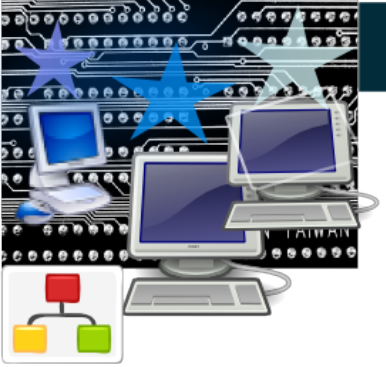


HTTP et Web

– TR2 –

Camille Coti[1]
camille.coti@lipn.univ-paris13.fr

[1] Département R&T, IUT de Villetaneuse, Université de Paris XIII

Plan du cours

Présentation

HTTP, HTML et Web

Le protocole HTTP et le langage d'affichage HTML sont liés depuis leur conception

- Serveur HTTP
- Navigateur
  - communique avec le serveur HTTP
  - affiche du HTML

L'apparition de HTTP et HTML est considérée comme la naissance du *World Wide Web*

Le World Wide Web

Ensemble de documents

- contenant des références les uns vers les autres (*hyperliens*)
- accessibles via Internet

Histoire

Débuts au CERN

But : améliorer les diffusions des données en interne au CERN

- Début du travail : 1984
- Création du WWW et proposition pour le projet HyperText : 1990
  - Premier navigateur
  - Trois concepts de base : URL, HTML HTTP

Normalisation

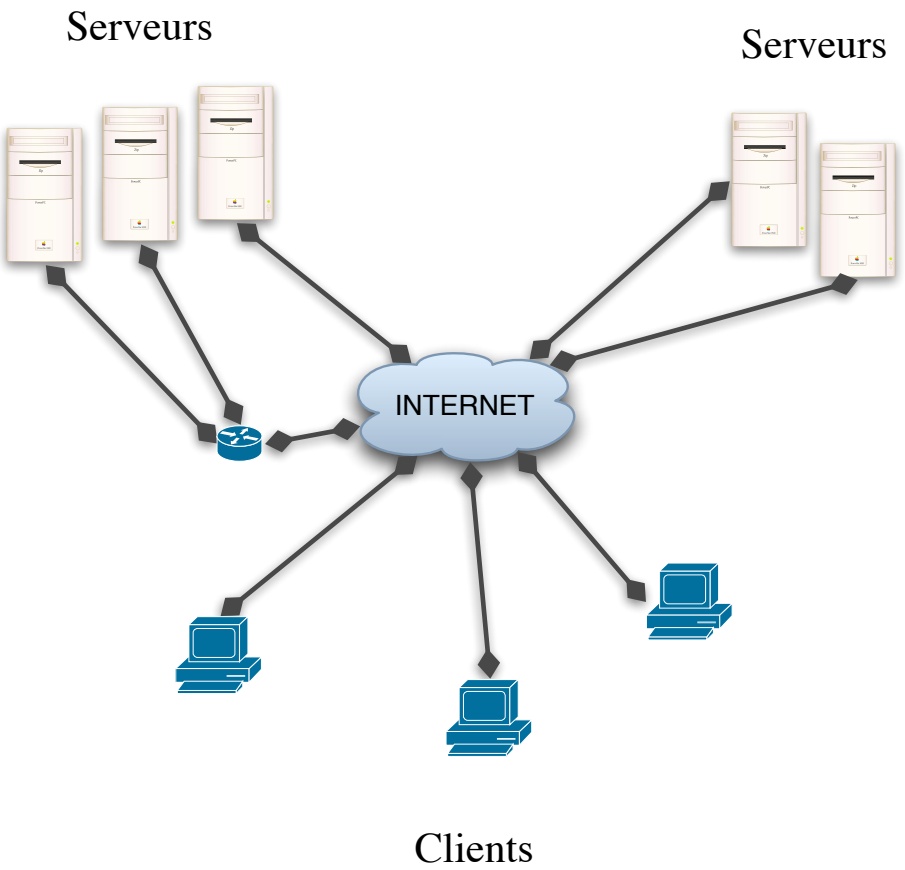
Publication de RFC :

- 1994 : RFC 1738 (notion d'URL), 2010 : RFC 5785 (notion d'URI)
- 1995 : RFC 1866 (HTML 2.0)
- 1996 : RFC 1945 (HTTP/1.0)
- 1997 puis 1999 puis 2000 : RFC 2068 puis 2616 puis 2817 (HTTP/1.1)

Consortiums :

