HTML 5

PARTIE 1 / PRESENTATION

Table des matières

1. Le contexte historique

1.1. HTML 1.0

L'histoire du Html (*HyperText Markup Language*) commence en 1990. Tim Berners-Lee, informaticien britannique au Centre d'Énergie et de la Recherche Nucléaire de Genève (CERN), veut mettre au point un système pour échanger des informations entre scientifiques par l'intermédiaire d'Internet. Les caractéristiques sont un formatage de texte simple, l'incorporation de graphiques et l'élément qui va s'avérer génial : la possibilité de faire des liens vers d'autres documents.

Le langage Html 1.0 est né. À cause de cet élément hypertexte, tout le projet est baptisé World Wide Web (réseau mondial ou plus communément appelé la toile).

Les éléments induits par ce contexte historique sont :

- À l'origine, les pages Web sont bien d'essence textuelle et bien éloignées de l'aspect visuel qu'on leur a apporté depuis.
- Le Html reprend ses termes de l'anglais qui est la langue la plus répandue dans les milieux scientifiques internationaux.
- À l'époque, le matériel électronique est rustique par rapport au matériel actuel ; disque dur et mémoire de petite capacité, le top de l'affichage est de 256 couleurs. Les lignes téléphoniques sont encore loin de l'ADSL ou du câble.
- Pour s'échanger des documents, on n'utilise aucun logiciel spécifique comme Word, WordPerfect... On a bien créé un langage spécifique et indépendant du système d'exploitation et ce, dans le plus pur esprit d'Internet.

Très rapidement (1992-1993), les premières applications pour visionner en ligne les documents Html (ou navigateurs), font leur apparition : Mosaic, d'abord et Netscape, ensuite. La porte du Web est enfin ouverte au grand public. C'est le "Big-bang" du Web.

Conscientes des énormes possibilités offertes par le langage Html et poussées par l'impatience des pionniers, ces firmes vont très vite créer leurs propres évolutions du code Html, mais de façon quelque peu confuse, voire anarchique.

1.2. HTML 2.0

Extensions apportées par les firmes, sans réelle vision pour de futurs développements. La poussée créatrice était bien trop forte et s'est donc poursuivie, surtout que Microsoft a pris le train du Web en marche et a apporté ses propres spécifications. En fait, dès sa parution, le Html 2.0 est déjà, dans la pratique, complètement dépassé.

Conscient de ces dérapages, le même Tim Berners-Lee crée en 1994, le W3C (pour consortium du WWW). Ce consortium reprend des délégués des principaux intervenants du Web et du monde de l'informatique comme par exemple IBM, Microsoft, America Online, Netscape, Apple, Adobe, Macromedia, Sun Microsystems, etc. Les objectifs du W3C sont, à cette époque (et encore aujourd'hui) de standardiser la publication sur le Web et de rendre le Web accessible à tous ses utilisateurs quelles que soient leurs différences de culture, d'éducation, de ressources et/ou de handicap physique.

Ces objectifs ambitieux n'ont trouvé que difficilement un consensus auprès des différents partenaires et la mise en route est laborieuse et peu fructueuse. On en tient pour preuve, une version Html 3.0 qui n'a jamais vu le jour.

1.3. HTML 3.2

l a fallu attendre janvier 1997, pour obtenir une première version réellement standard (avec le Html 3.2) qui a été reconnue et prise en charge par la majorité des navigateurs.

Entre-temps, l'autorité du W3C s'est affirmée et a donné le jour à toute une série de recommandations et de nouveaux langages qui sont maintenant acceptés de façon unanime. Au cours de cet ouvrage, le W3C sera souvent cité comme référence de la publication sur le Web. Le W3C a bien entendu son site www.w3.org qui reprend toutes les spécifications techniques.

1.4. HTML4.0

Date : décembre 1999. Avec la spécification Html 4.0 qui introduit de nouvelles commandes, officialise les feuilles de style et apporte ses recommandations en terme d'écriture, et le langage Html dispose enfin d'un standard qui fait l'unanimité.

NB : Il existe aussi une version supérieure (4.01) qui corrige les erreurs de la version 4.0.

