

# **Développement d'applications web**

Cours de la deuxième année licence  
Informatique (4ème semestre)

DR. MOHAMED AMINE BEGHOURA <sup>1</sup>

1. Maître de conférences - B - au département d'Informatique, Faculté des mathématiques et d'informatique, Université de Bordj Bou Arreridj.

# Table des matières

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Introduction au World Wide Web</b>                                     | <b>4</b>  |
| 1.1      | Définitions . . . . .   | 4         |
| 1.2      | historique . . . . .  | 4         |
| 1.3      | Architecture Client/Serveur . . . . .                                     | 5         |
| 1.4      | Protocole HTTP . . . . .  | 7         |
| 1.4.1    | Requête HTTP . . . . .  | 7         |
| 1.4.2    | Réponse HTTP . . . . .  | 8         |
| 1.5      | Notions de base de Web 2.0 (X.0) . . . . .                                | 9         |
| <b>2</b> | <b>Langages de programmation pour le Web</b>                              | <b>10</b> |
| 2.1      | Généralités : page statique, page dynamique et applications web . . . . . | 10        |
| 2.1.1    | Page statique . . . . .   | 11        |
| 2.1.2    | Page dynamique . . . . .  | 11        |
| 2.2      | Langages de balise . . . . .  | 12        |
| 2.3      | HTML . . . . .  | 12        |
| 2.3.1    | Qu'est ce que le HTML ? . . . . .   | 12        |
| 2.3.2    | Contexte d'exécution HTML . . . . .                                       | 12        |
| 2.3.3    | HTML de base . . . . .  | 13        |
| 2.3.4    | HTML 5.0 . . . . .  | 15        |
| 2.4      | Feuilles de style (CSS) . . . . .   | 20        |
| 2.4.1    | L'enregistrement du style . . . . .                                       | 21        |
| 2.4.2    | La structure d'une feuille CSS . . . . .                                  | 21        |
| 2.4.3    | Les sélecteurs id et classe . . . . .                                     | 22        |
| 2.4.4    | La balise <div> . . . . .   | 22        |
| <b>3</b> | <b>Langage de programmation coté serveur (PHP)</b>                        | <b>24</b> |
| 3.1      | Introduction . . . . .  | 24        |
| 3.2      | Syntaxe de base . . . . .   | 24        |
| 3.2.1    | Le passage du HTML au PHP . . . . .                                       | 25        |
| 3.2.2    | Les séparateurs d'Instructions . . . . .                                  | 25        |
| 3.2.3    | Les commentaires . . . . .  | 25        |
| 3.2.4    | La fonction echo . . . . .  | 26        |

|          |   |           |
|----------|---|-----------|
| 3.3      | Types, variables et opérateurs . . . . .                | 26        |
| 3.3.1    | Les types . . . . .                                     | 26        |
| 3.3.2    | Les variables . . . . .                                 | 27        |
| 3.3.3    | Les opérateurs . . . . .                                | 27        |
| 3.4      | Structures de contrôles . . . . .                       | 29        |
| 3.4.1    | Les conditions . . . . .                                | 29        |
| 3.4.2    | Les boucles . . . . .                                   | 32        |
| 3.5      | Les fonctions . . . . .                                 | 34        |
| 3.6      | Les tableaux . . . . .                                  | 35        |
| 3.6.1    | La création des tableaux en PHP . . . . .               | 35        |
| 3.6.2    | Tableaux indexés en PHP . . . . .                       | 35        |
| 3.6.3    | Les tableaux associatifs . . . . .                      | 36        |
| 3.6.4    | Tableaux multidimensionnels en PHP . . . . .            | 37        |
| 3.7      | Classes et objets . . . . .                             | 38        |
| 3.8      | Caractéristiques . . . . .                              | 39        |
| 3.8.1    | Gestion des erreurs . . . . .                           | 39        |
| 3.8.2    | Récupération des données d'un formulaire . . . . .      | 40        |
| 3.8.3    | Connexions persistantes aux Bases de Données. . . . .   | 41        |
| 3.8.4    | Gestion des sessions . . . . .                          | 44        |
| <b>4</b> | <b>Services Web : notions de base</b>                   | <b>47</b> |
| 4.1      | Introduction . . . . .                                  | 47        |
| 4.2      | Architecture orientée services (SOA) . . . . .          | 47        |
| 4.3      | Standards de base pour les services Web . . . . .       | 49        |
| 4.3.1    | SOAP . . . . .  | 49        |
| 4.3.2    | WSDL . . . . .  | 49        |
| 4.3.3    | UDDI . . . . .  | 49        |
| 4.3.4    | Une application web sous forme de service web . . . . . | 50        |
| <b>5</b> | <b>Exercices</b>  | <b>55</b> |

# Préface

## Objectifs de l'enseignement de la matière

Présenter les systèmes d'information dans le contexte Internet. Le module initie à la programmation Web via les langages HTML, JavaScript et PHP. En plus, il initie au développement des services web. Une étude pratique renforce les concepts acquis.

## Recommandations

Insister sur une étude de cas durant le module.

## Connaissances préalables recommandées

Notions de base d'internet, initiation en HTML.

## Mode d'évaluation

Contrôle continu et Examen écrit.

## Liens utiles

- <http://openclassrooms.com/>
- <http://www.developpez.com/>
- <https://www.w3schools.com/>
- <http://tutsplus.com/> et <http://code.tutsplus.com/>

# 1

## Introduction au World Wide Web

### 1.1 Définitions

Le réseau Internet est un réseau de machines (ordinateurs, serveurs, etc.) inter-connectées et qui offre une variété de services tels que : courriers électroniques, transfert de fichiers et le web.

Le World Wide Web (WWW, W3) est un système d'information constitué d'un ensemble de documents interdépendants et disponibles sur le réseau Internet.

Les documents disponibles sur le web sont appelés pages web, ils sont accessibles à travers un navigateur (Browser en Anglais) ou bien un client web. Ces pages web peuvent contenir du texte, des images, des vidéos ou autres composants.

Les pages web sont formés avec ce qu'on appelle « HTML », qui signifie HyperText Markup Language. Il est la base pour presque toutes les pages Web (Même s'il peut y avoir des documents, images, vidéos, etc).

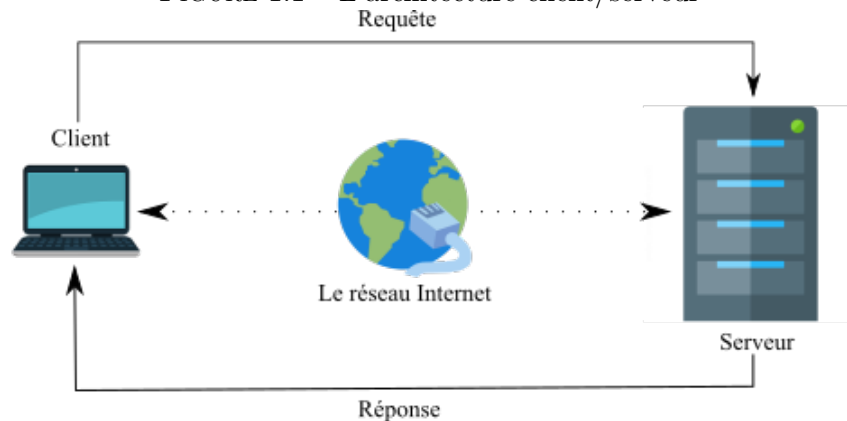
#### **Remarque :**

Ne pas confondre entre le World Wide Web et le terme Internet qui est le réseau informatique mondial accessible au public principalement composé par des millions de réseaux aussi bien publics que privés.

### 1.2 historique

L'histoire du web a commencé avec Tim Berners-Lee qui est un informaticien britannique et ancien employé du CERN. Il est considéré comme l'inventeur du Web. En Mars 1989, Berners-Lee a écrit une proposition de ce qui allait devenir le World Wide Web. La proposition de 1989 a été conçue

FIGURE 1.1 – L'architecture client/serveur



pour un système de communication plus efficace pour l'entreprise CERN. Par contre, Tim Berners-Lee a réalisé que le concept qu'il a proposé pourrait être déployé à travers le monde.

*Berners-Lee* et un autre informaticien belge (*Robert Cailliau*) ont proposé le concept du WWW en 1990 et qui consiste à utiliser des documents hypertexte pour relier et accéder à des informations sur un réseau de nœuds dans lequel l'utilisateur peut naviguer d'un document à un autre à volonté.

### 1.3 Architecture Client/Serveur

Le Web est un service Internet qui fonctionne selon les règles de l'architecture client/serveur. Un émetteur (client) envoie une requête sous forme d'un lien URL vers un serveur qui répond par l'envoi du fichier demandé et qui sera visualisé sur le client.

#### Le fonctionnement de l'architecture client/serveur

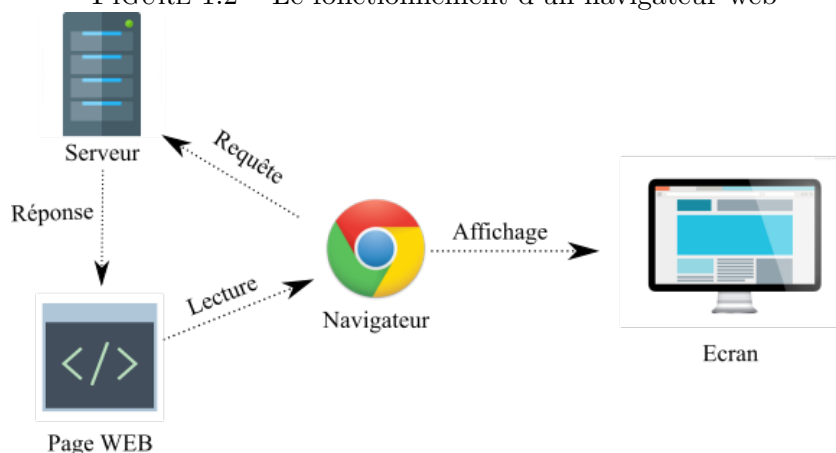
La communication entre le navigateur et le serveur se fait en deux temps (voir la figure 1.1) :

- Le navigateur effectue une requête HTTP (exemple : demande d'affichage de la page `index.html`)
- Le serveur traite la requête puis envoie une réponse HTTP (le serveur envoie la page `index.html` au client web)
- Le navigateur interprète et affiche alors la page à l'utilisateur.

#### Les entités impliquées dans la communication client/serveur

**Un serveur :** est un dispositif informatique utilisé pour héberger des sites web sur Internet ou un intranet.

FIGURE 1.2 – Le fonctionnement d'un navigateur web



**Un serveur logiciel :** est le logiciel utilisé sur le serveur pour exécuter les requêtes du client. La fonction principale d'un serveur web est de stocker et délivrer les pages webs. Il existe plusieurs serveurs à l'exemple de :

- Apache HTTP Server de la « Apache Software Foundation » (voir [www.wampserver.com](http://www.wampserver.com) ou [ouwww.easyphp.org](http://ouwww.easyphp.org))
- Apache Tomcat de la Apache Software Foundation, l'évolution de Apache pour J2EE ;
- Google Web Server de Google ;
- Internet Information Services (IIS) de Microsoft ;

**Un client :** est le dispositif informatique utilisé pour l'émission des requêtes vers le serveur.

**Un client web :** est le logiciel utilisé pour gérer les requêtes et interpréter et/ou afficher les résultats de la requête à l'exemple des navigateurs web (Firefox, Chrome, Opera, etc.).

### Le fonctionnement d'un navigateur web

Le navigateur désigne une page web par son adresse URL (Uniform Resource Locator). Le serveur répond aux demandes en envoyant les pages au navigateur Web. Le navigateur interprète et affiche alors les pages à l'utilisateur (voir la figure 1.2).