- HTTP Working Group
- World Wide Web Consortium (W3C)

Le World Wide Web



Modèle client-serveur

- Internet relie les clients et les serveurs
  - Les documents sont sur les serveurs
  - Les clients accèdent aux serveurs et récupèrent les documents

Communication client-serveur

Protocole HTTP

- Requête-réponse

Affichage par le client

Langage d'affichage HTML

- Formattage de l'affichage

Protocole de transfert de données

- FTP ne suffit pas au transfert de documents
- Affichage dans un navigateur
- Liens entre les documents

Protocole réseau

TCP/IP généralement utilisé

- Mais pas d'hypothèse faite a priori dans le protocole
- Mode connecté
- Les données transmises sont considérées comme un flux

Transmission d'une page

Modèle client-serveur

Le serveur tourne en permanence

- Un client se connecte et envoie une requête
- Le serveur lui répond
- Fin de la conversation

Communication mode ASCII

Requête : mode texte

- Syntaxe : `COMMANDE RESSOURCE`

Réponse : mode texte

- Encodage en texte, utilisation du type MIME pour les contenus non-texte

Commandes disponibles

Principales commandes

Toutes les commandes ne sont pas forcément disponibles ! (sécurité)

- `GET` : demande d'une ressource (commande la plus utilisée)
- `POST` : ajout d'une nouvelle ressource, contient des données
- `HEAD` : obtenir des informations sur la ressource
- `OPTIONS` : obtenir des informations sur les possibilités offertes par le serveur
- `CONNECT` : utilisation d'un proxy
- `TRACE` : retourner ce que le serveur a reçu (utilisé comme diagnostic)

Commandes spécifiques

Nécessitent un accès privilégié (authentifié)

- `PUT` : remplacer une ressource sur le serveur
- `DELETE` : supprimer une ressource du serveur

Exemple : récupération d'une page

Connexion au serveur

```
$ telnet www.iutv.univ-paris13.fr 80
Trying 194.254.173.2...
Connected to www.iutv.univ-paris13.fr.
Escape character is '^]'.
```

Demande d'une page

```
GET /iutv/index.php
```

Réponse du serveur

```
<HTML> [...]
</HTML>

Connection closed by foreign host.
```

- C'est ce qui est interprêté par le navigateur

Demandes partielles

GET conditionnel

Page récupérée seulement si elle a été depuis une date donnée

- `If-Modified-Since: Sat, 29 Oct 1994 19:43:31 GMT`

Si la page n'a pas été modifiée :

- Le serveur n'envoie qu'une ligne (code 304)
- Le navigateur affiche une version en cache

Sinon :

- Le serveur envoie la dernière version

HEAD

Ne demande que l'en-tête de la page

- Permet de déterminer la date de dernière modification
- Obtient les informations d'indexation de la page
- Vérifie la validité d'une URL

Réponse du serveur

La réponse du serveur vient avec un code (3 chiffres) indicateur du résultat de la requête :

- 1xx : information
- 2xx : succès
- 3xx : redirection
- 4xx : erreur du client
- 5xx : erreur du serveur