1.5. HTML 5

Le HTML5 est devenue une recommandation officielle (*W3C Recommendation*) de la part du W3C le 28 octobre 2014. Voici l'URL du HTML5 dans le site du W3C : http://www.w3.org/TR/html5/

Le W3C (http://www.w3.org) est chargé d'édicter des **standards** pour le Web. Avec ces standards, vous êtes sûr d'avoir des développements pérennes et qui soient gages d'interopérabilité. C'est bien aux navigateurs d'adopter ces standards, afin de pouvoir créer des pages Web qui soient universelles, lisibles et affichées de manière identique par tous les navigateurs.

Actuellement, le HTML5 permet de créer des applications Web dynamiques et non plus de simples pages Web statiques. Le HTML5 n'est pas venu à nous seul, mais avec des API JavaScript performantes pour le graphisme, l'audio, la vidéo, le glisser-déposer, la géolocalisation et la communication.

2. Langage à balise

WIKIPEDIA ➔En informatique, les langages de balisage représentent une classe de langages spécialisés dans l'enrichissement d'information textuelle. Ils utilisent des balises, unités syntaxiques délimitant une séquence de caractères ou marquant une position précise 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 balises sont des **commandes** à l'intention du navigateur et saisies entre des signes inférieu r à (<) et supérieur à (>).

Ainsi une balise s'écrit `<balise>`.

En règle générale, à toute balise d'ouverture (ou balise de début) correspond une balise de fermeture (ou balise de fin). Cette balise de fermeture marque la fin de la commande annoncée par la balise d'ouverture. Elle reprend le même énoncé que la balise d'ouverture mais est précédée d'une barre oblique (/). Ainsi à la balise d'ouverture `<balise>`, correspond la balise de fermeture `</balise>`.

Par exemple, le texte :

Il est `<souligné>`important`</souligné>` d'apprendre le langage `<italique>`HTML`</italique>` !

peut se comprendre de la façon suivante :

- écrire "Il est " de façon normale puisque rien n'est spécifié,
- ensuite écrire le mot (et uniquement ce mot) "important" et le souligner,
- reprendre l'écriture normale pour "d'apprendre le langage",
- écrire cette fois-ci en italique le mot "HTML",
- et terminer par "!" en écriture normale.

- C'est le principe du Html. Chaque fois que l'on donne un ordre (une commande) au navigateur par exemple formater du texte, commencer un tableau ou faire un lien vers une autre page, on met une balise de début. La balise de fermeture signale au navigateur que la commande est terminée.

<b>	b pour bold, ce qui signifie gras.
<i>	i pour italic, ce qui signifie italique.
<p>	p pour paragraph, ce qui signifie paragraphe.
<table>	table, ce qui signifie tableau.
<color>	color, ce qui signifie couleur.
...background...	background, ce qui signifie arrière-plan.
...bgcolor...	bg pour background, ce qui signifie couleur d'arrière-plan.
etc.	

D'autres langage à balise

Avec l'essor de XML, les outils de traitement des documents XML ont considérablement évolué. Il a donc été naturel d'utiliser XML lui-même pour définir d'autres langages à base de XML. Cette façon de voir a donné naissance à de nombreux langages de balisage (car à base de XML) et d'usages très hétéroclites. Exemples d'usages :

Transformation XML : XSLT ;

Programmation : XSP1 ;

Description d'images : SVG ;

Affichage de formules mathématiques : MathML.

2.1. Les attributs des balises

Il est parfois nécessaire de compléter une commande par des spécifications plus précises. Pour ce faire, le langage Html dispose des attributs de la balise.

L'attribut s'insère dans la balise, entre le mot de commande et le signe > final.

La spécification apportée par l'attribut comporte généralement une valeur, celle-ci s'indique en complément de l'attribut par un signe égal (=) suivi de la valeur. Il est recommandé de mettre cette valeur entre guillemets. Le strict respect de la syntaxe veut qu'il n'y ait pas d'espace avant et après le signe égal.

La syntaxe complète d'un attribut est :

```
attribut="valeur"
```

Il est possible d'utiliser plusieurs attributs, séparés par un espace, dans une même balise.

2.2. Le bon usage des balises

Les balises Html ne sont pas sensibles aux majuscules et minuscules (insensibles à la casse ou case *insensitive*). Il est, pour le Html, équivalent d'écrire `<BALISE>`, `<Balise>` ou `<balise>`.

On a longtemps préconisé d'écrire les balises Html en majuscules pour les différencier du texte normal. Cependant, les évolutions annoncées du Html, comme le langage XHTML, sont, elles, sensibles aux majuscules et minuscules (sensibles à la casse ou *case sensitive*) et l'usage veut que les balises de ces langages soient écrites en minuscules afin d'éviter toute source d'erreur.

