TP création et publication d'un site web statique

Ce TP va se décomposer en trois parties. Dans un premier temps nous créerons un site web statique. Le site sera créé avec l'éditeur Amaya pour respecter au mieux les recommandations du W3C. Il existe sous Linux d'autres outils de développement Web comme NVU, BLUEFISH et QUANTA+ (que je vous recommande aussi). Certains de ces éditeurs tournent aussi sous Windows (Amaya, NVU). Dans un second temps nous installerons et configurerons un serveur web apache2 avant de publier notre site. Enfin dans un troisième temps nous ferons évoluer notre site web.

En bonus, la partie 4 vous apprendra à créer des serveurs virtuels et enfin la partie 5 est un apport de cours sur le client/serveur.

PARTIE 1

I. Préparation

Connectez votre machine Linux sur le réseau du lycée. Mettre une adresse IP standard pour la salle de cours. Toutes les machines doivent donc « se voir » via le ping.

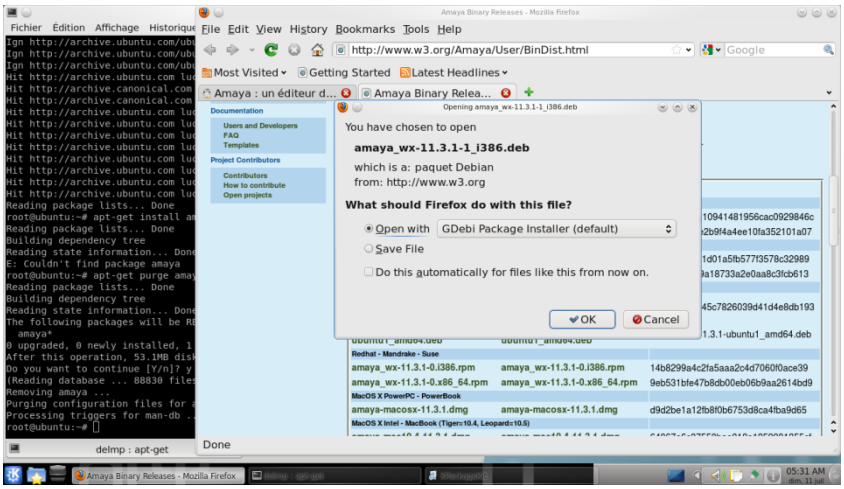
Rajoutez la passerelle éventuellement par la commande « route add default gw 172.16.0.199 ». Vérifiez votre table de routage.

Vérifiez vos serveurs DNS dans le fichier /etc/resolv.conf.

Changez, si nécessaire, le nom de votre machine Linux comme vous l'avez déjà fait au cours du TP précédent.

II. Installation de Amaya

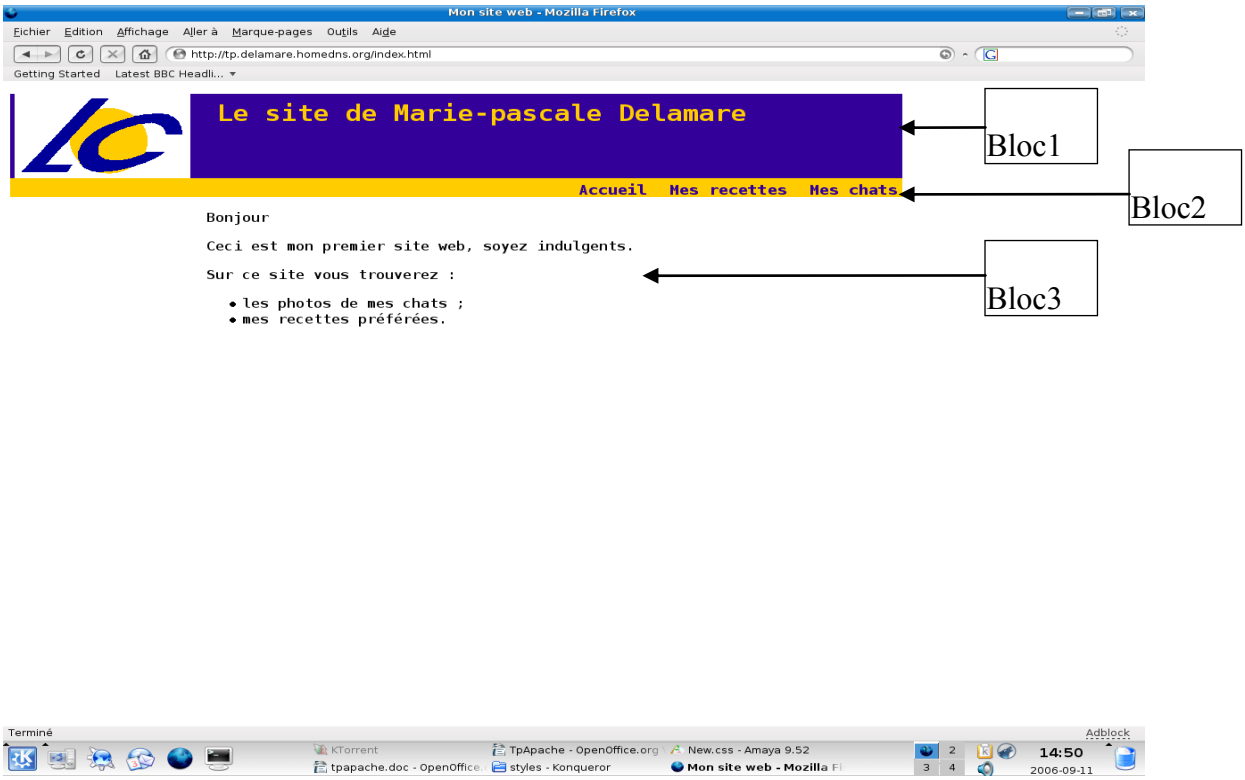
Allez sur la page d'accueil d'Amaya pour télécharger le .deb relatif à la version Ubuntu, et pour l'installer en utilisant l'installeur graphique ou en mode commande en vous positionnant dans votre répertoire dowload (si vous avez utilisé firefox) et en tapant la commande « dpkg –i (comme install) nom_du_package_à_installer. Ce qui donne dpkg –i amaya_wx-11. 3.1-1_i386.deb.



III. Création du contenu

Une fois Amaya lancé, choisissez, « Fichier », « Nouveau », « Nouveau document ». Choisissez votre répertoire personnel comme lieu de stockage, renseignez le titre en mettant votre nom, choisissez XHTML 1.1 et le code de caractères UTF-8 et enfin nommer le index.html. Ceci fait dans le menu « Affichage », choisissez d'afficher le code source. Vous apercevez les premières balises de votre page web.

Nous voulons obtenir le résultat suivant :



Si on analyse la page, on s'aperçoit que l'on veut un bloc d'en-tête contenant le logo et le titre, un second bloc qui contient la liste des rubriques et un troisième bloc de contenu.