Un client web utilise les adresses URL pour accéder aux ressources disponibles sur le web. Un URL est un identifiant unique qui désigne une page appartenant à un site web. Un URL comporte le nom du protocole de communication, le nom de domaine (DNS), le chemin vers la ressource demandée, le nom de la ressource ainsi qu'un ensemble de paramètres additionnels pour accéder à la ressource.

#### Exemple :

Voici un exemple de format d'adresse URL :

`http://www.exemple.com/test-dir/index.html?var1=temp1&var2=temp2`

- **http** : Le protocole de communication
- **www.exemple.com** : Le nom du domaine (DNS, unique)
- **Test-dir** : Répertoire sur la machine
- **Index.html** : Nom du document à afficher sur le navigateur
- **?var1=temp1&var2=temp2** : des paramètres additionnels pour accéder à la ressource index.html ou « ? » est un séparateur qui indique la liste des paramètres (var1 et var2 dans l'exemple) et le « & » est un séparateur qui est utilisé pour les variables

## 1.4 Protocole HTTP

Un protocole réseau est un ensemble de règles et procédures de communication utilisées par toutes les stations qui échangent des données sur le réseau.

Chaque service sur Internet a son propre protocole (sa manière d'envoyer des fichiers dans le système). Le protocole pour le Web est HTTP (HyperText Transfer Protocol), il est de permet le transfert de fichiers (essentiellement au format HTML) entre un navigateur (le client web) et un serveur Web. Le protocole permet de transférer des messages avec des en-têtes décrivant le contenu du message en utilisant un codage du type MIME (Multipurpose Internet Mail Extensions).

### 1.4.1 Requête HTTP

Une requête HTTP est un ensemble de lignes envoyé au serveur par le navigateur (voir la figure 1.3). Elle comprend :

- **Une ligne de requête** : c'est une ligne précisant
  - la méthode qui doit être appliquée (voir la table 1.1),
  - le chemin du document demandé,
  - et la version du protocole utilisée.
- **Les champs d'en-tête de la requête** : il s'agit d'un ensemble de lignes facultatives permettant de donner des informations supplémentaires sur la requête et/ou le client (exemple : le nom du navigateur, système d'exploitation, ...etc). Chacune de ces lignes est composée d'un nom qualifiant le type d'en-tête, suivi de deux points ( :) et de la valeur de l'en-tête.
- **Le corps de la requête** : c'est un ensemble de lignes optionnelles devant être séparées des lignes précédentes par une ligne vide.



FIGURE 1.3 – La structure d’une requête HTTP

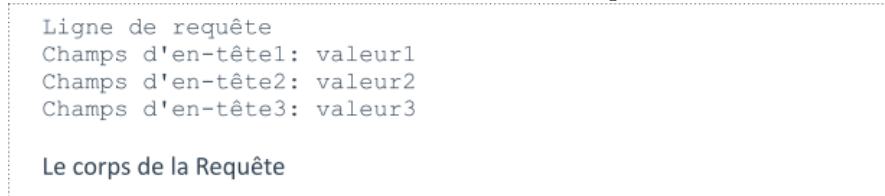


TABLE 1.1 – Les méthodes des requêtes HTTP

| Méthode | Description   |
|---------|---|
| GET     | Requête de la ressource située à l’URL spécifiée              |
| HEAD    | Requête de l’en-tête de la ressource située à l’URL spécifiée |
| POST    | Envoi de données au programme situé à l’URL spécifiée         |
| PUT     | Envoi de données à l’URL spécifiée                            |
| DELETE  | Suppression de la ressource située à l’URL spécifiée          |

**Exemple :**

```

1 GET www.exemple.com HTTP/1.0
2 Accept: text/html
3 If-Modified-Since: Sunday, 2-Friday-2015 14:37:11 GMT
4 User-Agent: Mozilla/23.0

```

### 1.4.2 Réponse HTTP

Une réponse HTTP est un ensemble de lignes envoyées au Client Web par le serveur (figure 1.4). Elle comprend :

- **Une ligne de statut** : c’est une ligne précisant la version du protocole utilisé et l’état du traitement de la requête à l’aide d’un code et d’un Message explicatif.
- **Les champs d’en-tête de la réponse** : il s’agit d’un ensemble de lignes facultatives permettant de donner des informations supplémentaires sur la réponse et/ou le serveur. Chacune de ces lignes est composée d’un nom qualifiant le type d’en-tête, suivi de deux points ( :) et de la valeur de l’en-tête
- **Le corps de la réponse** : il contient le document demandé.

**Exemple :**

```

1 HTTP/1.0 200 OK
2 Date: Fri, 2 Jan 2015 23:59:59 GMT
3 Content-Type: text/html
4 Content-Length: 1354
5
6 <html>

```

```
7|<body>
8|<h1>Hello World! </h1>
9|</body>
10|</html>
```

FIGURE 1.4 – La structure d’une réponse HTTP

```
Ligne de statut
Champs d'en-tête1: valeur1
Champs d'en-tête2: valeur2
Champs d'en-tête3: valeur3

Le corps de la Réponse
```

## 1.5 Notions de base de Web 2.0 (X.0)

Le web 2.0 fait référence aux site web qui focalise sur le contenu généré par les utilisateurs. Il peut permettre aux utilisateurs d’interagir et de collaborer les uns avec les autres en tant que créateurs de contenu généré par les utilisateurs dans une communauté virtuelle, contrairement à la première génération de sites Web 1.0. Exemples : sites de réseaux sociaux tels que Facebook, blogues, wikis, YouTube.

## 2

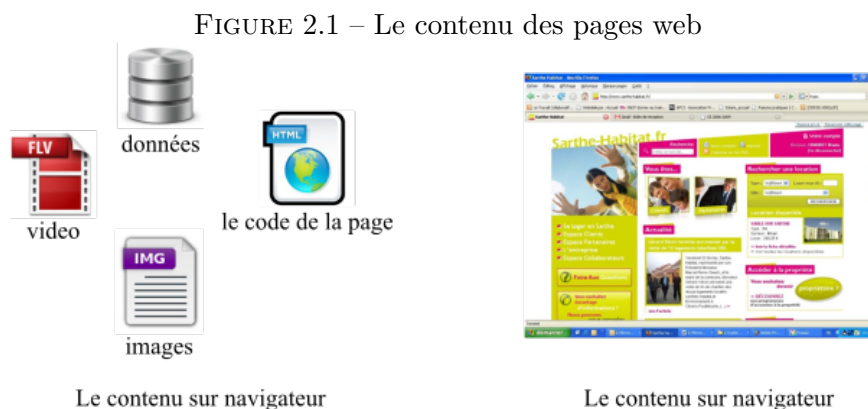
# Langages de programmation pour le Web

## 2.1 Généralités : page statique, page dynamique et applications web

Un ensemble de pages web et de ressources liées forment un site web (de l'anglais web site) qui est accessible via une adresse web. Un site web est hébergé sur un serveur web, lui-même accessible en utilisant un client web via un réseau internet ou intranet.

Dans le premier chapitre, nous avons parlé du contenu d'une page web ou nous avons vu qu'une page web peut contenir du texte, des images, des données, des vidéos, etc. Le contenu d'une page web (sauf les textes, s'ils sont écrits directement sur la page) est simplement relié à la page par des lignes de code (qui marque sa position vis-à-vis du texte et la page).

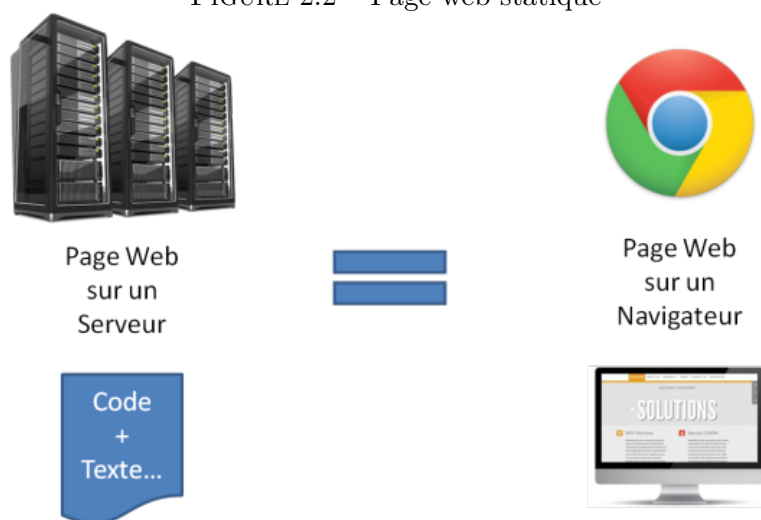
Le contenu et la mise en forme d'une page web, tel qu'on le voit sur écran, sont affichés qu'au moment du chargement de la page au travers d'un navigateur web (Figure 2.1).



### 2.1.1 Page statique

Une page statique est une page dont l'intégralité du code va être interprété directement par le navigateur (page programmée = page affichée).

FIGURE 2.2 – Page web statique



Un ordinateur qui se connecte au serveur, demande une page. Celle-ci lui est directement servie (elle est stockée toute prête sur le serveur).

### 2.1.2 Page dynamique

Le serveur, sur lequel la page est stockée, devra réaliser des opérations avant de pouvoir afficher le contenu de la page demandée. En fonction des « paramètres » (sélection réalisée par l'internaute), le serveur génère le contenu.

FIGURE 2.3 – Page web dynamique



Le contenu est obtenu en combinant l'utilisation d'un langage de scripts ou de programmation et une base de données (exemple, le langage PHP et

MySQL).

Pour une page web il existe plusieurs extensions de document telles que : `.html`, `.htm`, `.php`, `.aspx`, `.xml` etc. Les pages dites statiques portent en général l'extension `.html` et les pages dites dynamiques portent l'extension `.php`, `.aspx` ou d'autres.

## 2.2 Langages de balise

Les langages de balises représentent une classe de langages spécialisés dans l'enrichissement d'information textuelle. Ils utilisent des balises, unités syntaxiques délimitant des séquences de caractères à l'intérieur d'un flux de caractères (par exemple un fichier texte).

L'inclusion de balises permet de transférer à la fois la structure du document et son contenu. Cette structure est compréhensible par un programme informatique, ce qui permet un traitement automatisé du contenu.

Les langages à base de balises servent à structurer ou formater des documents. Il suffit en effet d'encadrer des portions de texte par des balises pour utiliser une fonctionnalité du langage.

Parmi les langages de définition des documents les plus populaires on compte DocBook, LaTeX, XML et le **HTML** que nous allons voir dans ce chapitre.

## 2.3 HTML

### 2.3.1 Qu'est ce que le HTML ?

L'HyperText Markup Language, généralement abrégé HTML, est le format de données conçu pour représenter les pages web. C'est un langage de balisage permettant d'écrire de l'hypertexte.

Le HTML permet : de structurer sémantiquement et mettre en forme le contenu, d'inclure des ressources multimédias dont des images, des formulaires de saisie et des programmes informatiques dans des pages web.

### 2.3.2 Contexte d'exécution HTML

Un fichier écrit en langage HTML n'est autre qu'un fichier texte, mais dont le contenu est structuré à l'aide de repères que l'on appelle des éléments. Chaque élément est constitué de balises et d'attributs qui permettent d'apporter des informations sur son contenu.

Pour pouvoir créer un document HTML, il faut créer un fichier texte et lui attribuer l'extension `.html`. l'édition du document se fait à l'aide des éditeurs de texte tels que : notepad, notepad++, brackets, gedit, etc.

### 2.3.3 HTML de base

HTML possède de nombreux éléments de gestion de l'apparence (ou formatage) de la page qui se rapprochent de ce que nous connaissons sur les éditeurs de texte : mise en gras (b), en italique (i), indentation, taille ou couleur des caractères (font).