Pour vous préparer aux futurs développements du Html, il est conseillé de prendre déjà la bonne habitude d'écrire les balises en minuscules.

- En HTML5, les balises ne sont pas sensibles à la casse. Ainsi, on peut écrire indifféremment `<BALISE>`, `<Balise>` ou `<balise>`. Certains voyaient même en l'utilisation des majuscules, un moyen efficace pour distinguer le code HTML du contenu dans un document. Cependant, l'usage s'est généralisé d'écrire les balises en minuscules (comme en XHTML).
- La règle générale veut que toute balise ouverte `<balise>` doit être fermée par `</balise>`.

- Les balises doivent être correctement imbriquées. Lorsqu'on affecte plusieurs balises à un élément, l'ordre de fermeture de celles-ci est essentiel. La première balise de fermeture doit

correspondre à la dernière balise d'ouverture non fermée. Un exemple et tout sera beaucoup plus clair :

Est correct : `<a><b><c>contenu</c></b></a>`.

Est incorrect : `<a><b><c>contenu</a></c></b>`.

-
- Les valeurs des attributs doivent toujours figurer entre des guillemets. Ici aussi, la rigueur dans le code reste de mise.

3. Les CSS

3.1. Présentation

Les feuilles de style sont des ajouts de code au langage Html (ou XHtml) qui vont prendre en charge la présentation du document.

Cette présentation peut varier de la simple présentation visuelle à l'écran (police, taille de caractères, interlignes, etc.) à la présentation en vue de l'impression du document ou encore à la présentation auditive par l'intermédiaire d'une interface vocale, etc.

Le concept de feuille de style repose sur le principe de la séparation du contenu de la présentation, dans l'élaboration de documents basés sur le Html.

Ainsi un même contenu pourrait être utilisé, selon la feuille de style adoptée, pour un affichage sur des médias aussi divers qu'un écran traditionnel, l'écran d'un ordinateur de poche, l'écran d'un mobile, des feuilles de papier, un clavier braille, etc.

C'est ainsi tout le domaine de la présentation qui est pris en charge par les feuilles de style, le rôle du Html se limitant alors à la structure et à l'encodage de l'information brute.

Le langage Html, conçu à l'origine pour l'échange de documents entre scientifiques, ne comporte que peu d'éléments de présentation car cette dernière n'était pas sa préoccupation première. En quelque sorte débordée par le succès du Web auprès des internautes et l'inventivité des concepteurs de pages, le Html ne manque pas de révéler des lacunes sur le plan de la présentation et d'entraîner, à la longue, une certaine monotonie. Citons :

- une taille de caractères limitée à sept valeurs ;
- une image d'arrière-plan obligatoirement répétée en mosaïque ;
- un seul modèle de bordures de tableaux ;
- une marge unique dans les cellules de tableaux ;
- aucun contrôle de l'espace entre les lignes et entre les caractères...

Les feuilles de style vont réveiller votre créativité en apportant de nombreuses possibilités de présentation, absentes dans le code Html. On peut rapporter pour exemple :

- une taille de caractère illimitée ;
- de nouvelles présentations de bordures ;
- le contrôle de l'interlignage et de l'interlettrage ;
- l'indentation des paragraphes ;

- le positionnement précis des images ;
- la modification du curseur...

3.2. Une simplification du code

Sans mettre en doute votre compétence à lire le code source des pages Web, il faut avouer qu'avec les limites du langage et l'obligation de passer pour une présentation agréable par une foule d'astuces comme de multiples tableaux imbriqués, le code source d'une page Html devient rapidement incompréhensible. Ce qui n'est pas sans poser des problèmes, même pour une simple correction ou une mise à jour mineure.

Les feuilles de style vont permettre d'alléger considérablement votre code source en le rendant plus lisible et plus accessible.

3.3. Une maintenance de site facilitée

De par l'utilisation croissante du Web comme source d'information, la dimension des sites Web a considérablement augmenté en termes de nombre de pages par site. Il n'est plus rare de rencontrer des sites de plusieurs centaines de pages. Une mise à jour graphique peut ainsi constituer un travail gigantesque.

Plutôt que de devoir reprendre et retravailler, une par une, chacune des pages du site, les feuilles de style permettent, par la seule modification d'un unique fichier, de changer la présentation graphique de l'ensemble d'un site tout en lui assurant une cohérence graphique.