Commencez par taper le texte du bloc de contenu dans la fenêtre du haut, sans vous préoccupez de la mise en page (vous devez cependant créer la liste à puce en utilisant les icônes proposées dans le cadre supérieur droit et marquer les paragraphes en tapant sur la touche « entrée »).

Créez maintenant votre entête. Revenez au dessus de « Bonjour » et cliquez sur l'icône ajout d'image dans le cadre supérieur droit. Choisissez l'image à ajouter (vous pouvez récupérer le logo du lycée sur le site web du lycée). Pour créer correctement votre site web, vous créez un répertoire « Images » dans votre répertoire personnel, et dans lequel vous enregistrez l'image choisie. Puis placez-vous sous l'image et tapez votre titre, ici « Le site de votre_nom ». On ne se préoccupe toujours pas de la mise en page.

Il reste à créer la liste des rubriques. Ces rubriques sont en fait des liens vers les autres pages Web de votre site. Tapez les mots « Accueil », « Mes recettes » et « Mes chats » sur la même ligne. Sélectionnez le mot « Accueil » et cliquez sur l'icône lien dans le cadre supérieur droit. Tapez le

nom du fichier contenant votre page d'accueil, ici index.html. Faire la même chose pour « Mes recettes » et « Mes chats » (il faudra ici penser à créer ces deux fichiers).

IV. Définition de la mise en page

Voilà, il reste maintenant à s'occuper de la mise en page. Pour cela nous allons créer un nouveau fichier que nous appellerons une feuille de styles. Faites « Fichier », « Nouveau », « Nouvelle Feuille de style ». Choisissez le codage « UTF8 » et **sauvegardez-la** dans votre répertoire courant. Attention vérifiez bien que le fichier feuille de style est bien créé dans votre répertoire.

Vous devez d'abord associer cette feuille de styles au document courant. Pour cela restez dans l'onglet de votre page web, puis choisissez « Format », « Feuille de style », « Ajouter » et sélectionnez le fichier feuille de style que vous venez de créer. Pour voir les changements dans votre code source, vous devez enregistrer votre page web. Si vous observez le code source, vous vous apercevez qu'une balise « link » vient d'être créée à l'intérieur de vos balises « head » et qu'elle pointe vers votre fichier feuille de styles.

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

Avant de créer vos styles, vous devez définir les différents blocs de votre page web. Pour cela, dans votre page web, sélectionnez tout le texte de votre bloc d'entête, puis cliquez sur l'icône « Division » du cadre supérieur droit. Vous voyez apparaître une balise avant le texte sélectionné et une balise fermante après le texte sélectionné. Dans la fenêtre inférieure, éditez la balise ouvrante comme ceci <div id= « bloc1 »>. Faire de même pour les autres blocs de votre page mais en remplaçant bloc1 par bloc2 et bloc3.

```
 <div id="bloc1">
        <p><img alt="Logo"  src="../images/logo.gif" /></p>
        <p>Le site de Marie-pascale Delamare</p>
</div>
```

Nota : Nous avons utilisé une balise « div ». Cette balise permet de définir des regroupements au sein d'une page HTML. Ces regroupements sont placés dans des boites dont on peut définir le style.

Dans votre feuille de styles sur la ligne 1, tapez « body {», puis choisissez l'outil « CSS » dans la barre d'outils. Une boîte de dialogue apparaît vous permettant de spécifier votre police de caractères, les cadres etc. Choisissez la police « courrier new » et la font-size « larger ». Fermez la boîte de dialogue. Ces caractéristiques s'inscrivent dans votre feuille de styles. Rajoutez une accolade fermante « } ». Ces choix s'appliquent désormais à toute votre page web, car celle-ci est comprise entre les balises ouvrante <body> et fermante </body>. Enregistrez votre feuille de style et vérifiez l'impact de votre feuille de style sur votre page web en retournant dans l'onglet de cette dernière.

Nota : Nous avons ici redéfini le style d'une balise HTML prédéfinie. Dans ce cas, on écrit simplement le nom de la balise HTML dans la feuille de styles et on ne précise que les éléments à redéfinir. De façon générale la définition d'une règle de style est constituée de deux parties principales :

- le sélecteur ici body ;
- la déclaration contenue entre accolades et constituée elle-même de :

- une ou plusieurs propriété(s), toujours suivie(s) d'un double point ici font-size par exemple ;
- une valeur, toujours suivie d'un point virgule ici larger ;

Le style associé à une balise HTML s'appliquera à tous les éléments du code contenu entre la balise ouvrante et la balise fermante, y compris aux autres balises HTML contenues qui vont héritées du style défini sur la balise de niveau supérieur, à moins qu'elles ne fassent elle-même l'objet d'une règle de style. On dit que les caractéristiques se propagent en cascade d'où le nom CSS, Cascading Style Sheet.

Continuons à définir la mise en page de nos différents blocs. Dans la feuille de styles, tapez maintenant « #bloc1 {», puis choisir l'outil « CSS ». Cette fois allez dans l'onglet couleurs et choisir une couleur dans la palette proposée. Fermez la boîte de dialogue et rajouter une accolade fermante. Enregistrez votre feuille de styles et vérifiez l'effet obtenu sur votre page web.

Nous voulons maintenant aligner le logo et le titre. Pour cela, rajouter dans le code HTML de la fenêtre inférieure un id= « logo » entre les balises d'insertion de l'image et id= « titre » dans la balise ouvrante du paragraphe du titre. Retournez dans votre feuille de styles et tapez « #titre { », faîtes apparaître la boîte de dialogue et précisez les informations voulues (voir la feuille de styles fournie en annexe). Faîtes de même pour le logo.

```
<div id="bloc1">
        <p><img alt="Logo" id="logo" src="../images/logo.gif" /></p>
        <p id="titre">Le site de Marie-pascale Delamare</p>
</div>
```

Nota : Nous avons utilisé ici le sélecteur « id » (identifiant). Ce sélecteur ne permet d'appliquer une règle de style qu'à un seul élément HTML. Un « id » doit être unique au sein d'une page HTML. Dans la feuille de styles on fait précéder son nom d'un dièse #bloc1, #titre, #logo.

Voyons maintenant notre menu. Dans la feuille de styles, tapez « #bloc2 {» et saisissez les caractéristiques souhaitées.