#### Les balises HTML

On trouve dans HTML des balises qui permettent de structurer les listes, titres, tableaux, citations, adresses... autant de structures du texte qui ne définissent pas l'apparence finale à l'écran.

La balise est le moyen de communiquer avec le navigateur. Elle apparaît entre les caractères réservés < et >. les balises (ou tags ou marqueurs) contrôlent l'affichage sur navigateur mais elles ne s'affichent pas sur la page web, elles sont interprétées par le navigateur.

**Exemple :** Pour créer un paragraphe et le souligné en HTML, on utilise les balises <p> et <u> comme suit :

```
1|<p> <u> Ceci est un paragraphe </u> </p>
```

La balise <p> nous permet d'indiquer l'emplacement du paragraphe dans la page tant dis que la balise <u> le souligne. Le résultat sur navigateur sera le suivant :

Ceci est un paragraphe

#### Les attributs des balises

Une balise peut comporter de zéro à plusieurs attributs. Les attributs sont des informations complémentaires qui la caractérisent. Ils se présentent sous la forme nom\_attribut="valeur", par exemple :

```
1|<p align= "right"> hello world!</p>
```

l'attribut align est utilisé dans la balise <p> avec la valeur "right" pour aligner le texte sur la droite.

#### Les types des balises

On distingue deux types de balises :

- **Les balises simples**, ce sont des balises qui sont dites "vides", c'est-à-dire qu'elles ne vont contenir aucune autre balise HTML. Ces balises n'ont pas besoin d'être fermées (Exemple : ).

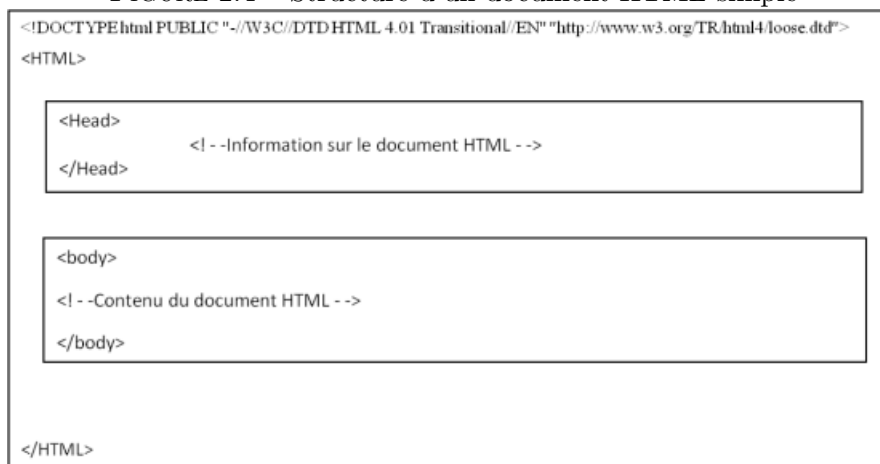
- **Les balises doubles**, elles nécessitent deux balises, une **ouvrante** et une **fermante** dans lesquelles on va pouvoir mettre d'autres balises ou du texte. La balise fermante est identique à la balise ouvrante, la différence qu'elle contient un "/" pour indiquer à quel endroit on la ferme (exemple : `<p> ici c'est du texte!!!< /p>`).

**Note :** consulter <http://www.w3schools.com/tags/> pour voir la liste des balises HTML.

### Ossature d'un document HTML (entête, corps, Liens, ... )

Un document HTML est fichier texte dont l'extension doit être \*.html (si la page est statique). Pour définir un document HTML, il faut impératif utiliser au moins une structure minimale qui a la forme suivante :

FIGURE 2.4 – Structure d'un document HTML simple



La première ligne du document (Figure 5) s'appelle le **DOCTYPE**. C'est une balise indispensable pour conserver la compatibilité du rendu de la page sur les différents navigateurs modernes.

Après le DOCTYPE, vient la balise `<html>`. C'est une balise racine qui doit être déclarée une seule fois dans un document HTML pour que le navigateur identifie qu'il s'agit d'un document HTML et pour contenir le reste des balises.

Le contenu de la balise `<html>` est principalement séparé sur deux parties :

- Une partie qui contient une déclaration des informations relatives au document. Cette partie est définie par la balise `<head>` (Les balises utilisées dans cette partie sont : `<title>`, `<meta />`, `<link />`, `<script>`, `<style>`)

- La deuxième partie qui est contenue entre la balise `<body>` sert à définir le contenu qui va être affiché sur le navigateur.

```
1|<!DOCTYPE html>
2|<html>
3|  <head>
4|    <title>Titre de la page</title>
5|    <meta http-equiv="content-type" content="text/html; charset=
6|      utf-8">
7|  </head>
8|  <body>
9|    <!-- le contenu de la page -->
10|  </body>
11|</html>
```

### 2.3.4 HTML 5.0

#### Les liens `<a>`

Les liens indiquent au navigateur où aller en utilisant un attribut href, qui stocke une URL.

```
1|<a href="http://google.com">Google !</a>
```

#### Les commentaires

Les commentaires HTML sont parfois utilisés dans le code pour expliquer certaines parties du balisage.

```
1|<!-- Ceci est un commentaire ! -->
```

#### Les titres `<h1>`, `<h2>`, ... ,`<h6>`

Les éléments de titre tels que `<h1>`, `<h2>`, `<h3>`, ... vous permettent d'utiliser six niveaux de titres de document, allant du plus grand au plus petit, découpant le document en sections logiques.

```
1|<h1> Section </h1>
2|<h2> Sous-section</h2>
```



## Règles horizontales <hr>

Cette balise crée une ligne noire d'un pixel d'épaisseur qui traverse tout son conteneur.

```
1 | Ce text est séparé
2 | <hr>
3 | ... de ce texte!
```

## les images <img>

La balise <img> intègre une image dans votre code HTML. L'attribut 'src' indique au navigateur où trouver l'image.

```
1 | <img src='mon_image.jpg' />
```

## Sauts de ligne <br/>

Cette balise est utilisée dans un bloc de texte pour forcer un saut de ligne.

```
1 | <p> un texte <br/> la suite du texte </p>
```

## Les listes non-ordonnées <ul>

Les listes non ordonnées ne sont que des listes dont les éléments sont désignés par des puces.

```
1 | <ul>
2 |   <li>Informatique</li>
3 |   <li>Mathématiques</li>
4 |   <li>Recherche Operationnelle</li>
5 | </ul>
```

## Les listes ordonnées <ol>

Les éléments des listes ordonnées sont indiqués par des chiffres.

```
1 | <ol>
2 |   <li>Le premier élément!</li>
3 |   <li>Le deuxième élément!</li>
4 |   <li>Le troisième élément!</li>
5 | </ol>
```

## Les tableaux <table>, <th>, <tr>, <td>

Dans cette section, nous présentons un exemple de création d'un tableau en HTML. Les balises nécessaires pour ceci sont : <table>, <th>, <tr> et <td>. Le code HTML d'une page qui va contenir le tableau est le suivant :

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Mon tableau</title>
5   </head>
6   <body>
7
8     <table >
9       <tr>
10        <th>Module</th>
11        <th>Credit</th>
12      </tr>
13      <tr>
14        <td>Web</td>
15        <td>4</td>
16      </tr>
17      <tr>
18        <td>GL</td>
19        <td>4</td>
20      </tr>
21    </table>
22
23  </body>
24 </html>
```

Le résultat du code sur le navigateur est le suivant :

| Module | Credit |
|--------|--------|
| Web    | 4      |
| GL     | 4      |

## Les formulaires <form>, <input>, <select>, ...

En HTML, les formulaires apportent l'interactivité à une page web. Ils permettent aux utilisateurs d'introduire des données, sélectionner des choix ou de valider quelque chose.

Cependant, le langage HTML ne permet pas d'analyser ou de traiter ces informations. Ceci, nécessite l'utilisation d'un autre langage complémentaire tel que le langage PHP que nous allons voir dans le chapitre suivant de ce cours.

La balise pour créer un formulaire en HTML est : <form>. L'objectif est de signaler que tout ce qui est entre cette balise double fait partie du formulaire. Il faut inclure les champs du formulaire selon le besoin. Pour

cela, il existe plusieurs types de balises telles que : `<input />`, `<textarea />`, ... etc.

**Exemple :** le code HTML suivant permet d'afficher un formulaire de login

```
1 <form>
2   <label>Votre pseudo :</label>
3   <input type="text" name="pseudo" />
4
5   <label for="pass">Votre mot de passe :</label>
6   <input type="password" name="pass" />
7
8   <input type="submit" value="Envoyer" />
9 </form>
```

Le résultat de ce code sur le navigateur est le suivant (table 2.1) :



La balise `<input />` crée des champs pour saisir les informations. On peut ajouter un certain nombre d'autres attributs à la balise `<input />` pour personnaliser son fonctionnement :

- On peut agrandir le champ avec **size**.
- On peut limiter le nombre de caractères que l'on peut saisir avec **maxlength**.
- On peut pré-remplir le champ avec une valeur par défaut à l'aide de **value**.
- On peut donner une indication sur le contenu du champ avec **placeholder**. Cette indication disparaîtra dès que le visiteur aura cliqué à l'intérieur du champ.
- On peut donner un nom (unique) pour le champ de texte avec l'attribut **name**.

Pour personnaliser le type de la balise `<input>` selon l'usage à l'aide de l'attribut `type` (Voir la table 2.1) :

TABLE 2.1 – Les valeurs du l’attribut type de la balise input

| input                    | Type                | Description  |
|--------------------------|---------------------|--|
| <input type="text"/>     | Texte               | Création d’une zone de texte simple                          |
| <input type="password"/> | Texte               | Zone de mot de passe   |
| <input type="email"/>    | Texte               | Zone pour saisir une adresse email                           |
| <input type="url"/>      | Texte               | Zone pour contenir un URL                                    |
| <input type="tel"/>      | Numéro de téléphone | saisie de numéros de téléphone                               |
| <input type="number"/>   | Nombre              | saisir un nombre entier                                      |
| <input type="range"/>    | Nombre              | sélectionner un nombre avec un curseur (aussi appelé slider) |
| <input type="date"/>     | Date                | sélection de date  |
| <input type="file"/>     | Fichier             | Sélection d’un fichier pour upload                           |

La balise <input /> peut prendre d’autres formes que des champs à saisie, qui sont :

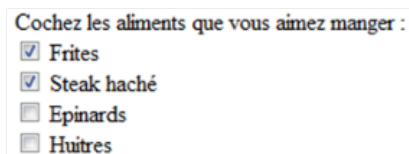
#### Case à cocher :

```

1 <p>Cochez les aliments que vous aimez manger :</p>
2 <input type="checkbox" name="frites" /><label for="frites">
  Frites</label><br />
3 <input type="checkbox" name="steak" /><label for="steak">Steak
  haché</label><br />
4 <input type="checkbox" name="epinards" /><label for="epinards">
  Epinards</label><br />
5 <input type="checkbox" name="huitres" /><label for="huitres">
  Huitres</label>

```

#### Résultat :



Cochez les aliments que vous aimez manger :

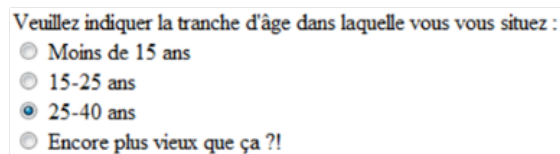
- ☒ Frites
- ☒ Steak haché
- ☐ Epinards
- ☐ Huitres

On peut associer un attribut `for` à la balise `<label>` pour indiquer le nom du champ input dont on veut lier le libellé.

#### Les zones d'options :