3.4. Une voie vers l'accessibilité

Depuis quelques années, l'accent a été mis sur l'accessibilité des sites Web pour les personnes présentant des déficiences visuelles. Beaucoup de concepteurs de sites ont pris conscience de l'importance du Web comme média d'information irremplaçable pour ces personnes ayant un handicap de la vue. Pour rendre un site accessible, soit procurer aux malvoyants un accès immédiat à la même information et au même volume d'information, l'usage des feuilles de style s'avère être un outil essentiel.

Sans entrer dans les détails, les feuilles de style offrent la possibilité, d'un simple clic, d'afficher une page de texte avec des caractères plus grands et donc plus lisibles.

4. Les Navigateurs

Les navigateurs de bureau comme les navigateurs mobiles ont fait des efforts particuliers pour intégrer le HTML5 sans attendre sa finalisation prévue pour le quatrième trimestre 2014. On peut affirmer que le HTML5 est globalement bien intégré dans les navigateurs récents. Cette intégration n'est cependant pas encore complète. Citons par exemple les nouveaux formulaires de HTML5 et quelques CSS3 expérimentales.

Il est important de noter que les différents navigateurs se sont conformés aux recommandations du W3C. La conformité par rapport aux standards du Web a atteint un niveau sans précédent dans l'histoire de la toile. Ce qui laisse à penser que le HTML5 sera une évolution majeure des applications Web présentes et à venir.

5. Les Bonnes pratiques

5.1. La séparation

Avec la sortie du HTML 4, le W3C propose les CSS 1. L'objectif est clair : séparer le contenu de la mise en forme et de la mise en page.

Chaque langage a son objectif : le HTML décrit la structure et le contenu des pages Web et les CSS permettent de les mettre en forme et de les mettre en page.

En travaillant de la sorte, vous n'aurez que des avantages :

- bien séparer les deux langages,
- avoir un code plus propre, plus rigoureux et plus lisible,
- séparer la gestion du contenu de la mise en forme et de la mise en page,
- centraliser la mise en forme et la mise en page en CSS,
- homogénéiser la mise en forme et la mise en page en CSS,
- avoir des mises à jour des CSS facilitées et rapides.

Donc, dans la création des pages Web, il faudra "éviter", autant que faire se peut, de "mélanger"

5.2. La sémantique

la structure, le contenu et la mise en forme ; autrement dit le code HTML et le code CSS.

Le HTML5 est un langage qui est parfaitement sémantique. Chaque élément HTML est fait pour contenir un type de contenu. À nouveau, pour avoir un code propre, lisible, valide et accessible, vous devez respecter cette structure sémantique.

L'élément `<p>` est fait pour contenir du texte courant dans des paragraphes.

L'élément `<h1>` est fait pour afficher des titres de premier niveau, pour les titres les plus importants des pages.

L'élément `<dl>` est fait pour concevoir des listes de définitions.

Donc dans votre code, utilisez à bon escient les éléments HTML, servez-vous-en pour ce à quoi ils sont destinés.

5.3. Optimiser le code et organiser vos fichiers

En tant que développeur, vous savez qu'il faut optimiser, bien organiser votre code.

Dans vos pages Web, veillez à bien indenter les lignes de code. Cela sera toujours plus facile à reprendre par la suite, que ce soit par vous ou une autre personne.

Les commentaires sont indispensables pour bien expliquer votre code. À nouveau, c'est faciliter la tâche aux personnes qui reprendront vos pages plus tard ou pour vous-même.

Essayez de nommer les sélecteurs CSS de manière intelligible et logique. Encore une fois, c'est du bon sens et il est toujours plus facile et rapide de reprendre du code bien créé.

Pour l'organisation des fichiers, prenez la bonne habitude de créer des dossiers par types de fichiers utilisés dans vos développements. Il est très classique d'avoir un dossier **css** pour tous les fichiers feuilles de style CSS, un dossier **js** pour tous les fichiers JavaScript, ou encore un dossier **img** pour les médias de type images.