Pour ce bloc, il nous reste à aligner nos liens sur la droite de la page web. Retournez dans votre page web. Dans la fenêtre inférieure, entourez vos liens d'une nouvelle balise ouvrante <div class= « rubriques »> et d'une balise fermante . Retournez dans votre feuille de styles et tapez « .rubriques { » . Faîtes apparaître la boîte de dialogue et renseigner les informations voulues. Nous voulons maintenant que les liens ne soient pas soulignés, qu'ils apparaissent en gras et soient écrits en bleu. Dans votre feuille de style, tapez « .rubriques a { », faîtes apparaître la boîte de dialogue et saisissez les informations souhaitées.

```
<div class="rubriques">
        <a href="index.html">Accueil</a>
        <a href="recettes.html">Mes recettes</a>
        <a href="chats.html">Mes chats</a>
</div>
```

Nota : Nous avons utilisé ici le sélecteur « class » (classe). Ce sélecteur permet d'appliquer une règle de style à plusieurs éléments HTML, éventuellement de types différents paragraphes (balise

« p »), titres (balises « h1 », « h2 », « h3 »), etc. Il suffit dans le code HTML de préciser que la mise en page de cet élément sera définie par son nom de classe (class=). Un nom de classe peut donc apparaître plusieurs fois au sein d'une page HTML. Dans la feuille de style on fait précéder son nom d'un « . » .rubriques. La seconde déclaration « .rubriques a » permet de définir un style particulier pour les balises « a » (liens) de la classe « rubriques ».

Bien maintenant centrons le bloc de contenu. Je vous laisse faire.

Vous pouvez maintenant créer, par copie du code source, vos deux autres pages web. Il suffira de modifier le texte du bloc de contenu.

V. Vérification de la conformité aux recommandations du W3C

Pour cela nous allons sur le site http://validator.w3.org

Nous choisissons le choix « Validate By File Upload » et ceci pour chacune de nos pages.

Nous vérifions aussi la conformité de notre feuille de styles.

Notre site est donc conforme aux recommandations du W3C.

VI. Intérêt de la feuille de styles externe

Les indications de mise en page auraient pu être intégrées directement au code HTML, mais ceci ne permet pas une bonne maintenance de votre site web.

En effet si vous voulez changer une couleur de fond par exemple, avec une feuille de styles externe vous ne faîtes qu'une modification dans la feuille de styles et elle s'applique immédiatement à toutes les pages. Dans le cas où, vous avez inséré les indications de mise en page dans le code HTML, vous devez reprendre le code de chacune de vos pages. Donc si vous avez 10 pages cela fait 10 fois plus de travail.

D'autre part un site web peut proposer plusieurs mises en page à ses visiteurs (pensez aux malvoyants par exemple qui ont besoin de polices plus grosses) uniquement si les indications de mise en page se trouvent dans des fichiers externes. Pensez-y pour vos petits sites web.

Nota : Si vous voulez approfondir cette partie, vous pouvez acheter le hors série numéro 1 du magazine Linux pratique (6,40 €).

ANNEXE : Feuille de styles

```
body {  font-family: Courier New,Courier,monospace;
  font-size: larger;
}
#bloc1 {
  background-color: #0000ff;
  height: 132px                    Attention cette hauteur doit être en cohérence avec la hauteur de votre image en pixel
}
#logo {
  margin-top: 10px;
  margin-bottom: 10px;
  margin-left: 10px;
  margin-right: 10px;
  float: left
}
#titre {
  font-size: xx-large;
  font-weight:bold;
  color: #ffa500;
  margin-top: 6px;
  margin-left: 60px;
  padding-left: 100px;
  padding-top : 30px
}
#bloc2 {
  background-color: #ffa500;
  height: 40px ;
}
.rubrique {
  margin-right: 10px;
}
.rubrique a {
  float: right;
  font-size: medium;
  font-weight:bold;
  text-decoration: none;
  color: #0000ff;
  margin-left: 15px;
}
#bloc3 {
  margin-left: 227px;                    ATTENTION, ce décalage doit être en phase avec la largeur de votre image en pixel
}
```

Voici donc la suite de notre TP. Dans un premier temps nous avons créé un site web statique. Ce site a été créé avec l'éditeur Amaya pour respecter au mieux les recommandations du W3C. Aujourd'hui nous installerons et configurerons un serveur web apache2 avant de publier notre site. Enfin dans un troisième temps nous ferons évoluer notre site web.

En bonus, la partie 4 vous apprendra à créer des serveurs virtuels et enfin la partie 5 est un apport de cours sur le client/serveur.

PARTIE 2

I. Installation du serveur web apache2

Vérifiez les ports d'écoute ouverts par la commande netstat –nl --protocol=inet (cela ne présentera que les ports d'écoute internet ouverts sous forme numérique).

En console : apt-get update suivi de apt-get install apache2. Acceptez l'installation des dépendances requises.

Vérifiez que votre serveur Apache2 fonctionne en interrogeant les processus lancés. Apache2 tourne-t-il ?

Vérifiez quel nouveau port d'écoute a été ouvert par Apache2 en utilisant la commande netstat -nl. Quel est le port d'écoute du serveur web Apache2 ?

Interrogez votre serveur web en utilisant Firefox ou Konqueror. Pour cela, utilisez soit l'URL http://127.0.0.1 soit l'URL http://localhost.

Il nous reste maintenant à configurer Apache2 et à installer notre site web.

II. Configuration de apache2

Nous allons changer le répertoire racine du site web.

En root, créez un répertoire /var/www/votre_nom et donnez-le à l'utilisateur www-data (qui est l'utilisateur sous lequel tourne le serveur apache2) et au groupe www-data.

Avec vim, éditez le fichier /etc/apache2/sites-available/default. Vous devez rechercher le paramètre « DocumentRoot » qui indique à apache2 où se situe son répertoire racine. Changez-le en /var/www/votre_nom, et remplacez ensuite toutes les occurrences de /var/www par /var/www/votre_nom.

Toute modification effectuée dans un fichier de configuration nécessite l'arrêt et le redémarrage du serveur web. Pour l'arrêter : /etc/init.d/apache2 stop, pour le redémarrer /etc/init.d/apache2 restart ou apche2ctl restart.

Vérifiez que des processus apache2 tournent bien sur votre machine.

Vérifiez votre modification en interrogeant votre serveur web.

Que se passe-t-il ?

III. Installer votre site web

Récupérez maintenant votre site créé avec Amaya et présent dans votre répertoire personnel et

copiez le dans le répertoire /var/www/votre_nom. Vous devez avoir une page index.html, une page recettes.html, une page chats.html et votre feuille de style. N'oubliez pas de copier aussi le répertoire contenant les images de votre site web.

Vérifiez, voir modifiez les droits sur tout le contenu de /var/www/votre_nom. Il doit appartenir à l'utilisateur www-data et au groupe www-data.

Interrogez votre site web. Que se passe-t-il ?