```
1 <p>Veuillez indiquer la tranche d'âge dans laquelle vous vous
   situez :</p><br />
2 <input type="radio" name="age" value="moins15" /><label>Moins
   de 15 ans</label><br />
3 <input type="radio" name="age" value="medium15-25" />15-25 ans<
   br />
4 <input type="radio" name="age" value="medium25-40" />25-40 ans<
   br />
5 <input type="radio" name="age" value="medium25-40" />plus de 40
   ans<br />
```

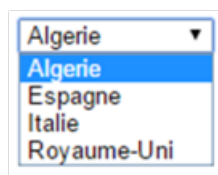
#### Résultat :



#### Les listes déroulantes :

```
1 <select name="pays" id="pays">
2 <option value="algerie">Algerie</option>
3 <option value="espagne">Espagne</option>
4 <option value="italie">Italie</option>
5 <option value="royaume-uni">Royaume-Uni</option>
6 </select>
```

#### Résultat :



**Note :** Pour récupérer ce que les visiteurs ont saisi, le langage HTML ne suffit pas. Il faut utiliser un langage « serveur » comme PHP.

## 2.4 Feuilles de style (CSS)

Les feuilles de style en cascade, généralement appelées **CSS** de l'anglais **Cascading Style Sheets**, forment un langage informatique qui décrit la présentation des documents HTML.

La feuille de style CSS contient la déclaration de toute la mise en forme des pages : le positionnement des éléments, l'image de fond, les polices de caractère, les couleurs, etc. Celle-ci sera liée à chaque page html. Ainsi, lorsqu'on en modifiera un élément, cela engendra le même effet sur les pages html liées à la feuille de style CSS.

### 2.4.1 L'enregistrement du style

Il existe donc trois façons d'enregistrer les styles :

**1. Dans une feuille de style externe.** Enregistrer le code CSS dans un fichier s'appelant (par exemple) "style.css", et mettre dans l'en-tête de la page html (entre les balises <head></head>) :

```
1 | <link href="style.css" rel="stylesheet" media="all" type="text/
   | css">
```

**2. Dans une feuille de style interne.** Pour déclarer des styles qui ne s'appliqueront qu'à la page considérée, les styles sont à déclarer dans la balise <head></head> entre les balises suivantes :

```
1 | <style type="text/css">
2 | ....
3 | </style>
```

**3. Dans le code au sein des balises html (inline).** Si l'on veut attribuer un style à un seul endroit, on peut déclarer le style à l'intérieur d'une balise html à l'aide de l'attribut style. Par exemple :

```
1 | <p style="text-align:center; color:red"> Ici mon texte. </p>
```

### 2.4.2 La structure d'une feuille CSS

la structure d'une feuille CSS est la suivante ou pour chaque déclaration, la structure est toujours la même :

```
1 | Sélecteur {
2 |   Propriété: valeur;
3 | }
```

- Le sélecteur, c'est la balise (body ; h1 ; p, etc.), l'identifiant (id) ou la classe (class) ;
- La propriété, c'est l'attribut qu'on veut appliquer (font ; background ; margin ; etc.). vous trouveriez la liste des propriétés CSS sur ce lien <https://www.w3schools.com/cssref/default.asp>.
- Et enfin la valeur qui précise les caractéristiques de la propriété

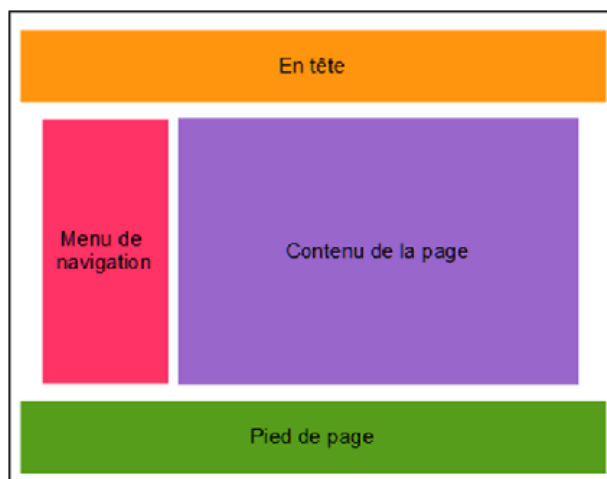
### 2.4.3 Les sélecteurs id et classe

Les sélecteurs **Class** et **id** sont différents d'où le sélecteur **id** ne peut être utilisé qu'une seule fois dans une page alors que le sélecteur **class** peut être utilisé plusieurs fois dans la même page.

### 2.4.4 La balise <div>

Pour une mise en page souple et cohérente, on divisera la page en "blocs" avec la balise <div></div> (des div, appelés aussi "boîtes" ou "calques"), qui ont l'avantage de pouvoir être déplacés de gauche à droite, ou de haut en bas grâce aux CSS.

**Exemple :** Considérons la page web dans la figure suivante.



Les codes HTML et css qui nous permettent de reproduire cet affichage sont les suivants :

```
1 <div id="entete">
2   En tete
3 </div>
4
5 <div id="main">
6   <div id="menu">
7     Menu
8   </div>
9
10  <div id="contenu">
11    Contenu
12  </div>
13 </div>
14
15 <div id="footer">
16   Pied de Page
```

17|</div>

```
1 #entete, #menu, #contenu, #footer {
2   padding:1px 0;
3 }
4 #entete {
5   background-color:#FF9900;
6   text-align:center;
7 }
8 #main {
9   max-width:960px;
10  margin:auto;
11 }
12 #menu {
13   float:left;
14   width:240px;
15   background-color:#FF3366;
16 }
17 #contenu {
18   margin-left:245px;
19   background-color:#9966FF;
20 }
21 #footer {
22   background-color:#669933;
23   text-align:center;
24   clear:both;
25 }
```



## 3

# Langage de programmation coté serveur (PHP)

### 3.1 Introduction

PHP (officiellement, ce sigle est un acronyme récursif pour "PHP : Hyper text Preprocessor") est un langage de script généraliste, Open Source, et spécialement conçu pour le développement d'applications web. Il peut être intégré facilement au page HTML.

Le langage PHP est un langage de programmation interpréter (et non pas compiler) et supporté par de nombreux serveurs WEB.

Le code PHP est inséré dans les pages WEB et repéré par un serveur WEB (si il est muni de l'extension PHP) qu'il l'enverra à PHP pour l'interpréter.

Grâce à ces portions de code PHP insérées dans les pages WEB, PHP permettra d'écrire des pages WEB à contenus dynamiques (vous écrivez une page HTML avec du code PHP inclus à l'intérieur afin de réaliser une action précise.)

Ce qui distingue le PHP des langages de script comme le Javascript est que le code écrit en PHP est exécuté sur le serveur web. Le client web ne reçoit que le résultat du script, sans aucun moyen d'avoir accès au code qui a produit ce résultat.

### 3.2 Syntaxe de base

Pour que le serveur qui héberge la page web puisse repérer les portions de code en PHP, il suffit simplement de lui indiquer le début ainsi que la fin du code PHP en utilisant les deux marqueurs suivant :

- La balise `<?php` marque le début d'une portion de code PHP
- La balise `?>` marque la fin d'une portion de code PHP

**Exemple : déclaration d'un bloc de code PHP**

```
1|<?php  ?>
```

### 3.2.1 Le passage du HTML au PHP

Il faut changer l'extension de ce fichier en .php lorsque vous insérez le moindre petit bout de code PHP dans une page HTML, (si vous avez une page nommée index.html et que vous y insérez du code PHP, il vous faudra la renommer en index.php). Pour programmer en PHP il faut :

- Un éditeur de texte pour écrire vos portions de code en PHP
- Un environnement de développement afin de tester ces portions de code (exemples : EasyPHP (Windows), wampserver (Windows), mamp (Apple), lamp (linux))

#### Exemple : insertion du code PHP dans une page HTML

```
1|<html>
2|<head>
3|<title>Test</title>
4|</head>
5|
6|<body>
7|  <p>un bout de code en HTML</p>
8|  <?php
9|    echo 'Mon premier script en PHP';
10|  ?>
11|</body>
12|</html>
```

### 3.2.2 Les séparateurs d'Instructions

Comme en C, PHP requiert que les instructions soient terminées par un point-virgule (;) à la fin de chaque instruction. La balise fermante d'un bloc de code PHP implique automatiquement un point-virgule.

#### Exemple : l'utilisation du séparateur d'instructions

```
1|<?php
2|    echo 'Ceci est un test';
3|?>
```

### 3.2.3 Les commentaires

Comme tous les langages de programmation, il est possible de commenter le code source en PHP. Pour cela, il y a deux méthodes :

- pour commenter une seule ligne de code PHP, on précédera cette ligne de deux slashes //
- pour commenter une portion de code, on précédera la première ligne de code que l'on souhaite commenter par un /\* et on fera suivre la dernière ligne de code que l'on souhaite commenter par un \*/

#### Exemple : Insertion des commentaires dans un code php

```

1 <?php
2 // ceci est un commentaire sur une seule ligne
3
4 /* ceci est
5 un commentaire
6 sur plusieurs lignes */
7 ?>

```

### 3.2.4 La fonction echo

Cette fonction permet d'afficher à l'écran des chaînes de caractères définie entre deux apostrophes " ... ", qui peuvent être définies directement par l'utilisateur.

#### Exemple : affichage d'une chaîne de caractère.

```

1 <?php
2 echo "Bonjour";
3 ?>

```

Les chaînes de caractères peuvent être des messages (texte) comme elles peuvent être un code HTML. Les paramètres de la fonction echo peuvent ne pas être entourés de parenthèses.

#### Exemple : affichage d'un code HTML.

Cette portion du code affichera l'image mon\_image.jpg sur la page web.

```

1 < ?php
2 echo "<img src= 'mon_image.jpg'>";
3 ?>

```

## 3.3 Types, variables et opérateurs

### 3.3.1 Les types

Le langage PHP support les types suivants :

- **Chaîne de caractères (String)** : une séquence de caractères.
- **Entier (Integer)** : un nombre non-décimale entre -2,147,483,648 et 2,147,483,647.
- **Réel (Float)** : nombre décimale.
- **Booléen (Boolean)** : TRUE ou FALSE.
- **Tableau (Array)** : le stockage de plusieurs valeurs dans la même variable.
- **Object** : une instance d'une classe qui doit être définie au préalable.

### 3.3.2 Les variables

Pour déclarer des variables en PHP il faut que :

- les variables doivent être représentées par une chaîne de caractères, ayant toujours comme premier caractère, le caractère dollar (\$).
- Les variables peuvent avoir n'importe quelle lettre en deuxième caractère du moment qu'il ne s'agit pas d'un chiffre.
- Les noms de variables ne peuvent pas contenir d'espace.

Pour assigner une valeur à une variable, il faut d'utiliser l'opérateur =, tout en prenant soin de placer la variable qui reçoit le résultat d'une opération à gauche du signe =.

#### Exemple : déclaration correcte des variables en PHP

