ligne et contrôle de la configuration

Pour mettre en ligne le site, il faut utiliser un logiciel FTP, ou les fonctions ftp éventuellement proposées sur le site du provider, pour monter les pages.

Une fois le site en ligne, vous pouvez avoir des surprises, parfois très importantes et préjudiciables, sur l'exécution des pages. Comment peut-il y avoir des différences entre votre exécution en locale et celle en ligne ? C'est une histoire de différence de version. Par exemple, vous avez peut-être utilisé php5 et en ligne, le provider utilise l'interpréteur de php4. Evidemment ça change tout. Cependant, la plupart des providers offrent la possibilité de modifier certaines configuration, notamment avec le fichier .htaccess. Pour reprendre l'exemple assez courant qui vient d'être cité, pour forcer le provider à utiliser la version 5 du PHP (si toute fois cette version est installée), vous devez mettre en racine de votre site, un fichier portant le nom ".htaccess" et contenant la ligne :

```
SetEnv PHP_VER 5
```

Si votre site respecte les normes W3C, les seuls problèmes que vous pouvez rencontrer seront au niveau du langage serveur. Par exemple, si vous avez utilisé les balises raccourcies <? et ?> au lieu de <?PHP et ?> et que vous avez configuré en local le PHP pour qu'il accepte ces balises raccourcies, ne soyez pas surpris qu'une fois en ligne, le site ne fonctionne plus du tout.

Donc la règle à respecter et de ne pas changer en local la configuration par défaut de PHP afin d'avoir une compatibilité maximale avec les versions en ligne.

En ce qui concerne les modules d'extension (comme celui qu'on a utilisé avec exif pour gérer les images), c'est différent : les hébergeurs ont généralement une configuration assez large à ce niveau là.

Conclusion, il faut toujours tester le site une fois en ligne.

## 7. Tests sous différents navigateurs et norme W3C

### Les navigateurs

Lorsque vous créez un site, il faut prendre en compte le fait que, suivant les internautes, le site va tourner sur des navigateurs différents. Il se trouve que, malheureusement, il n'y a pas une compatibilité totale entre les sites. Cela signifie qu'il faut coder en faisant attention au résultat non pas sur un navigateur mais plusieurs.

Est-il possible de tester un site sur tous les navigateurs qui existent ? La réponse est à priori non car, contrairement à ce que l'on pourrait penser, il existe de nombreux navigateurs : plus d'une cinquantaine. Cependant, leur pourcentage d'utilisation n'est pas équivalent. Une étude datant d'octobre 2009 donne les pourcentages suivants :

Internet Explorer (65%), Firefox (24%), Safari (4%), Chrome (3,5%), Opera (2%), Netscape (0,5%), Autres (moins de 1%).



Cela suppose que tester le site sur Internet Explorer, Firefox, Safari, Chrome et Opera représente un panel extrêmement large des internautes. En tout cas, il faut au strict minimum tester sous Internet Explorer et Firefox.

Attention, ici seuls les navigateurs ont été cités. Cependant, pour chaque navigateur, il existe plusieurs versions qui souvent ont des différences notoires. Vous n'obtiendrez pas forcément le même résultat sous la version 7 d'Internet Explorer que sous la version 6.

### **La norme W3C**

La norme W3C a été présentée dans la séquence 1. Elle permet justement de palier à ce problème d'incompatibilité entre les navigateurs. Plus concrètement, comment est-ce possible ? La norme W3C dicte un ensemble de règles précisant d'écriture du code html de sorte que ce code soit qui suit la norme soit compatible sur tous les navigateurs. Cela permet aussi de faire en sorte que l'ensemble des développeurs suivent les mêmes règles.

Où trouver l'ensemble des règles de la norme ? Vous pouvez aller directement sur le site officiel du W3C, cependant pour avoir une information plus claire et résumée, il y a d'autres sites comme celui-ci : [http://openweb.eu.org/articles/html\\_au\\_xhtml](http://openweb.eu.org/articles/html_au_xhtml)

Est-il possible de faire contrôler son site pour voir s'il respecte les règles W3C ? oui. Pour cela, il faut que le site soit en ligne, donc placé chez un hébergeur. Puis, il suffit de donner le lien sur le site suivant pour que votre site soit testé (attention, le lien ne représente qu'une seule page, donc il faut à chaque fois donner le lien de chaque page pour qu'elles soient toutes testées) : <http://validator.w3.org/>

Si vous obtenez une liste impressionnante d'erreurs, ne vous inquiétez pas trop : comme dans un programme classique, une erreur provoque parfois en cascade de nombreuses autres erreurs. Le but bien sûr au final est d'éliminer toutes les erreurs.