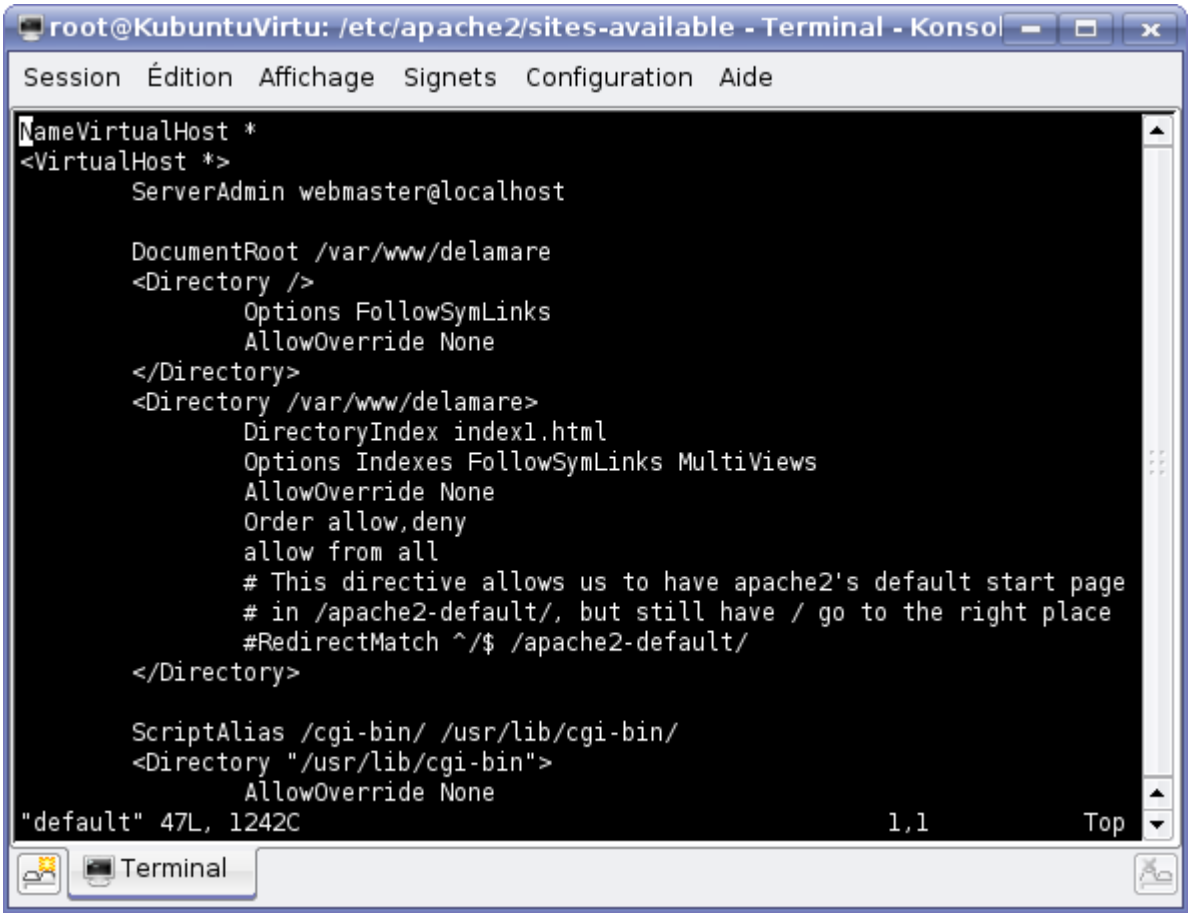
Il existe une directive sous Apache2, la directive « DirectoryIndex » qui définit la liste des ressources à rechercher lorsqu'un client fait une demande au serveur. Si la directive « DirectoryIndex » n'est pas présente, le fichier recherché par défaut par le serveur Apache2, dans le répertoire racine du site web, soit /var/www/votre_nom dans votre cas, est le fichier « index.html ».

Renommez dans le répertoire /var/www/votre_nom, votre fichier « index.html » en « index1.html ».

Interrogez votre site web. Que se passe-t-il ?

Si cette page « index.html » n'est pas présente, vous voyez apparaître le contenu du répertoire ciblé, c'est à dire la liste des fichiers déposés à la racine de votre site web.

Rajoutez maintenant une ligne « DirectoryIndex index1.html » dans la directive <Directory /var/www/votre_nom> du fichier /etc/apache2/sites-available/default. Ce qui donne :



Redémarrez Apache2 par /etc/init.d/apache2 restart ou la commande apache2ctl restart.

Interrogez votre site web. Que se passe-t-il ?

Pour la suite du TP, changez la directive « DirectoryIndex » pour la remettre à la valeur

index.html. Renommez votre fichier index1.html en index.html sinon vos liens ne fonctionneront plus.

IV. Nommer votre site web

a) Résolution de noms sans serveur DNS

Si vous avez lu le message qui apparaît à chaque démarrage de votre serveur web, vous savez que celui-ci n'a pas encore de nom. Nous allons commencer par lui donner un nom. Pour cela éditez le fichier /etc/apache2/sites-available/default et rajoutez sous la balise <VirtualHost *>, le paramètre ServerName suivi du nom que vous voulez lui donner par exemple st3d1.gsi.local (conservez le nom de domaine proposé, car nous utiliserons un serveur DNS par la suite).

Essayez d'atteindre votre serveur web par son nom dans firefox. Que se passe-t-il?

En fait si le serveur web est nommé, nous n'avons pas mis en place de mécanisme de résolution de nom qui permette d'associer à ce nom une adresse IP. Nous n'avons pas de serveur DNS sur notre réseau local pour l'instant et c'est normalement le rôle d'un serveur DNS d'effectuer cette résolution de nom. Nous allons donc utiliser le fichier /etc/hosts. Éditez ce fichier et rajoutez une ligne du type : 172.16.2.51 Linuxst2a1.gsi.local et vérifiez que sur la ligne 127.0.0.1 le nom que vous avez donné à votre machine Linux au cours du dernier TP est bien inscrit en face de cette adresse.

Essayez d'atteindre votre serveur web par son nom dans firefox. Que se passe-t-il?

Tous vos serveurs sont connectés au même réseau. Essayez d'atteindre le serveur web d'un quidam par son adresse IP. Que se passe-t-il ?

Par son nom. Que se passe-t-il ?

Que devez vous faire pour atteindre les serveurs web de vos camarades par leur nom ? Faîtes cette modification.

Résultat ?

Nota : Il existe aussi un fichier hosts sous Windows.