```

1 <? php
2 $nom = "bonjour";
3 // $nom contient la chaîne de caractère bonjour.
4
5 $mon_chiffre = 12;
6 // $mon_chiffre contient la valeur 12.
7
8 $5toto = "test";
9 // Cette déclaration n'est pas valide car le nom de la variable
   commence par un chiffre.
10 ?>
```

### 3.3.3 Les opérateurs

Les opérateurs sont utilisés pour effectuer des opérations sur les variables. Le langage PHP est doté d'un ensemble d'opérateurs divisé sur 7 sous-ensembles :

TABLE 3.1 – Les opérateurs arithmétiques

| Opérateur | Nom            | Exemple      | Résultat                            |
|-----------|----------------|--------------|-------------------------------------|
| +         | Addition       | $\$x + \$y$  | Somme de $\$x$ et $\$y$             |
| -         | soustraction   | $\$x - \$y$  | Différence de $\$x$ et $\$y$        |
| *         | Multiplication | $\$x * \$y$  | Produit de $\$x$ et $\$y$           |
| /         | Division       | $\$x / \$y$  | Quotient de $\$x$ et $\$y$          |
| %         | Modulus        | $\$x \% \$y$ | Le reste de $\$x$ divisée sur $\$y$ |

TABLE 3.2 – Les opérateurs d’affectation

| Affectation | similaire que... | Description  |
|-------------|------------------|--|
| $x = y$     | $x = y$          | La variable sur la gauche de l’opérateur recevoir la valeur de la variable sur la droite |
| $x += y$    | $x = x + y$      | Addition   |
| $x -= y$    | $x = x - y$      | soustraction   |
| $x *= y$    | $x = x * y$      | Multiplication   |
| $x /= y$    | $x = x / y$      | Division   |
| $x \% = y$  | $x = x \% y$     | Modulus  |

TABLE 3.3 – Les opérateurs de comparaison

|     | Name              | Exemple       | Résultat   |
|-----|-------------------|---------------|--|
| ==  | égalité           | $\$x == \$y$  | Renvoi true si $\$x$ est égal à $\$y$                                |
| === | Identique         | $\$x === \$y$ | Renvoi true si $\$x$ est égal à $\$y$ , et ils sont du même type     |
| !=  | Pas égal          | $\$x != \$y$  | Renvoie true si $\$x$ est différent à $\$y$                          |
| <>  | Pas égal          | $\$x <> \$y$  | Renvoi true si $\$x$ est différent $\$y$                             |
| !== | Non-identique     | $\$x !== \$y$ | Renvoie true si $\$x$ est différent à $\$y$ , et d’un type différent |
| >   | Supérieur à       | $\$x > \$y$   | Renvoi true si $\$x$ est supérieur à $\$y$                           |
| <   | Inférieur à       | $\$x < \$y$   | Envoi true si $\$x$ est inférieur à $\$y$                            |
| >=  | Supérieur ou égal | $\$x >= \$y$  | Renvoi true si $\$x$ est supérieur ou égal a $\$y$                   |
| <=  | Inférieur ou égal | $\$x <= \$y$  | Renvoi true si $\$x$ inférieur ou égal a $\$y$                       |

TABLE 3.4 – Les opérateurs / Décrément d’incrément

|       | Nom            | Description                                     |
|-------|----------------|---|
| ++\$x | Pre-incrément  | Incrémente \$x par un et return le résultat \$x |
| \$x++ | Post-incrément | Renvoi \$x, ensuite incrémente \$x par un       |
| -\$x  | Pre-décrément  | Décrémente \$x par, et retourne \$x             |
| \$x-- | Post-décrément | Retourne \$x, ensuite décrémente \$x par un     |

TABLE 3.5 – Les opérateurs logiques

|     | Nom | Exemple     | Résultat                                     |
|-----|-----|-------------|--|
| and | Et  | \$x and \$y | True si \$x et \$y sont vrais                |
| or  | Ou  | \$x or \$y  | True si \$x ou \$y est vrai                  |
| xor | Xor | \$x xor \$y | True si \$x ou \$y est vrai et l’autre False |
| &&  | Et  | \$x && \$y  | True si \$x et \$y sont vrais                |
|     | Or  | \$x    \$y  | True si \$x ou \$y est vrai                  |
| !   | Not | !\$x        | True si \$x est False                        |

TABLE 3.6 – Les opérateurs de chaînes de caractères

|    | Nom                          | Exemple          | Résultat                          |
|----|------------------------------|------------------|-----------------------------------|
| .  | Concaténation                | \$txt1 . \$txt2  | Concaténation de \$txt1 et \$txt2 |
| .= | Concaténation et affectation | \$txt1 .= \$txt2 | Jointure de \$txt2 à \$txt1       |

## 3.4 Structures de contrôles

### 3.4.1 Les conditions

Les expressions conditionnelles sont utilisées pour effectuer des actions à la base des conditions. Les expressions qui permettent de déclarer des conditions en PHP sont :

- **if**, exécution d’un code si la condition est vraie.
- **if...else**, exécution d’un code si une condition est vraie et un autre code si elle est fausse
- **if...elseif....else**, le elseif est utilisé pour spécifier une condition si la première est fausse
- **switch**, pour sélectionner un code à exécuter parmi plusieurs

**l’expression if**

```

1 | if (condition) {
2 |     /*le code á executer*/;
3 | }

```

### Exemple :

si l'heure est inférieure à 20h, un message s'affichera sur le navigateur

```

1 | <?php
2 | $t = date("H");
3 |
4 | if ($t < "20") {
5 |     echo "Bonne nuit!";
6 | }
7 | ?>

```

### if...else

```

1 | if (condition) {
2 |     /* code á exécuter si la condition est vraie */
3 | } else {
4 |     /* le code á exécuter si la condition est fausse */
5 | }

```

### Exemple :

le code suivant affiche bonjour si l'heure est inférieure à 20h et bonsoir si elle est supérieure à 20h

```

1 | <?php
2 | $t = date("H");
3 |
4 | if ($t < "20") {
5 |     echo "bonjour";
6 | } else {
7 |     echo "bonne nuit";
8 | }
9 | ?>

```

### if...elseif...else

```

1 | if (condition1) {
2 |     /*le code á executer si la condition 1 est vraie*/
3 | } elseif (condition2) {
4 |     /*le code á exécuter si la condition 1 est fausse et la
   |     condition 2 est vraie*/

```

```

5 } else {
6   /*le code à exécuter si la condition 1 et la conditions 2 sont
   fausses */
7 }

```

### Exemple :

le code suivant affiche bonjour si l'heure est supérieure à 6 et inférieur à 15, bonsoir si l'heure est supérieure à 15h et inférieure ou égale à 20h et bonne nuit si l'heure est supérieure à 20.

```

1 <?php
2 $t = date("H");
3
4 if ($t < "6" and $t > "15") {
5   echo "Bonjour!";
6 } elseif ($t < "15" and $t >= "20") {
7   echo "Bonsoir!";
8 } else {
9   echo "Bonne nuit!";
10 }
11 ?>

```

### Switch Statement

```

1 switch (n) {
2   case label1:
3     /* si n==label1*/
4     break;
5   case label2:
6     /* si n==label2*/
7     break;
8   case label3:
9     /* si n==label3*/
10    break;
11
12   default:
13     /*le code à exécuter si n est différent de tous les labels;*/
14 }

```

### Exemple :

Le code suivant affichera le message « la couleur choisie est rouge! »

```

1 <?php
2 $favcolor = "rouge";
3
4 switch ($favcolor) {

```



```

5 | case "rouge":
6 |     echo "la couleur choisie est rouge!";
7 |     break;
8 | case "bleu":
9 |     echo "la couleur choisie est bleu!";
10 |    break;
11 | case "vert":
12 |     echo "la couleur choisie est vert!";
13 |     break;
14 | default:
15 |     echo "aucune couleur n'a été choisie ";
16 | }
17 | ?>

```

### 3.4.2 Les boucles

Souvent, lorsque nous écrivons un code, nous voulons qu'un bloque de code s'exécute plusieurs fois. Au lieu de répéter les lignes de code que nous voulons répéter, nous utiliserons les boucles.

En PHP, il existe plusieurs façons de déclarer une boucle :

- **while**, Une boucle qui se répète jusqu'à ce que la condition spécifiée soit fausse.
- **do...while**, Une boucle qui s'exécute une fois, ensuite elle se répète jusqu'à ce que la condition soit fausse.
- **for**, Une boucle qui s'exécute un nombre de fois.
- **foreach**, Une boucle qui parcourt tous les éléments d'un tableau.

#### While

```

1 | while (condition is true) {
2 |     code to be executed;
3 | }

```

#### Exemple :

Affecter la value 1 à \$x, et répéter la boucle 5 fois (\$x <= 5).

```

1 | <?php
2 | $x = 1;
3 |
4 | while($x <= 5) {
5 |     echo "The number is: $x <br>";
6 |     $x++;
7 | }
8 | ?>

```

## Do ... while

```
1 do {  
2     /* code to be executed; */  
3 } while (condition is true);
```

### Exemple :

```
1 <?php  
2 $x = 1;  
3  
4 do {  
5     echo "The number is: $x <br>";  
6     $x++;  
7 } while ($x <= 5);  
8 ?>
```

## For

```
1 for (init_counter; test_counter; increment_counter) {  
2     /* code to be executed; */  
3 }
```

### Paramètres :

- initcounter, initialise le compteur de la boucle
- testcounter, Evaluer la condition d'arrêt de la boucle.
- incrementcounter, incrémentation/décrémentation du compteur.

### Exemple :

Affiche les valeurs de 0 jusqu'à 10.

```
1 <?php  
2 for ($x = 0; $x <= 10; $x++) {  
3     echo "The number is: $x <br>";  
4 }  
5 ?>
```

## Foreach

```
1 foreach ($array as $value) {  
2     code to be executed;  
3 }
```

### Example :

Affichage du contenu d'un tableau.

```
1|<?php
2|$colors = array("red", "green", "blue", "yellow");
3|
4|foreach ($colors as $value) {
5|    echo "$value <br>";
6|}
7|?>
```

## 3.5 Les fonctions

Une fonction est un bloque de code qui peut être utilisé plusieurs fois dans un programme. Le langage PHP présente la possibilité de déclarer des fonctions sur une page web. Par contre une fonction ne sera exécutée que lorsqu'on lui fait appel (et non pas lors de l'affichage de la page). La syntaxe suivante est utilisée pour déclarer une fonction en PHP :

```
1|function nom_function() {
2|    /*code à executer;*/
3|}
```

Des informations peuvent être envoyées à une fonction à travers les arguments qui sont déclarés entre les parenthèses et séparés par des virgules.

```
1|<?php
2|function familyName($fname, $year) {
3|    echo $fname. "Born in ". $year. " <br>";
4|}
5|
6|familyName("Hege", "1975");
7|familyName("Stale", "1978");
8|familyName("Kai Jim", "1983");
9|?>
```

Les fonctions en PHP peuvent avoir une valeur de retour qui sera renvoyée en utilisant le mot clé return.

```
1|<?php
2|function sum($x, $y) {
3|    $z = $x + $y;
4|    return $z;
5|}
6|
7|echo "5 + 10 = " . sum(5, 10) . " <br>";
8|echo "7 + 13 = " . sum(7, 13) . " <br>";
```

```

9 | echo "2 + 4 = " . sum(2, 4);
10| ?>

```

## 3.6 Les tableaux

Un tableau stocke plusieurs valeurs en une seule variable, il permet de stocker plusieurs valeurs à la fois.

### Exemple

Soit une liste d'éléments (une liste des noms de voitures), le stockage des voitures dans des variables individuelles sera comme suit :

```

1 | $cars1 = "Volvo";
2 | $cars2 = "BMW";
3 | $cars3 = "Toyota";

```

En PHP :

```

1 | <?php
2 | $cars = array("Volvo", "BMW", "Toyota");
3 | echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2]
   | . ".";
4 | ?>

```

### 3.6.1 La création des tableaux en PHP

En PHP, la fonction `array()` est utilisée pour créer un tableau `"array()"`. Il existe trois types de tableaux en PHP :

- **Tableaux indexés**, tableaux avec un numéro d'indexe.
- **Tableaux associatifs**, tableaux avec des clés nommées.
- **Tableaux Multidimensionnels**, tableaux contenant un ou plusieurs tableaux.

### 3.6.2 Tableaux indexés en PHP

Il existe deux façons de créer un tableau indexé :

1. L'index est affecté automatiquement (l'indexe commence toujours par 0).

```

1 | $cars = array("Volvo", "BMW", "Toyota");

```

2. Ou il peut être affecté manuellement.