### **Les tests**

Vous l'avez compris, pour tester votre site, en dehors des tests classiques des différentes fonctionnalités, vous devez :

- faire tester votre site par le validateur W3C et éliminer toutes les erreurs
- tester votre site sur plusieurs navigateurs, au moins les plus connus, pour contrôler que l'affichage correspond bien à ce que vous attendez (même validé, j'ai remarqué qu'il y a parfois des petites différences, généralement anodines mais on ne sait jamais...)
- réaliser les tests aussi en ligne

## **8. Gestion et maintenance**

Enfin, il faut prévoir une gestion du site et sa maintenance.

### **Gestion du site**

Si le site est statique, il se suffit à lui-même et ne nécessite à priori pas de gestion. En réalité, il faut tout de même vérifier de temps en temps si son contenu n'est pas obsolète.

Si le site est dynamique, alors il est nécessaire de mettre à jour les données (généralement par la partie back-office), voire de modérer le site, s'il contient des informations déposées par les internautes (dans un forum, un livre d'or...).

Une personne, ou plusieurs, doivent donc être désignées et posséder les droits nécessaires pour intervenir dans ce sens.

### **Maintenance**

Au-delà de la gestion du site, se pose le problème de la maintenance. Le site peut comporter des erreurs qui vont éventuellement se révéler en cours d'utilisation. Il peut aussi faire l'objet d'attaques malveillantes. Autre problème courant : la base de données peut rencontrer des problèmes qui se répercutent sur le site.

Pour régler tous ces problèmes, il faut des compétences techniques. Une personne ayant les compétences doit être aussi désignée pour gérer la maintenance.

## Synthèse

### Référencement :

Pour améliorer le référencement :

- remplir les balises META
- mettre des informations pertinentes dans les autres balises
- augmenter les liens externes qui pointent sur le site (forums...)
- favoriser l'utilisation de l'URL rewriting

### URL rewriting :

Informations dans le fichier .htaccess permettant de renommer les url du site en url plus simples en favorisant les extensions html. Ce renommage permet de protéger les données transférées en mode GET et aussi d'améliorer le référencement.

### Informations légales :

Attention, mettre un site en ligne suppose de respecter un certain nombre de règles dictées par les lois. Ces règles sont différentes suivant le type de site (personnel, commercial...).

### Choix du provider et nom de domaine :

Le choix du provider (celui qui va héberger le site) va se faire suivant certains critères : place réservée au site et à la base de données, bande passante, nombre de connexions simultanées, tarifs...

Il est possible aussi d'acheter un nom de domaine pour avoir une adresse personnalisée.

### Mise en ligne et contrôle de configuration :

La mise en ligne se fait en utilisant un logiciel ftp pour monter les pages, ou en passant directement par les fonctions ftp proposées sur le site du provider.

Il faut contrôler le site en ligne pour voir les éventuels problèmes de configuration.

### Tests sur différents navigateurs :

Le site peut avoir des réactions différentes suivant les navigateurs : il faut donc faire des tests.

Le respect de la norme W3C permet de minimiser les problèmes. Pour voir si le site passe le test de la norme W3C, voici le lien :

<http://validator.w3.org/>

### Gestion et maintenance :

Pour qu'un site dynamique fonctionne, il faut un ou plusieurs responsables du site pour gérer le back office, ainsi qu'un administrateur pour s'occuper de la maintenance en cas de problème technique.

## INFORMATIONS COMPLEMENTAIRES

web, c'est une toute petite partie d'internet.

Internet n'est rien d'autre qu'un réseau de transport de données.

Il permet de transporter un paquet de données d'un ordinateur A à un ordinateur B, rien d'autre.

Le web est construit au **dessus** d'internet.

Il met en oeuvre plein de choses (protocole HTTP, formats HTML, CSS, JPEG...).

Le web utilise internet pour transporter les données.

Il y a plein d'autres applications que le web qui sont construites au dessus d'internet:

- le mail (protocoles SMTP, POP3...)
- les newsgroups (Usenet, protocole NNTP)
- le chat (protocoles Jabber, AIM...)
- la vidéo (protocoles Real, etc.)
- le système DNS (pour donner des noms aux machines)
- les VPN (pour relier des réseaux informatique privés entre eux)
- et des centaines d'autres choses.

Bref, le web c'est la partie la plus visible et la plus connue d'internet, mais c'est loin d'être la plus importante.

Le web n'est vraiment qu'une partie d'internet.

Ou pour être plus exact: Le web est l'une des *utilisations* d'internet.