Et bien sûr, mais vous le savez, faites régulièrement des sauvegardes et utilisez, pourquoi pas, un outil de gestion des versions.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8" />
<style>
.titre {
font-size: 16pt;
font-weight: bold;
}
</style>
</head>
<body>
<p class="titre">Maecenas faucibus mollis interdum</p>
<p>Curabitur blandit tempus porttitor...</p>
<p><center>Aenean eu leo quam. Pellentesque...</center></p>
Cras mattis consectetur purus sit amet...
<p> </p><p> </p>
<h2>Nullam quis risus eget urna mollis ornare vel eu leo</h2>
<p>Sed posuere consectetur est at lobortis...</p>
<table>
<tr>
<td><img src="tigre.jpg" /></td>
<td><img src="suricate.jpg" /></td>
<td><img src="loutre.jpg" /></td>
</tr>
</table>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
    <title>Ma petite page web</title>
    <meta charset="UTF-8" />
    <style>
        .paragraphe-centre {
            text-align: center;
        }
        .espace-avant {
            margin-top: 68px;
        }
        .img-flotte-gauche {
            float: left;
            margin-right: 10px;
        }
    </style>
</head>
<body>
    <h1>Maecenas faucibus mollis interdum</h1>
    <p>Curabitur blandit tempus porttitor...</p>
    <p class="paragraphe-centre">Aenean eu leo quam. Pellentesque...</p>
    <p>Cras mattis consectetur purus sit amet...</p>
    <h2 class="espace-avant">Nullam quis risus eget urna mollis
ornare vel eu leo</h2>
    <p>Sed posuere consectetur est at lobortis...</p>
    <div>
        <p><img src="tigre.jpg" class="img-flotte-gauche" />
        <img src="suricate.jpg" class="img-flotte-gauche" />
        <img src="loutre.jpg" class="img-flotte-gauche" />
        </p>
    </div>
</table>
</body>
</html>
```

6. LE HTML 5 Présentation

- Le HTML5 est une évolution du HTML.
- Le principe de la séparation du contenu et de la présentation reste de mise et se voit même renforcé.
- La simplification du code et le désir d'éviter toutes complications souvent inutiles sont deux principes suivis dans l'élaboration du HTML5
- Une chasse aux plug-ins qui viennent encombrer le code et compliquer la tâche des concepteurs, par leur prise en charge directe dans le navigateur
- La reconnaissance du JavaScript comme partenaire de la publication sur la toile
- Le HTML5 devient également une plateforme d'interfaces d'applications (API)

6.1. Les nouveautés

- Un nouveau doctype simplifié et unifié.
- La suppression des balises et attributs de présentation.
- De nouvelles balises sémantiques ou d'organisation.
- De nouvelles balises audio et vidéo qui ne nécessitent plus l'appel à des plug-ins dédiés.
- Le dessin 2D et bientôt 3D par la nouvelle balise `<canvas>`.
- Une profusion de formulaires novateurs comme les curseurs ou les calendriers et la prise en charge de façon native par les navigateurs de la validation des données.
- Etc.

6.2. Les API JavaScript

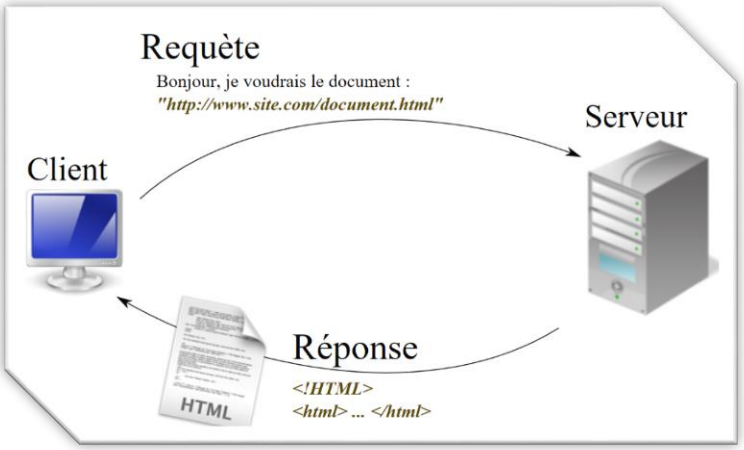
- La géolocalisation qui permet de localiser (avec sa permission) l'utilisateur par ses coordonnées de longitude et latitude.