```

1 | $cars[0] = "Volvo";
2 | $cars[1] = "BMW";
3 | $cars[2] = "Toyota";

```

On utilise l'indexe pour récupérer les éléments du tableau :

```

1 | <?php
2 | $cars = array("Volvo", "BMW", "Toyota");
3 | echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2]
   | . ".";
4 | ?>

```

Dans cet exemple, nous avons créé un tableau indexé nommé \$cars, nous avons affecté trois éléments à ce tableau, et nous avons imprimé un texte qui contient les valeurs du tableau.

### La fonction count()

La fonction count() permet de retourner la longueur ( nombre d'éléments) du tableau :

```

1 | <?php
2 | $cars = array("Volvo", "BMW", "Toyota");
3 | echo count($cars); // le résultat sera 3
4 | ?>

```

### L'utilisation des boucles dans un tableau indexé

Pour afficher toutes les valeurs d'un tableau indexé, on peut utiliser la boucle **for** comme suit :

```

1 | <?php
2 | $cars = array("Volvo", "BMW", "Toyota");
3 | $arlength = count($cars);
4 |
5 | for($x = 0; $x < $arlength; $x++) {
6 |     echo $cars[$x];
7 |     echo "<br>";
8 | }
9 | ?>

```

### 3.6.3 Les tableaux associatifs

Les tableaux associatifs utilisent des clés nommées que vous devez affecter. Il existe deux façons de création d'un tableau associatif :

```
1 | $age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
```

Ou :

```
1 | $age['Peter'] = "35";
2 | $age['Ben'] = "37";
3 | $age['Joe'] = "43";
```

Dans un tableau associatif, on utilise des clés pour indexer les valeurs du tableau. Dans l'exemple, Peter est la clé de la valeur 35. Donc l'affichage se fait de la manière suivante :

```
1 | <?php
2 | $age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
3 | echo "Peter is " . $age['Peter'] . " yearsold.";
4 | ?>
```

### L'utilisation des boucles dans un tableau associatif

Pour afficher toutes les valeurs d'un tableau associatif, il faut utiliser la boucle **Foreach** :

```
1 | <?php
2 | $age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
3 |
4 | foreach($age as $x => $x_value) {
5 |     echo "Key=" . $x . " , Value=" . $x_value;
6 |     echo "<br>";
7 | }
8 | ?>
```

### 3.6.4 Tableaux multidimensionnels en PHP

Un tableau multidimensionnel est un tableau qui contient un ou plusieurs tableaux.

**Exemple :** Considérons Le tableau suivant :

| Nom        | Quantité disponible | Quantité vendu |
|------------|---------------------|----------------|
| Volvo      | 22                  | 18             |
| BMW        | 15                  | 13             |
| Saab       | 5                   | 2              |
| Land Rover | 17                  | 15             |

Vous pouvez stocker les données du tableau ci-dessus dans un tableau de deux dimensions comme suit :

```

1 $cars = array
2 (
3 array( "Volvo",22,18) ,
4 array( "BMW",15,13) ,
5 array( "Saab",5,2) ,
6 array( "Land Rover",17,15)
7 );

```

Le tableau de cet exemple contient quatre tableaux, et possède deux indices : l'indice des lignes et l'indice des colonnes. Pour accéder aux éléments du tableau \$cars, il faut pointer sur les deux indices (celui des lignes et celui des colonnes) :

```

1 <?php
2 echo $cars[0][0]. " : In stock: " . $cars[0][1]. " , sold: " . $cars
   [0][2]. "<br>";
3 echo $cars[1][0]. " : In stock: " . $cars[1][1]. " , sold: " . $cars
   [1][2]. "<br>";
4 echo $cars[2][0]. " : In stock: " . $cars[2][1]. " , sold: " . $cars
   [2][2]. "<br>";
5 echo $cars[3][0]. " : In stock: " . $cars[3][1]. " , sold: " . $cars
   [3][2]. "<br>";
6 ?>

```

Pour afficher les valeurs de ce tableau, il faut utiliser deux boucles **for** (vous devez toujours pointer sur les deux indices) :

```

1 <?php
2 for ($row = 0; $row < 4; $row++) {
3     echo "<p><b>Row number $row</b></p>";
4     echo "<ul>";
5     for ($col = 0; $col < 3; $col++) {
6         echo "<li>" . $cars[$row][$col]. "</li>";
7     }
8     echo "</ul>";
9 }
10 ?>

```

### 3.7 Classes et objets

PHP est un langage de programmation orienté objet, ce qui signifie que vous pouvez créer des objets qui peuvent contenir des variables et des fonctions.

Lorsque nous parlons d'objets, nous faisons référence aux variables appartenant à ces objets en tant qu'attributs, et les fonctions sont appelées méthodes.

Il est possible de regrouper les fonctions et les données en un seul endroit. La créations des objets peut se faire en utilisant des classes (constructeurs d'objets), ce qui permettra la création de beaucoup d'instances (objets, qui ont été construits via une classe).

**Exemple :**

```
1 <?php
2 // The code ci-dessous crée une classe
3 class Personne {
4     // Creating some properties (variables tied to an object)
5     public $estVivante = true;
6     public $nom;
7     public $prenom;
8     public $age;
9
10    // affectation des valeurs dans le constructeur
11    public function __construct($nom, $prenom, $age) {
12        $this->nom = $nom;
13        $this->prenom = $prenom;
14        $this->age = $age;
15    }
16
17    // Création d'une méthode pour afficher les valeurs des
18    // attributs
19    public function afficher() {
20        return "Bonjour, mon nom est " . $this->nom . " " . $this->
21        prenom . " ";
22    }
23
24    // Création d'un nouvel objet de type personne
25    $a = new Personne('BEGHOURA', 'Mohamed Amine', 30);
26
27    // l'utilisation de la méthode d'affichage
28    echo $a->afficher();
29 ?>
```

Dans le code de cet exemple, nous avons créé une classe Personne et une instance (objet) stockées dans \$a sur la ligne 24. Puis la méthode afficher() de l'objet \$moi est appelée sur la ligne 27.

## 3.8 Caractéristiques

### 3.8.1 Gestion des erreurs

Nous utilisons principalement trois méthodes pour gérer les erreurs en PHP :

- La fonction die() qui va stopper l'exécution du script en cas d'erreur ;
- La création des gestionnaires d'erreurs personnalisés à l'aide des fonctions PHP de type error ;



— L'utilisation des outils de rapport d'erreurs natifs en PHP.

**Exemple :** l'utilisation de la fonction die().

```
1 <?php
2     $x = 0;
3
4     if($x == 0){
5         die('X est égal á zéro !!!');
6     }
7
8     echo "Ce message ne s'affichera que si x est différent de 0";
9 ?>
```

### 3.8.2 Récupération des données d'un formulaire

Les superglobales PHP \$\_GET et \$\_POST sont utilisés pour collecter des données de formulaire. Considérons le formulaire suivant :

```
1 <html>
2 <body>
3     <form action="afficher.php" method="post">
4         Nom: <input type="text" name="nom"><br>
5         E-mail: <input type="text" name="email"><br>
6         <input type="submit" value="Envoyer">
7     </form>
8 </body>
9 </html>
```

Lorsque l'utilisateur remplit le formulaire ci-dessus et clique sur le bouton Envoyer, les données du formulaire sont envoyées pour traitement dans un fichier PHP nommé "afficher.php". Les données de formulaire sont envoyées avec la méthode HTTP POST.

Pour afficher les données introduite par l'utilisateur dans ce formulaire, le script du fichier afficher.php doit ressembler à celui-ci :

```
1 <html>
2 <body>
3
4 Bienvenue <?php echo $_POST["nom"]; ?><br>
5 Votre adresse email est: <?php echo $_POST["email"]; ?>
6
7 </body>
8 </html>
```

**Note :** Pour utiliser la méthode GET, il faut remplacer la valeur de l'attribut method de la balise form par "get" et utiliser la superglobale \$\_GET pour récupérer les données.

### 3.8.3 Connexions persistantes aux Bases de Données.

Le PHP permet le stockage permanent des données dans des bases de données relationnelles. Dans cette partie, il y a les opérations essentielles pour manipuler une base de données MySQL.

#### Connexion à une base de données MySQL

Pour pouvoir se connecter à une base de données, il est nécessaire de connaître l'adresse du serveur (**\$servername**) où cette base est hébergée, le nom (**\$username**) et le mot de passe de l'utilisateur (**\$password**).

```
1 <?php
2 $servername = "localhost";
3 $username = "username";
4 $password = "password";
5
6 // Création d'une connection
7 $conn = mysqli_connect($servername , $username , $password);
8
9 // Vérification de la connection
10 if (!$conn) {
11     die("Connection failed: " . mysqli_connect_error());
12 }
13 echo "Connected successfully";
14
15 mysqli_close($conn);
16 ?>
```

La fonction `mysqli_connect(...)` permet d'ouvrir une connexion avec le SGBD MySQL.

`mysqli_close($conn)` ferme la connexion qui a été créée.

#### L'insertion des enregistrement dans une table

Supposons qu'il y a une base de données nommées "**myDB**" qui contient une table **myGuests** dont la structure est la suivante :

**myGuests**(id, nom, email)

la clé primaire est le champs **id** dont le type est entier et la valeur est gérée automatiquement par le SGBD où elle s'incrémente par un à chaque insertion d'enregistrement.

L'insertion d'un nouvel enregistrement, dont les données sont (**beghoura, amine.beg@gmail.com**), dans la table **myGuests** se fait de la manière suivante :

```

1 <?php
2     $servername = "localhost";
3     $username = "username";
4     $password = "password";
5     $dbname = "myDB";
6
7     // Création et sélection d'une base de données
8     $conn = mysqli_connect($servername, $username, $password,
9         $dbname);
10
11     $sql = "INSERT INTO MyGuests (nom, email)
12     VALUES ( 'beghoura ', 'amine.beg@gmail.com' ) ";
13
14     if (mysqli_query($conn, $sql)) {
15         $last_id = mysqli_insert_id($conn);
16         echo "Insertion complète. la valeur du champs id attribuée à
17         l'enregistrement est : " . $last_id;
18     }
19     mysqli_close($conn);
20 ?>

```

La fonction `mysqli_query($conn, $sql)` permet d'exécuter la requête SQL stockée dans la variable `$sql`.

La fonction `mysqli_insert_id($conn)` récupère la valeur du dernier id attribuée à l'enregistrement.

## La suppression d'un enregistrement

Pour supprimer un enregistrement de la table, il est nécessaire d'utiliser la requête DELETE. l'implémentation du code de la suppression est :

```

1 <?php
2     $servername = "localhost";
3     $username = "username";
4     $password = "password";
5     $dbname = "myDB";
6
7     $conn = mysqli_connect($servername, $username, $password,
8         $dbname);
9
10    // la requete de suppression
11    $sql = "DELETE FROM MyGuests WHERE id=3";
12
13    if (mysqli_query($conn, $sql)) {
14        echo "Suppression effectuée avec succès";
15    } else {
16        echo "Erreur : " . mysqli_error($conn);
17    }
18    mysqli_close($conn);

```

19| ?>

l'exécution du code supprimera l'enregistrement dont l'identifiant `id = 3` (spécifié dans la requête SQL : `DELETE FROM MyGuests WHERE id=3`).

## Mettre à jour un enregistrement

La modification d'un enregistrement en PHP nécessite l'utilisation de la requête **UPDATE**.

```
1 <?php
2 $servername = "localhost";
3 $username = "username";
4 $password = "password";
5 $dbname = "myDB";
6
7 $conn = mysqli_connect($servername, $username, $password,
8     $dbname);
9
10 // la requete de modification
11 $sql = "UPDATE MyGuests SET nom='amine' WHERE id=2";
12
13 if (mysqli_query($conn, $sql)) {
14     echo "Modification effectuée avec succès";
15 } else {
16     echo "Erreur : " . mysqli_error($conn);
17 }
18
19 mysqli_close($conn);
20 ?>
```

Le code écrase la valeur du champs nom de l'enregistrement dont la valeur est 2 et la remplace par la valeur "amine" (`UPDATE MyGuests SET nom='amine' WHERE id=2`).

## La récupération des données

La récupération des enregistrement se fait par la requête **SELECT**.