200 : OK

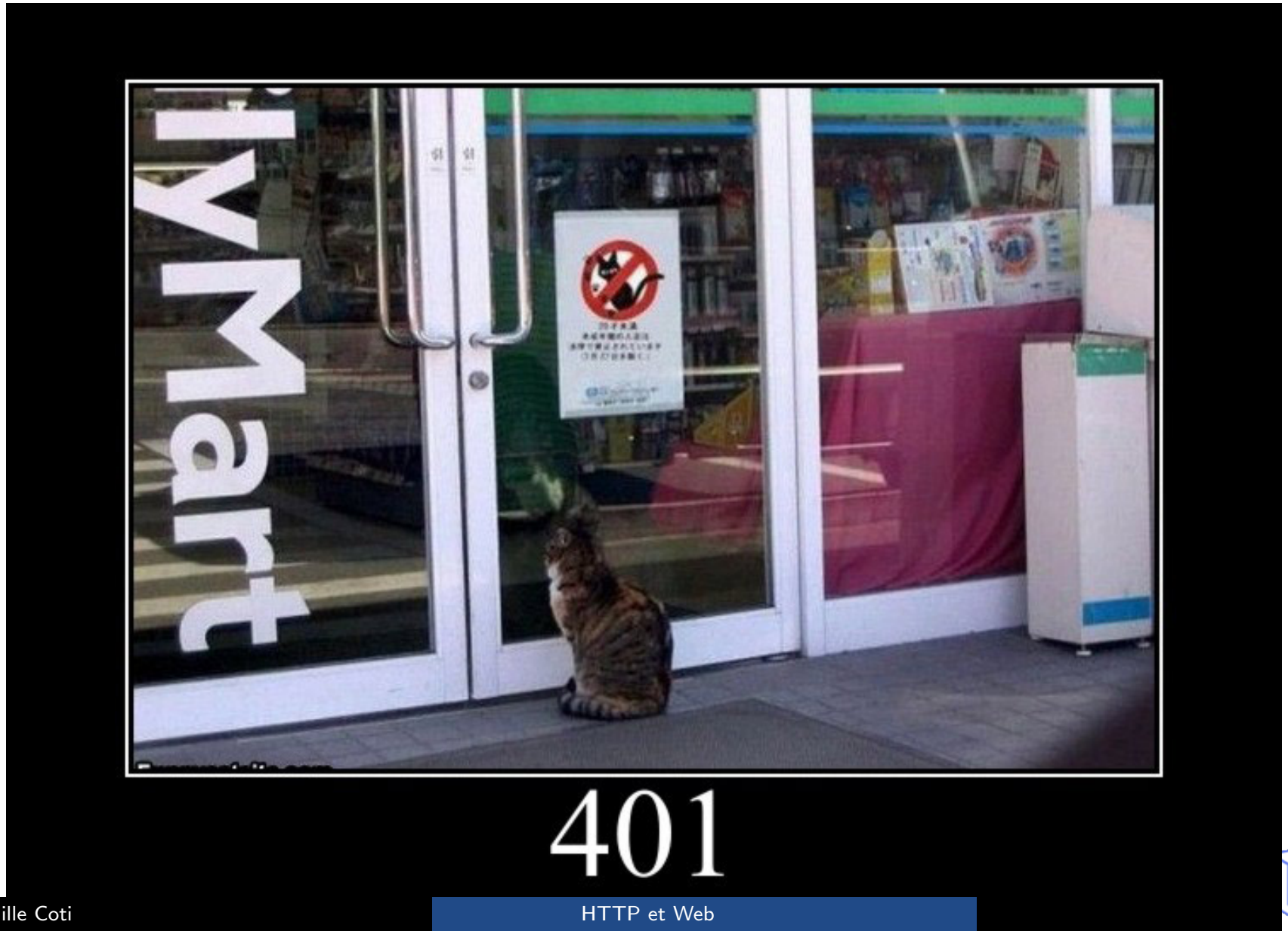
200 : requête traitée avec succès

100 : Continue

100 : attente de la suite de la requête

401 : Unauthorized

401 : authentification nécessaire pour accéder à la ressource demandée



403 : Forbidden

403 : authentification refusée

404 : Not found

404 : la ressource demandée n'a pas été trouvée

405 : Method not allowed

405 : méthode de requête non autorisée

408 : Request timeout

408 : temps d'attente d'une réponse du serveur écoulé

409 : Conflict

409 : la requête ne peut pas être traitée pour le moment (conflit de versions...)

413 : Request entity too large

413 : requête trop grosse

414 : Request IRU too long

414 : l'URI demandée est trop longue

418 : I am a teapot

418 : je suis une théière (RFC 2324 définissant le Hyper Text Coffee Pot Control Protocol)

426 : Upgrade required

426 : le client utilise une version trop ancienne du protocole

429 : Too many requests

429 : le client a envoyé trop de requêtes en un temps donné

431 : Request Header Fields Too Large

431 : l'en-tête de la requête ou un de ses champs est trop grand

450 : Blocked by Windows Parental Controls

450 : extension du contrôle parental de Microsoft

500 : Internal server error

500 : erreur côté serveur

599 : Network connect timeout error

599 : timeout côté serveur (proxy)

Le type MIME

Multipurpose Internet Mail Extensions

Standard Internet

- RFC 2045, RFC 2046, RFC 2047, RFC 2048 et RFC 2077
- Transmettre n'importe quel contenu dans de l'ASCII

Origine

L'encodage des mails est en ASCII 7 bits

- Impossibilité de transmettre des langages basés sur l'alphabet Latin nécessitant des caractères spéciaux
- Impossibilité de transmettre des langages basés sur un autre alphabet
- Impossibilité de transmettre autre chose que du texte

Utiliser un type MIME permet d'encode autre chose dans le corps ou dans l'en-tête d'un mail