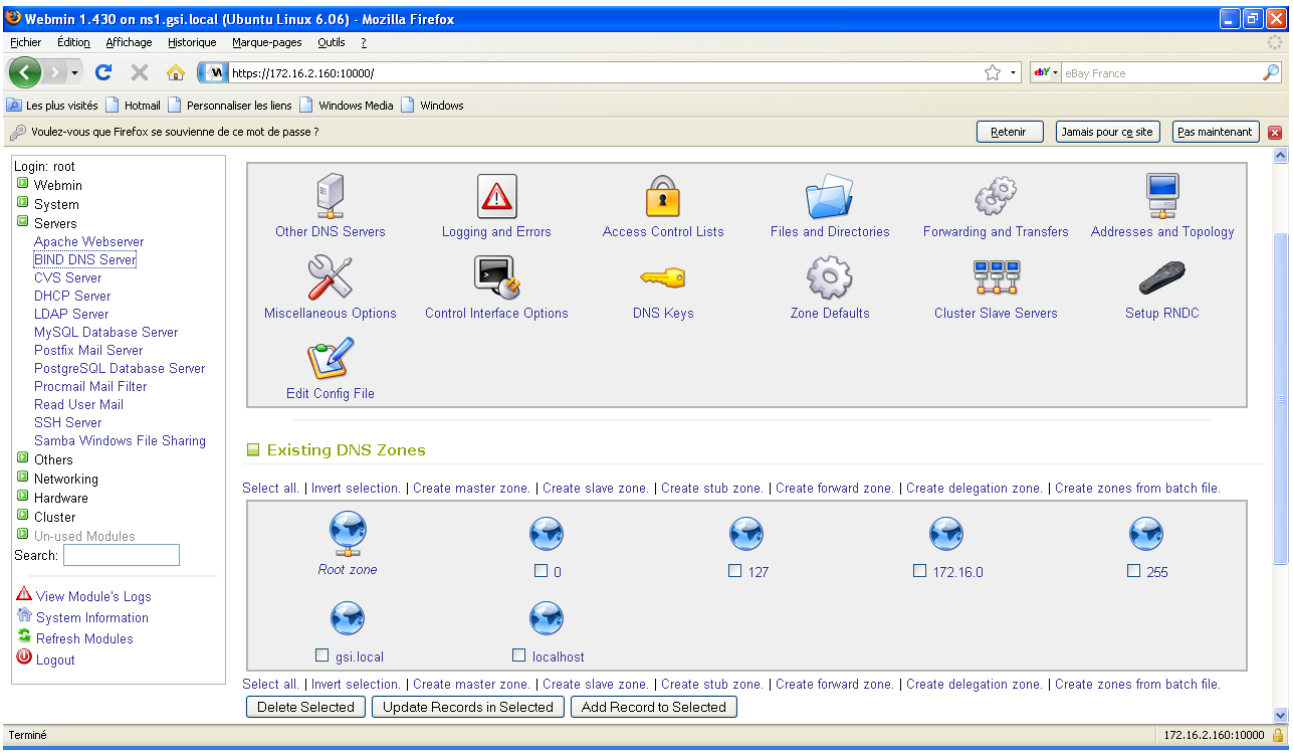
b) Résolution de noms avec serveur DNS

Pour commencer détruisez toutes les lignes que vous venez d'insérer dans votre fichier /etc/hosts.

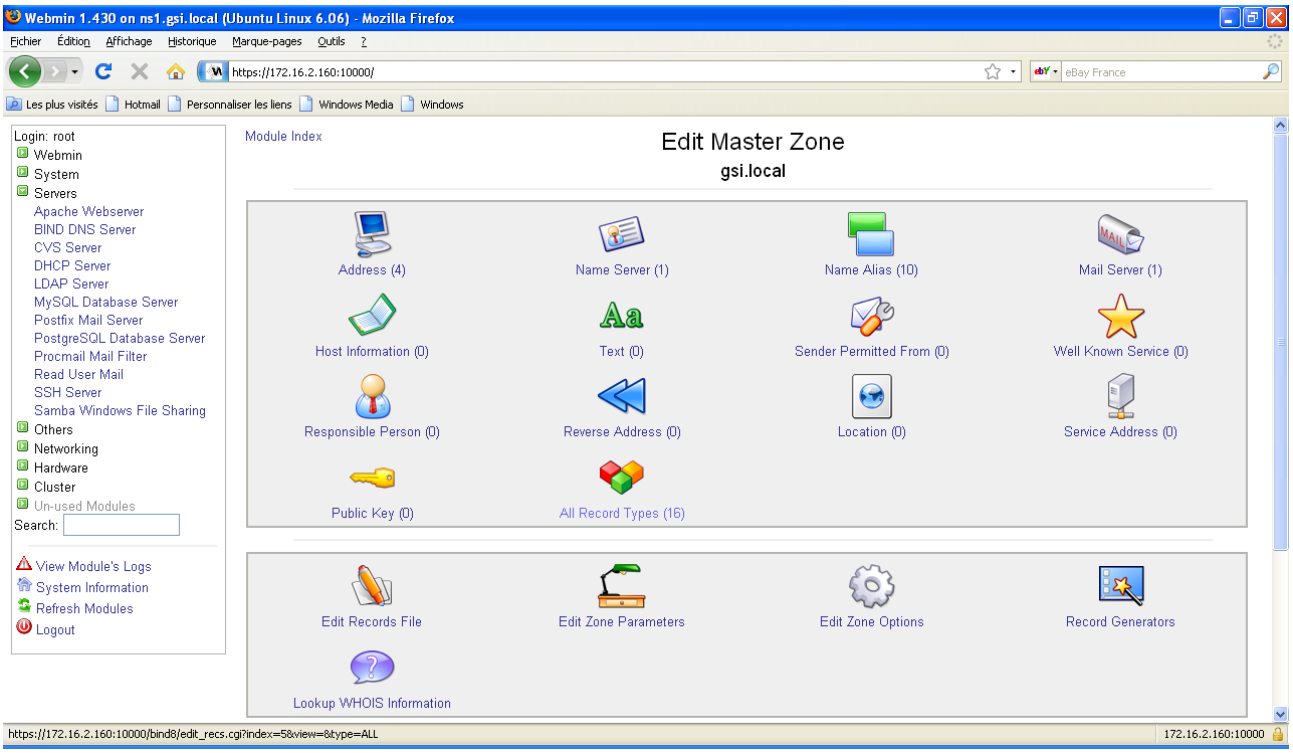
Notre server DNS est situé à l'adresse 172.16.2.160. Il dispose de l'outil d'administration Webmin. Grâce à cet outil, le serveur DNS est administrable à distance via un simple navigateur internet.

Création du nom de votre machine dans le serveur DNS.