```
1 <?php
2 $servername = "localhost";
3 $username = "username";
4 $password = "password";
5 $dbname = "myDB";
6
7 $conn = mysqli_connect($servername, $username, $password,
8     $dbname);
9
10 // la requete de sélection
```

```

10 $sql = "SELECT * FROM MyGuests";
11
12 $result = mysqli_query($conn, $sql);
13
14 if (mysqli_num_rows($result) > 0) {
15     // Afficher le résultat
16     while($row = mysqli_fetch_assoc($result)) {
17         echo "id: " . $row["id"]. " - Nom : " . $row["nom"]. "<br>";
18     }
19 } else {
20     echo "0 results";
21 }
22
23 mysqli_close($conn);
24 ?>

```

la fonction `mysqli_num_rows()` vérifie s'il y a plus de zéro lignes retournées.

la fonction `mysqli_fetch_assoc()` place tous les résultats dans un tableau associatif que l'on peut parcourir. La boucle `while()` est utilisé pour parcourir le tableau des résultats.

### 3.8.4 Gestion des sessions

Une session est une façon de stocker des informations (dans les variables) pour être utilisé sur plusieurs pages. Les informations sont stockées sur le serveur (Contrairement à ce qu'on appelle des cookies où l'information est stockée sur l'ordinateur des utilisateurs.).

#### Démarrage et utilisation d'une session PHP

Une session est lancée avec la fonction `session_start()`. Les variables de session sont définies avec la variable globale PHP : `$_SESSION`.

La fonction `session_start()` doit être utilisée en premier dans le document avant toutes les balises HTML :

```

1 <?php
2     // Démarrage de la session
3     session_start();
4 ?>
5
6 <!DOCTYPE html>
7 <html>
8 <head></head>
9 <body>
10
11 <?php
12     // le stockage des valeurs

```

```

13     $_SESSION["favcolor"] = "green";
14     $_SESSION["favanimal"] = "cat";
15
16     //
17     ?>
18
19 </body>
20 </html>

```

## Récupération et modification d'une variable de session PHP

Toutes les valeurs de variables de session sont stockés dans la variable globale `$_SESSION`. Pour modifier une variable de session, il suffit juste d'écraser ses valeurs.

```

1 <? php
2     session_start ();
3 ?>
4 <! DOCTYPE html>
5 <html>
6 <body>
7 <? php
8     // pour récupérer une variable de session , on utilise le nom
      qu'on lui a attribué
9     echo $_SESSION["favcolor"]; // cette ligne doit afficher la
      valeur green.
10
11     // Pour changer une variable de session , à l'écraser
12     $_SESSION["favcolor"] = "yellow";
13     echo $_SESSION["favcolor"]; // cette ligne doit afficher la
      valeur yellow.
14 ?>
15 </ body>
16 </ html>

```

## Détruire une session PHP

Pour supprimer toutes les variables globales de session et détruire la session, on utilise les fonctions `session_unset()` et `session_destroy()` :

```

1 <?php
2     session_start ();
3 ?>
4 <!DOCTYPE html>
5 <html>
6 <body>
7
8 <?php
9     // supprimer toutes les variables

```

```
10| session_unset();  
11|  
12| // détruire la session  
13| session_destroy();  
14| ?>  
15|  
16| </body>  
17| </html>
```

## 4

# Services Web : notions de base

## 4.1 Introduction

Un service web est une technologie permettant à des applications de dialoguer à distance via Internet, et ceci indépendamment des plates-formes et des langages sur lesquelles elles reposent. Pour ce faire, les services Web s'appuient sur un ensemble de protocoles Internet très répandus (HTTP), afin de communiquer. Cette communication est basée sur le principe de demandes et réponses, effectuées avec des messages XML.

Les service web ont été mises en places afin de répondre à un certains nombre de besoins :

- Remplacer les protocoles actuels (RPC, DCOM, RMI) par une approche entièrement ouverte et interopérable, basée sur la généralisation des serveurs Web avec scripts CGI
- Faire interagir des composants hétérogènes, distants, et indépendants avec un protocole standard (SOAP) Simplifier la communication entre ces composants
- Ne pas créer de nouvelles technologies, mais se baser sur celles qui existent déjà (XML, HTTP)

## 4.2 Architecture orientée services (SOA)

les Web Services sont des "applications modulaires basées sur Internet qui exécutent des tâches précises et qui respectent un format spécifique". Ce sont donc des unités logiques applicatives qui sont accessibles grâce au protocole Internet.

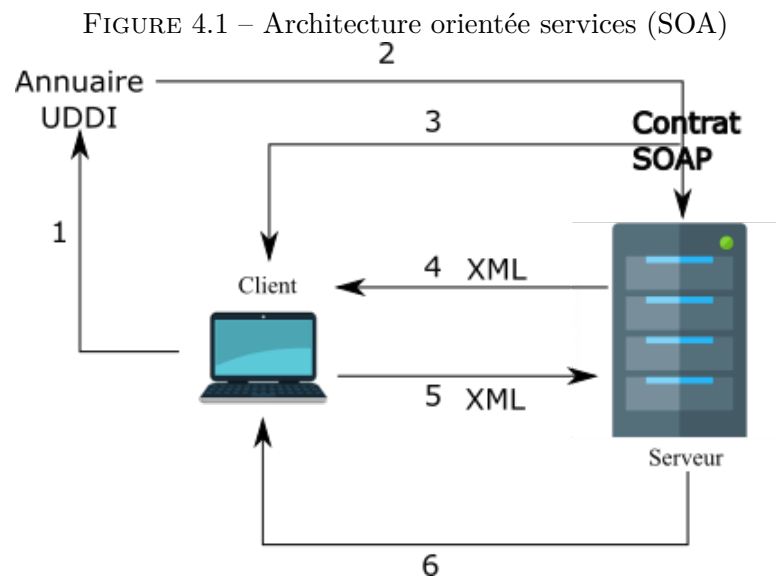
les services Web ne sont que des applications modulaires qui peuvent être présentées, publiées, situées et invoquées dans un réseau. Ainsi, les applications peuvent faire appel à des fonctionnalités situées sur d'autres machines



dans d'autres applications.

Les trois éléments les plus importants des services Web sont les fournisseurs de service, les annuaires de services et les consommateurs de service :

- Le fournisseur (ou serveur) crée le service Web et publie toutes ces caractéristiques dans l'annuaire de service.
- L'annuaire rend disponible les interfaces d'accès au service et donnant le contrat et l'architecture employée pour permettre les interactions.
- Le consommateur (ou client) quant à lui, accède à l'annuaire pour rechercher le service Web dont il a besoin et avec lui les normalisation à obtenir. Il peut ainsi envoyer ses requêtes au service désiré et obtenir les réponses qu'il pourra analyser.



- Le client envoie une requête à l'annuaire de Service pour trouver le service Web dont il a besoin.
- L'annuaire cherche pour le client, trouve le service Web approprié et renvoie une réponse au client en lui indiquant quel serveur détient ce qu'il recherche.
- Le client envoie une deuxième requête au serveur pour obtenir le contrat de normalisation de ses données.
- Le serveur envoie sa réponse sous la forme établie par WSDL en langage XML.
- Le client peut maintenant rédiger sa requête pour traiter les données dont il a besoin.
- Le serveur fait les calculs nécessaires suite à la requête du client, et renvoie sa réponse sous la même forme normalisée.

## 4.3 Standards de base pour les services Web

### 4.3.1 SOAP

SOAP (Object Access Protocol) est un protocole de RPC (Remote Procedure Call) permettant d'invoquer des méthodes d'objets distants. Il utilise XML pour définir les fonctions et les définitions disponibles. Il prend en charge divers protocoles de transport, tels que HTTP et SMTP, ainsi que différents formats comme MIME.

SOAP étant un protocole d'échange d'informations entre diverses machines sur un réseau, il nécessite un format pour transporter les données. Pour cela elle utilise des messages SOAP qui sont en fait des documents XML.

### 4.3.2 WSDL

WSDL (Web Services Description Language : langage de description des services Web). C'est un fichier qui spécifie ce que doit contenir un message de requête et l'apparence du message de réponse dans une notation sans ambiguïté.

La notation utilisée par un fichier WSDL pour décrire les formats de messages est basé sur la norme du schéma XML, ce qui signifie que WSDL est à la fois neutre par rapport au langage de programmation et à la plateforme.

Outre la description du contenu des messages, WSDL définit l'endroit où le service est disponible et le protocole de communications utilisé pour converser avec le service. Cela signifie que le fichier WSDL définit tout ce qui est nécessaire pour écrire un programme fonctionnant avec un service Web.

En résumé WSDL c'est un contrat entre un client et un serveur qui fait état :

- des spécifications d'interfaces qui décrivent toutes les méthodes publiques,
- des spécifications relatives aux types de donnée de messages mis en oeuvre dans les questions-réponses,
- des informations liées au protocole de transport utilisé,
- des informations d'adresse permettant de localiser le service décrit.

### 4.3.3 UDDI

l'UDDI est l'équivalent des pages jaunes dans les services Web. Tout comme avec les pages jaunes classiques, on peut y rechercher une entreprise offrant les services dont on a besoin et consulter le service offert .

Bien entendu, on peut proposer un service Web sans l'inscrire, mais si on souhaite toucher un public plus important, UDDI permet aux clients éventuels de trouver vos services.

Une entrée du répertoire UDDI est constituée d'un fichier XML qui décrit une entreprise et les services qu'elle offre. Chaque entrée du répertoire UDDI est constituée de trois parties.

Les "pages blanches" décrivent l'entreprise qui offre le service : nom, adresse, contacts, etc. Les "pages jaunes" comportent les catégories industrielles. Les "pages vertes" décrivent l'interface vers le service avec suffisamment de détail pour qu'il soit possible d'écrire une application permettant d'utiliser le service Web.

#### 4.3.4 Une application web sous forme de service web

Le service web de l'exemple est programmé en PHP (coté fournisseur), javascript (coté consommateur) et JSON pour la représentation et transfert des données. Ce service présente les opérations nécessaires pour la manipulation de une table nommée participants (id, name, email).

##### Coté fournisseur :

Le code suivant fournit les fonctions nécessaires à la manipulation de la table participants. Les opérations sont déployées en fonction de la méthode utilisée dans la requête HTTP (POST, GET, PUT, DELETE).

##### participants.html

```
1 <?php
2 header("Access-Control-Allow-Origin: *");
3 header("Content-Type: application/json; charset=UTF-8");
4 header("Access-Control-Allow-Methods: GET, POST, PUT, DELETE,
5     OPTIONS");
6 header("Access-Control-Max-Age: 3600");
7 header("Access-Control-Allow-Headers: Content-Type, Access-
8     Control-Allow-Headers, Authorization, X-Requested-With");
9
10 $conn = mysqli_connect("localhost", "root", "", "servicesweb");
11
12 if($_SERVER['REQUEST_METHOD'] == "POST"){
13
14     $data = json_decode(file_get_contents("php://input"));
15
16     $name = isset($data->name) ? mysqli_real_escape_string($conn,
17         $data->name) : "";
18     $email = isset($data->email) ? mysqli_real_escape_string($conn,
19         $data->email) : "";
20
21     $sql = "INSERT INTO participants (name, email) VALUE ('".$name
22         ."', '".$email."')";
23
24     $result = mysqli_query($conn, $sql);
```