- Des super cookies avec *Web Storage* qui permettra un stockage plus important de données dans le navigateur.
- L'utilisation des applications Web hors connexion après la mise en cache des ressources nécessaires.
- Les *Workers* qui peuvent exécuter des tâches de fond en parallèle du programme JavaScript principal dans un environnement totalement séparé de la page.
- Les *Websockets* qui permettent d'établir une communication bidirectionnelle asynchrone entre le navigateur et le serveur.
- Le glisser-déposer (*drag/drop*) en natif dans le navigateur.

- L'attribut *ContentEditable* qui permet l'édition en ligne du contenu d'un élément. Il fait apparaître un éditeur WYSIWYG basique qui permet donc d'éditer directement le contenu dans la page. Etc.

NB : Un certain nombre de ces fonctions sont déjà reprises dans les cadriciels (framework) JavaScript comme Dojo et jQuery.

7. Coté réseau

Traditionnellement les modèles Clients/Serveur sont associés à des architectures deux tiers. Dans une relation client/serveur, un programme (le client) demande un service ou une ressource à un autre programme (le serveur)



Le client établit une connexion au serveur sur un réseau local,ou étendu tel qu'internet. Lorsque le serveur a répondu à la demande du client, la connexion est terminée. Un navigateur Internet est un programme client qui demande un service à un serveur. Le service et la ressource fournis par le serveur donnent lieu à l'affichage d'une page Web.

Dans cette architecture il est nécessaire de devoir installer un programme client sur le poste de travail. Il faut donc définir des caractéristiques minimales du poste de travail et des compatibilités avec les différents systèmes d'exploitation. Ainsi il faut développer la partie client pour les différents systèmes d'exploitation (Windows, Linux, Mac OS, …). Ceci peut représenter des coûts importants.

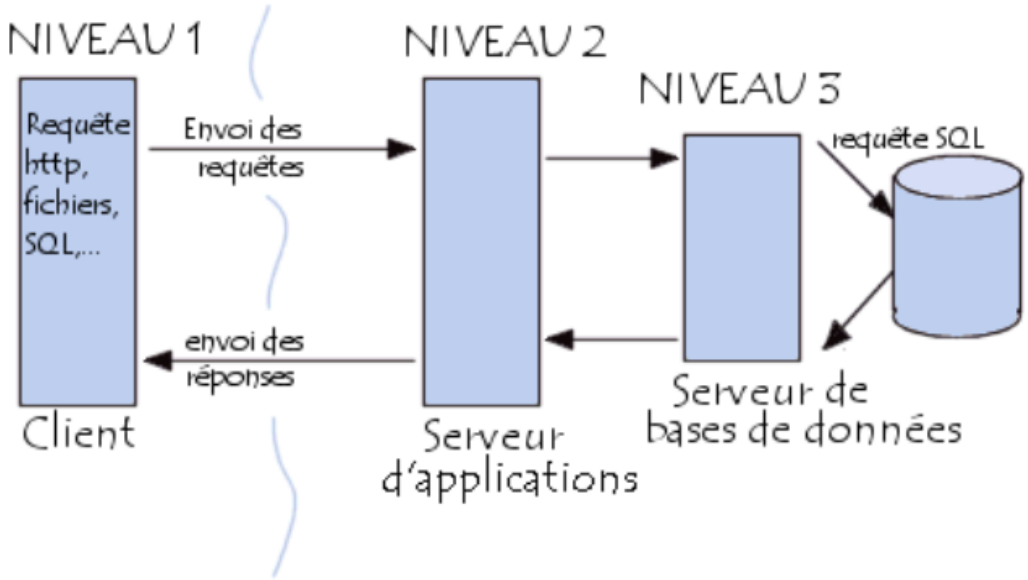
L'architecture des applications Clients/serveur est composée de trois couches d'abstraction,

- la couche de présentation,
  - HTML5
  - ANGULAR JS
  - CSS 3
  - JAVASCRIPT
  - JQUERY
- La logique applicative, les traitements,
  - PHP
  - PYTHON
  - ASP

- PERL
- …

– Les données
  - MICROSOFT SQLSERVER
  - ORACLE
  - POSTGRESQL
  - MYSQL

Dans l'architecture 3-tiers, un nouveau niveau fait son apparition. En effet, nous avons toujours le niveau 1 qui est le client. Le client est très léger étant donné qu'il n'a aucun rôle de traitement. Au niveau 2 nous avons le serveur d'application et enfin, au dernier niveau le serveur de base de donnée.



Source : www.commentcamarche.net

Dans l'architecture 3 tiers, l'utilisation de la couche WEB permet de s'abstenir de développer une application à installer sur le poste client. L'utilisation d'un navigateur WEB comme application client permet de faire abstraction de l'OS du client et du serveur. Les échanges passent par des protocoles (HTTP / HTTPS / …).