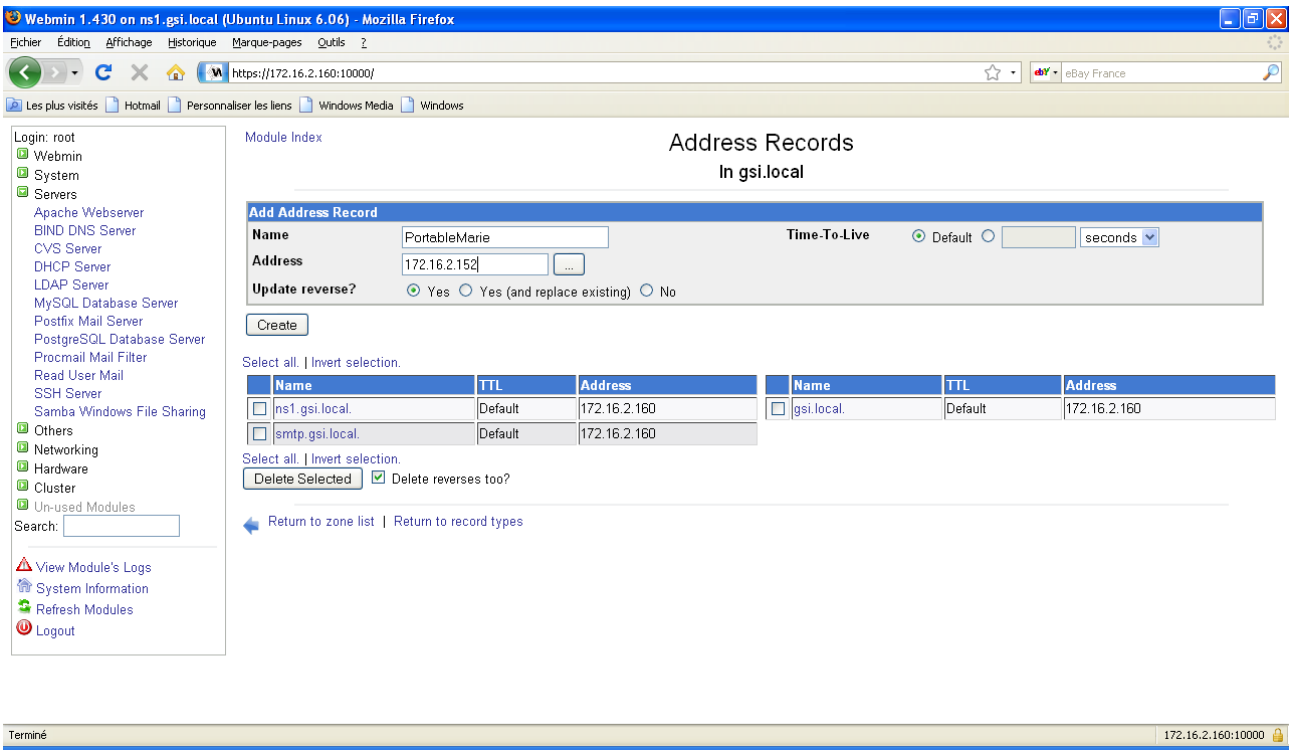
Dans votre navigateur, tapez l'URL https://172.16.2.160:10000 et connectez vous en root, mot de passe root. Dans le cadre supérieur gauche, développez l'option « Servers » et choisissez « BIND DNS server ». Puis dans la partie « Existing DNS zones »,



cliquez sur l'icône « gsi.local », puis sur l'icône « Address » :



Saisissez et créez le nom de votre machine (sans nom de domaine) et votre adresse IP :



Choisissez alors « Return to zone list » en bas de l'écran et cliquez sur « Apply Changes » en bas de l'écran.

Vous venez de créer votre nom de machine dans les fichiers de configuration du serveur DNS. Il reste maintenant à l'utiliser.

Utilisation du serveur DNS

Vous devez indiquer à votre machine où se situe l'adresse du server DNS que vous voulez désormais interroger en priorité. Il suffit d'indiquer l'adresse de ce serveur DNS dans votre configuration réseau (Paramètres du système, Configuration réseau , onglet « Système de noms de domaines). Supprimez alors les adresses de vos anciens serveurs pour les remplacer par l'adresse du serveur DNS que vous venez de configurer.

Interrogez maintenant votre site web par son URL. Que se passe-t-il ?

V. Protéger votre site Web

Vous devez vérifier la présence du script *htpasswd*. Recherchez ce script :

Vous allez ensuite créer, grâce à cette commande, un fichier contenant les utilisateurs du serveur web ainsi que leur mot de passe. Créez deux utilisateurs (l'existence sous Linux n'est pas une obligation sauf si une partie des pages publiables est rangée dans les répertoires des utilisateurs que l'on crée ce qui n'est pas le cas ici).

Allez dans /var/www/votre_nom. Pour le premier utilisateur tapez *htpasswd –c .htpasswd user1*.

Puis créez le deuxième utilisateur en tapant la commande *htpasswd .htpasswd user2* etc. Il y a toujours le man si vous avez des difficultés. Vérifiez le contenu du fichier /var/www/votre-nom/ *.htpasswd.*

Vous devez maintenant expliquer à apache qu'il doit prendre en compte ces utilisateurs pour gérer les droits d'accès à votre site web. Pour ce faire, il faut modifier de nouveau le fichier /etc/apache2/sites-available/default et remplacer le paramètre « AllowOverride None » **du répertoire de votre site web** par « AllowOverride AuthConfig ».

Il faut ensuite créer un fichier « .htaccess » dans le répertoire contenant votre site web et insérer dans ce fichier les lignes suivantes :

```
AuthType Basic
AuthUserFile /var/www/votre_nom/.htpasswd
AuthName "Privé"
require user user1
```

Ou la signification des paramètres est la suivante :

AuthUserFile /var/www/votre_nom/.htpasswd : Nom du fichier contenant les comptes des utilisateurs apache2
A*uthName* "Privé" : Titre de la fenêtre de connexion
require user user1 : Type d'authenfication "user" pour compte utilisateur et nom du compte utilisateur attendu (user1).

Relancez apache2. Testez vos modifications. Qui a accès à votre site web ?

Nota : Nous aurions pu gérer les autorisations d'accès par groupe d'utilisateur. Pour cela il aurait fallu créer aussi un fichier .htgroup dont la structure est la suivante : nomdugroupe: util1 util2, où nomdugroupe représente le nom du groupe et util1 util2 représentent les utilisateurs rattachés à ce groupe. Ensuite il fallait changer la ligne, dans le fichier .htaccess, « require user user1 » par « require group nom_du_groupe » et le paramètre *AuthUserFile* par *AuthGroupFile.*

Voyons maintenant les choses en grand.