```

23     if($result){
24         $json = array("status" => 1, "msg" => "student added");
25     }else{
26         $json = array("status" => 0, "msg" => "error adding the
27             student");
28     }
29 }elseif($_SERVER['REQUEST_METHOD'] == "DELETE"){
30
31     $ID = isset($_GET['id_participant']) ?
32         mysqli_real_escape_string($conn, $_GET['id_participant']) :
33         "";
34
35     if(!empty($ID)){
36         $result = mysqli_query($conn, "DELETE from participants
37             where id_participant='". $ID. "'");
38
39         if($result){
40             $json = array("status" => 1, "msg" => "student deleted");
41         }else{
42             $json = array("status" => 0, "msg" => "error deleting the
43                 student");
44         }
45     }elseif($_SERVER['REQUEST_METHOD'] == "PUT"){
46
47         $data = json_decode(file_get_contents("php://input"));
48
49         $ID = isset($_GET['id_participant']) ?
50             mysqli_real_escape_string($conn, $_GET['id_participant']) :
51             "";
52
53         $name = isset($data->name) ? mysqli_real_escape_string($conn,
54             $data->name) : "";
55         $email = isset($data->email) ? mysqli_real_escape_string($conn
56             , $data->email) : "";
57
58         if(!empty($ID)){
59             $result = mysqli_query($conn, "UPDATE participants SET name
60                 ='". $name. "', email = '". $email. "' WHERE id_participant='".
61                 $ID. "'");
62
63             if($result){
64                 $json = array("status" => 1, "msg" => "update succesful");
65             }else{
66                 $json = array("status" => 0, "msg" => "error updating");
67             }
68         }else{
69             $json = array("status" => 0, "msg" => "set the id");
70         }
71     }

```

```

66
67 } elseif($_SERVER['REQUEST_METHOD'] == "GET"){
68
69     $ID = isset($_GET["id_participant"]) ?
        mysqli_real_escape_string($conn, $_GET["id_participant"]) :
        "";
70
71     if(!empty($ID)){
72         $results = mysqli_query($conn, "select * from participants
        where id_participant='". $ID . "'");
73         $countRes = mysqli_query($conn, "select COUNT(*) from
        participants where id_participant='". $ID . "'");
74     } else {
75         $results = mysqli_query($conn, "select * from participants")
        ;
76         $countRes = mysqli_query($conn, "select COUNT(*) from
        participants");
77     }
78     $count = mysqli_fetch_assoc($countRes);
79
80     $records = array();
81     while($row = mysqli_fetch_assoc($results)){
82         extract($row);
83         $records[] = array("id_participant" => $id_participant, "
        name" => $name, "email" => $email);
84     }
85
86
87     $json = array("status" => 1, "Count" => $count["COUNT(*)"], "
        records" => $records);
88
89 } else {
90     $json = array("status" => 0, "msg" => "Method not accepted");
91 }
92
93 mysqli_close($conn);
94
95 echo json_encode($json);

```

### Coté consommateur :

Pour le coté consommateur, nous avons un code qui affiche le contenu de la table ainsi qu'un formulaire pour l'ajout de nouvel enregistrement dans la table participants.

#### index.html

```

1 <html>
2 <head>
3   <meta charset="utf-8"/>
4 </head>
5 <body>

```

```

6   <ul id="myList"></ul>
7 </body>
8
9 <script src="jquery-3.2.1.min.js"></script>
10 <script>
11     $( document ).ready(function() {
12         $.ajax({
13             crossOrigin: true,
14             type: 'get',
15             url: 'http://localhost/vacances/api/participants.php',
16             success: function (data,status,xhr) {
17                 $.each( data.records, function( key, value ) {
18                     $("#myList").append("<li>"+value.name+ " " + value.email
19 + " " + "<button class='remove-btn' data-id='"+value.
20 id_participant+"' href='#'> Supprimer</button>"+ "</li>");
21                 });
22             }
23         });
24
25         $(document).on('click', '.remove-btn',function(){
26             var id = $(this).attr('data-id');
27             $.ajax({
28                 url: "http://localhost/vacances/api/participants.php?
29 id_participant="+id,
30                 type: "DELETE",
31                 crossDomain: true,
32                 dataType: "json",
33                 contentType: "application/json; charset=utf-8"
34             });
35         });
36     });
37 </script>
38 </html>

```

### form.html

```

1 <html>
2 <head>
3   <meta charset="utf-8" />
4 </head>
5 <body>
6   <h1>Hello World</h1>
7   <form id="my-form">
8     <input type="text" id="name">
9     <input type="text" id="email">
10
11     <input type="submit" value="sauvegarder">
12   </form>
13 </body>
14

```

```

15 <script src="jquery-3.2.1.min.js"></script>
16 <script>
17     $( document ).ready(function() {
18         $('#my-form').submit( save );
19
20         function save(e){
21             var data = { "name" : $("#name").val() ,
22                         "email" : $("#email").val() };
23
24             $.ajax({
25                 type: 'POST',
26                 url: 'http://localhost/vacances/api/participants.php',
27                 data: JSON.stringify(data),
28                 success: function (data) {
29                     }
30             });
31
32             $("#my-form")[0].reset();
33             e.preventDefault();
34         }
35     });
36
37 });
38
39 </script>
40 </html>

```

**5**

## **Exercices**



# 1

## Codage en HTML

- Sur le serveur web, créer un répertoire nommé "fables" dans le quel vous créez une page web "**index.html**" dont le code HTML est le suivant :

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Les fables</title>
5 </head>
6 <body>
7   <h1><u>Les Fables de la Fontaine</u></h1>
8   <ul>
9     <li><a href="#">Livre I</a></li>
10    <li><a href="#">Livre II</a></li>
11    <li><a href="#">Livre III</a></li>
12  </ul>
13  
14 </body>
15 </html>
```

- Sur le navigateur web, taper l'URL suivant : `http://localhost/fables`
- Pour chaque lien dans la page **index.html**, créer des pages web (**livre1.html**, **livre2.html**, **livre3.html**).
- Dans chaque page web, mettre un titre, une image et un paragraphe.
- Remplacer les # dans la balise <a> par : **livre1.html**, **livre2.html** et **livre3.html**

## 2

# Manipulation des tableaux en HTML

Reproduire le code HTML qui affiche le même contenu sur la figure en utilisant les balises HTML suivantes :

<!DOCTYPE html>, <html>,<head>,<title>, <meta>, <body>,<h2>, <p>, <hr>, <table>, <tr>, et <td>

### HTML Tables

HTML tables start with a table tag.

Table rows start with a tr tag.

Table data start with a td tag.

---

#### 1 Column:

100

---

#### 1 Row and 3 Columns:

100 200 300

---

#### 3 Rows and 3 Columns:

100 200 300  
400 500 600  
700 800 900

---

### 3

## Création des formulaires

Reproduiser le code HTML qui affiche le formulaire suivant :

|   |   |
|---|---|
| Nom:  | <input type="text"/>                            |
| Prénom:                                     | <input type="text"/>                            |
| Pseudo:                                     | <input type="text"/>                            |
| e-mail:                                     | <input type="text"/>                            |
| Confirmation:                               | <input type="text"/>                            |
| Mot de passe:                               | <input type="text"/>                            |
| Confirmation Mot de passe:                  | <input type="text"/>                            |
| Région (Bordj Bou Arréridj, Sétif, M'sila): | <input type="text" value="Bordj Bou Arréridj"/> |

---

*Sélectionner les options que vous désirez*

- ☐ Option 1
- ☐ Option 2
- ☐ Option 3

---

*Sélectionner un seul choix*

- ☐ Option 1
- ☐ Option 2
- ☐ Option 3
- ☒ Autre

## 4

# HTML / CSS

Reproduire le code HTML/CSS suivant sur machine :

```
1 <!doctype html>
2 <html>
3   <head>
4     <title> Page avec style </title>
5     <style type="text/css">
6       #entete, #menu, #contenu, #footer {
7         padding:1px 0;
8       }
9       #entete {
10        background-color:#FF9900;
11        text-align:center;
12      }
13      #main {
14        max-width:960px;
15        margin:auto;
16      }
17      #menu {
18        float:left;
19        width:240px;
20        background-color:#FF3366;
21      }
22      #contenu {
23        margin-left:245px;
24        background-color:#9966FF;
25      }
26      #footer {
27        background-color:#669933;
28        text-align:center;
29        clear:both;
30      }
31    </style>
32  </head>
33
34  <body>
35    <div id="entete"> <p>En tete</p> </div>
36    <div id="main">
37      <div id="menu"> <p>Menu</p> </div>
38
39      <div id="contenu"> <p>Contenu</p> </div>
40    </div>
41    <div id="footer"> <p>Pied de Page</p> </div>
42  </body>
43 </html>
```

## 5

### PHP (notions de base)

#### EXERCICE 1 :

Afficher dans une page la phrase « Ceci est une ligne créée uniquement en PHP ».

Afficher à la ligne suivante : « Ceci est la 2ème phrase créée avec PHP ».

#### EXERCICE 2 :

Déclarer 2 variables : **x** et **y**. initialiser Les avec les valeurs « Bordj » et « Bou Arreridj » et les afficher sur la page en utilisant 2 modes différents :

- 2 commandes echo.
- 1 commande echo avec 1 seule chaîne de caractère.

#### EXERCICE 3 :

Afficher un titre H1 : « Calcul sur les variables ».

Affecter respectivement les valeurs 0.206, 150 et 10 aux variables **TVA**, **prix** et **quantité**.

Calculer le **prix HT** et le **prix TTC** pour les 10 articles et afficher les.

#### EXERCICE 4 :

Affecter respectivement les valeurs 150, 50 et 10 aux variables **prix\_table**, **prix\_armoire** et **quantité**.

Calculer le **prix HT** total pour les 10 armoires.

Comparer le prix de l'armoire et de la table et afficher quel est le prix le plus élevé.

#### EXERCICE 5 :

Affecter une valeur à la variable **nbre** ensuite afficher la somme des entiers de 1 jusqu'à nbre.

**Note :** réaliser cet exercice avec l'instruction FOR puis avec l'instruction WHILE.

## 6

### PHP (boucles et tableaux)

#### EXERCICE 1 :

Écrivez un tableau multidimensionnel associatif dont les clés sont des noms de personne et les valeurs des tableaux indicés contenant le prénom, la ville de résidence et l'âge de la personne.

#### EXERCICE 2 :

Écrivez un tableau multidimensionnel associatif dont les clés sont des noms de personne et les valeurs des tableaux associatifs dont les clés sont le prénom, la ville de résidence et l'âge de la personne avec une série de valeurs associées.

#### EXERCICE 3 :

Utilisez une boucle foreach pour lire les tableaux des exercices 1 et 2.

#### EXERCICE 4 :

Créez un tableau contenant une liste d'adresses de sites recommandés, puis créez un lien aléatoire vers le premier site de la liste après avoir trié le tableau en ordre aléatoire.

La fonction **shuffle()** mélange effectivement les éléments d'un tableau mais ne conserve pas les clés, elle n'est donc pas adaptée pour récupérer la clé et la valeur du tableau \$tab.

la fonction **array\_rand()** retourne la clé de l'élément pris au hasard. Cette clé permet de lire le nom du site et son adresse URL.

#### EXERCICE 5 :

Créez un tableau d'entiers variant de 1 à 63, puis à partir de celui-ci un autre tableau de nombres variant de 0 à 6.3. Créez ensuite un tableau associatif dont les clés X varient de 0 à 6.3 et dont les valeurs sont  $\sin(X)$ . Affichez le tableau de valeurs dans un tableau HTML.

#### EXERCICE 6 :

Créez un tableau contenant une liste d'adresses e-mail. Extrayez le nom de serveur de ces données, puis réalisez des statistiques sur les occurrences de chaque fournisseur d'accès.