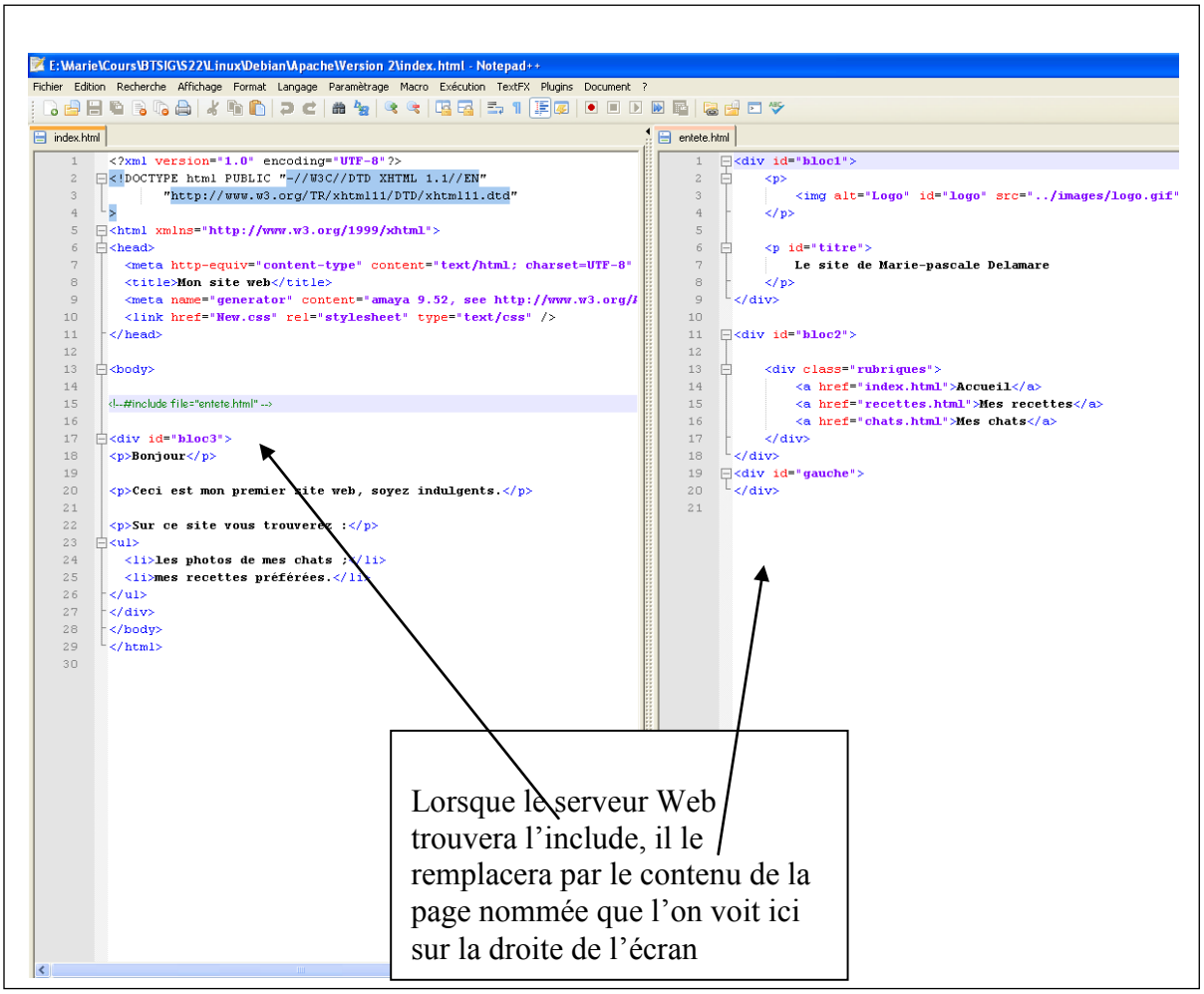
PARTIE 3

I. Expression du besoin

Nous voulons modifier le bloc d'entête, par exemple en modifiant le logo. Tel que nous avons conçu notre site, nous devons modifier chacune des trois pages, ce qui n'est pas vraiment intéressant puisque la modification à réaliser est exactement la même dans chaque page.

Nous aurions dû factoriser le code commun aux différentes pages (comme vous écrivez des sous-programmes ou des fonctions pour les parties communes de code en programmation). Ce qui revient à dire que nous souhaitons inclure dans une page html une autre page html.

Il existe plusieurs méthodes pour réaliser ceci. Nous allons mettre en place la méthode que l'on appelle le Server Side Include ou SSI. Dans cette méthode c'est le serveur web qui va scruter une page html, avant de la servir au client, à la recherche d'un include. Si un include est trouvé, alors le serveur web insérera, dans le fichier html scruté, le contenu du fichier html trouvé sur la ligne de l'include. Cette méthode nécessite de modifier le paramétrage de notre serveur web apache.



Nota : On aurait pu utiliser l'instruction « include » en php. Mais cela suppose que php soit installé sur la machine hébergeant le site web. C'est à dire que dans ce cas notre site web n'est plus statique mais dynamique. Vous verrez sans doute cela en S3.

II. Modification de la configuration d'apache

Dans un premier temps nous devons activer le module « include » de notre serveur web. Pour cela il faut créer un lien symbolique à partir du répertoire /etc/apache2/mods-enabled vers le fichier /etc/apache2/mods-available/include.load. Ceci s'effectue avec la commande ln :

ln -s /etc/apache2/mods-available/include.load /etc/apache2/mods-enabled/include.load

Ensuite dans le répertoire /etc/apache2/sites-available, ouvrez le fichier default (en fait dans le cas où nous aurions plusieurs sites web publiés au sein du même serveur web il faudrait choisir le fichier correspondant au site web pour lequel on veut activer le SSI). Recherchez alors la balise <Directory /var/www/votre_repertoire_racine> et modifiez le contenu pour obtenir ceci :

```
<Directory /var/www/votre_repertoire_racine/>
                Options Indexes FollowSymLinks MultiViews +Includes
                AddType text/html .html
                AddOutputFilter INCLUDES .html
                AllowOverride None
                Order allow,deny
                allow from all
</Directory>
```

Remarque : voici la signification des paramètres

+Includes : active le SSI (Server Side Include).
AddType text/html .html, AddOutputFilter INCLUDES .html : précisent le type de document à inclure.

Redémarrez votre serveur web.

III. Évolution du site web

Dans Amaya, ouvrez une de vos pages et sélectionnez la partie commune du code HTML.

```
<div id="bloc1">
        <p><img alt="Logo" id="logo" src="../images/logo.gif" /></p>
        <p id="titre"> Le site de Marie-pascale Delamare</p>
</div>
<div id="bloc2">
        <div class="rubriques">
                <a href="index.html">Accueil</a>
                <a href="recettes.html">Mes recettes</a>
                <a href="chats.html">Mes chats</a>
        </div>
</div>
```

Créez un nouveau fichier et copiez-y le code commun. Enregistrez votre fichier sous le nom

« entete.html ».

Dans chacune des trois pages, supprimez cette partie commune de code et remplacez-la par la ligne

<!--#include file=« entete.html » --> Ceci est en fait un commentaire en HTML

L'impact de cette modification n'est pas visible dans Amaya, car c'est le serveur web qui se charge de faire inclusion et non pas le client.

Vérifiez que vous avez bien copié les fichiers modifiés dans le répertoire racine de votre site web /var/www/votre_nom. Vous devez y trouver les fichiers entete.html, index.html, chats.html, recettes.html et la feuille de style et les images.

Vérifiez que votre site web fonctionne toujours.

PARTIE 4 (Pour aller plus loin)

Rêvons que nous devenons un hébergeur professionnel. Dans ce cas nous allons devoir héberger plusieurs sites web au sein du même serveur web. Chaque site doit être accessible sous un nom différent, mais nous ne possédons qu'une adresse IP. Pour réaliser ceci avec apache, nous allons utiliser les serveurs virtuels.

Créez un répertoire /var/www/deuxième_site. Copiez-y votre site web et modifiez le titre de la page d'accueil (par exemple « Voici mon deuxième site »). Voilà tout est prêt. Il reste à configurer apache pour qu'il prenne en compte ce nouveau site.

Rendez-vous dans le répertoire /etc/apache2/sites-available et copiez le fichier default dans un fichier de nom « deuxième » par exemple. Dans ce fichier « deuxième », supprimez la première ligne (NameVirtualHost * :80), rajoutez, sous la balise <VirtualHost *>, le nom de votre nouveau site situé derrière le paramètre ServerName et modifiez toutes les références au répertoire racine de ce nouveau site (dans notre cas /var/www/votre_nom devient /var/www/deuxième_site). Enregistrez ce fichier.

Rendez-vous maintenant dans le répertoire /etc/apache2/sites-enabled et créez un lien symbolique vers le fichier /etc/apache2/sites-available/deuxième en utilisant la commande ln options -s. Ceci a pour effet d'activer ce deuxième site.

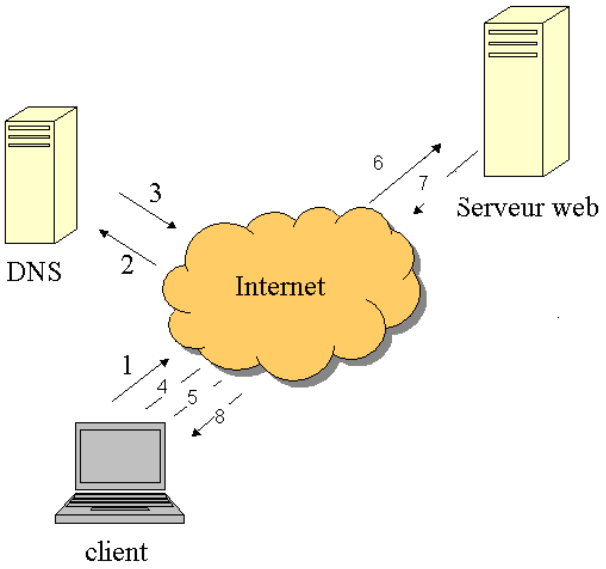
Il vous reste à mettre à jour votre fichier « hosts » ou le serveur DNS pour permettre la résolution de noms en local ou en réseau et à tester votre travail.

Nota : Si, chez vous, vous utilisez les services DynDNS pour gérer votre nom, sachez que vous avez le droit d'avoir jusqu'à six noms par compte.

PARTIE 5 Le client/serveur

Nous allons commencer par décrire ce qu'est un serveur Web et comment il s'inscrit dans le fonctionnement global d'Internet.

Derrière votre écran, avec votre navigateur préféré, vous êtes un "client" qui demande à un "serveur" de lui envoyer des données qu'il pourra interpréter (du (X)HTML par exemple). Pour savoir à quel serveur s'adresser, vous fournissez via votre navigateur une adresse (ex. http://btsinfo-carcouet.servebbs.com). L'aiguillage sur Internet ne se faisant pas directement via des adresses "alphanumérique" mais des adresses IP (ex. 82.127.58.57), la conversion doit être faite. Ceci est le rôle du serveur DNS (Domain Name System). Une fois la conversion effectuée, votre requête peut correctement être dirigée vers le serveur que vous interrogez. La requête arrive donc au serveur. Le serveur traite la requête et retourne les informations au navigateur client qui les affiche sur l'écran. Voici un petit schéma résumant ceci :



Maintenant que nous avons vu comment globalement s'organisent les interactions client/serveurs web, nous allons voir comment le serveur fait pour "répondre" au client.

Tout d'abord il nous faut énoncer qu'un serveur est en fait un programme qui tourne en tâche de fond et qui attend qu'on le sollicite. Plus techniquement on dit qu'un serveur "écoute" sur un "port" identifié. Pour que le client puisse atteindre le serveur, il lui faudra donc non seulement s'adresser à la bonne adresse mais aussi au bon port pour interroger le bon serveur sur la machine. En effet, sur une même machine (un serveur web par exemple), nous pouvons avoir (nous avons !), plusieurs serveurs qui écoutent différents ports. On peut classiquement parler du serveur HTTP qui écoute par défaut le port 80, le serveur MySQL qui écoute par défaut le port 3306, le serveur FTP écoute le port 21, etc … Bref, Un serveur web à proprement parlé est un serveur HTTP, c'est à dire un serveur répondant aux requêtes "http://…". Il comprend donc le protocole http, protocole de niveau applicatif.

Enfin pour finir, il est important de comprendre que les traitements à réaliser peuvent être répartis entre le client et le serveur.

Dans la partie 2 de ce TP, le serveur web sert la page demandée, mais le code XHTML est interprété par votre navigateur qui réalise l'affichage de la page.

Coté client	Coté serveur
Interprétation du code XHTML et affichage de la page	Recherche de la page demandée et envoi au client.

Les premières balises de votre page (X)HTML sont donc très importantes :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
```

Elles permettent de dire à votre navigateur quelles particularités, peut avoir votre document concernant sa structure et sa présentation. La ligne ci-dessus par exemple indique à votre navigateur que le format XHTML est respecté (ceci implique par exemple que les balises comme leurs attributs doivent être écrits en minuscule ou encore que vous vous êtes engagés à bien séparer la forme du fond). Pour plus de détails, voir l'article « **XHTML mieux que HTML ? »** issu du linux pratique hors série numéro 1.

Dans la partie 3 de ce TP, les traitements sont répartis entre le client et le serveur. Si le client continue simplement d'afficher après interprétation du XHTML, la page demandée. Le serveur réalise ici un traitement qu'il n'effectuait pas dans la partie 2 de ce TP. Il commence par scruter la page demandée à la recherche d'un « include » pour injecter la partie de code XHTML demandée. Comme ce traitement s'effectue coté serveur, vous ne pouvez pas visualiser l'impact de cette modification en affichant directement votre page dans un navigateur (vous avez constaté je crois que cette modification n'était pas visible dans Amaya).

De façon générale, lorsqu'on travaille avec une architecture client/serveur, les traitements sont répartis entre le client et le serveur. Dans la partie 3 de ce TP, la répartition est la suivante :

Coté client	Coté serveur
Interprétation du code XHTML et affichage de la page	Recherche de la page demandée, recherche d'un éventuel « include », construction et envoi de la page demandée.

On verra par la suite que l'utilisation de « javascript » ou de « php » vient enrichir cette répartition des traitements entre le client et le serveur.