- Par extension, dans toute communication basée sur ASCII

MIME : exemple dans un mail

En-tête

On spécifie la version

- `MIME-Version: 1.0`

On annonce ce qui vient dans le corps

- ```
  Content-Type: Multipart/related;
  charset="ISO-8859-1";
  type="multipart/alternative";
  boundary="------------Boundary-00=_JXS9QL80000000000000"
  ```

Séparation entre les éléments

Utilisation d'un séparateur quand le type change

- `--------------Boundary-00=_JXS9LVC00000000000000--`

Chaque élément est transmis comme une partie du mail
  ```

MIME : exemple dans un mail

Transmission d'une image

```
--------------Boundary-00=_JXS9QL80000000000000
Content-Type: image/gif;
name="ATT1.gif"
Content-Transfer-Encoding: base64
Content-ID: <D0DEE211-0A46-4E4F-AEC1-23417EEED16A>
```

Texte

L'image est terminée, on repasse à du texte

```
--------------Boundary-00=_JXS9LVC0000000000000
Content-Type: Text/Plain;
charset="ISO-8859-1"
Content-Transfer-Encoding: quoted-printable
```

HTML (pouvant être interprêté par le client mail)

```
--------------Boundary-00=_JXS9LVC0000000000000
Content-Type: Text/HTML;
charset="ISO-8859-1"
Content-Transfer-Encoding: quoted-printable
<HTML><HEAD>
```

Serveur Web

Fonctionnement

Le serveur Web écoute sur un port (généralement le port 80)

- Les clients se connectent sur ce port
  - un client demande une page
  - le serveur lui envoie cette page
  - le client se déconnecte

Exemples

- Open source : Apache HTTP Server (Apache), Tomcat, Jigsaw...
- Propriétaire : Microsoft IIS, Zeus...

Adressage des documents

Notion d'URL

- Universal Ressource Locator

L'URL contient :

- L'adresse IP ou symbolique (DNS) du domaine de la machine
- Le chemin vers le document sur ce serveur

Adressage des documents

Notion d'URL (Universal Ressource Locator)

Permet d'identifier

- Le serveur sur lequel se situe le document
- Le document lui-même

Composition d'une URL

L'URL contient :

- L'adresse IP ou symbolique (DNS) du domaine dont fait partie la machine
  - ex : univ-paris13.fr
- Le nom de la machine sur laquelle tourne le serveur
  - ex : www.iutv
- Le chemin relatif vers le document (par rapport au répertoires du serveur Web)
  - ex : /iutv/reseaux_telecommunications.php

L'URL de ce document est alors :

www.iutv.univ-paris13.fr/iutv/reseaux_telecommunications.php

Client Web

Fonctionnement

Envoi des requêtes aux serveurs

- Interprétation des URL
- Résolution DNS, localisation du serveur
- Mise en forme des requêtes HTTP

Interprétation des réponses

Affichage des documents

- Interprétation de l'HTML
- Exécution des scripts côté client
- Conformation aux standards vérifiée par les tests Acid

Si le document contient d'autres parties (images, etc)

- Émission de requêtes pour les obtenir

Exemples

FireFox, Fennec, Opera, Chrome, Safari, Konqueror, Internet Explorer...

Applications Web

Types d'applications

Applications côté serveur

- Le serveur fait le calcul
- Sa réponse dépend de ce qu'il a calculé
- Envoi de code HTML
- Exemple : PHP

Applications côté client

- Le client fait le calcul
- Exemple : Javascript, applet Java...

Comparatif

Confidentialité

- Applications côté serveur : le code exécuté est invisible du client

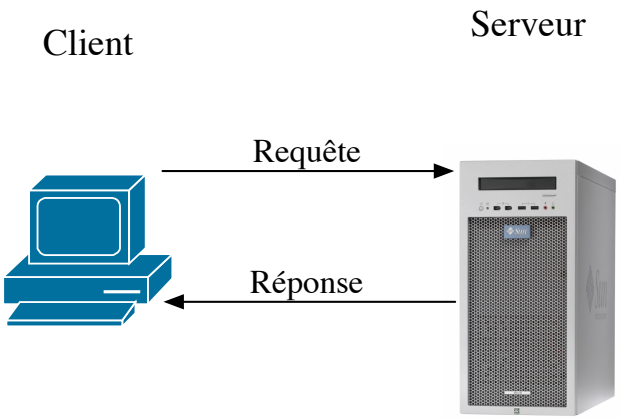
Charge

- Applications côté serveur : le client fait le travail, pas le serveuur

Illustation

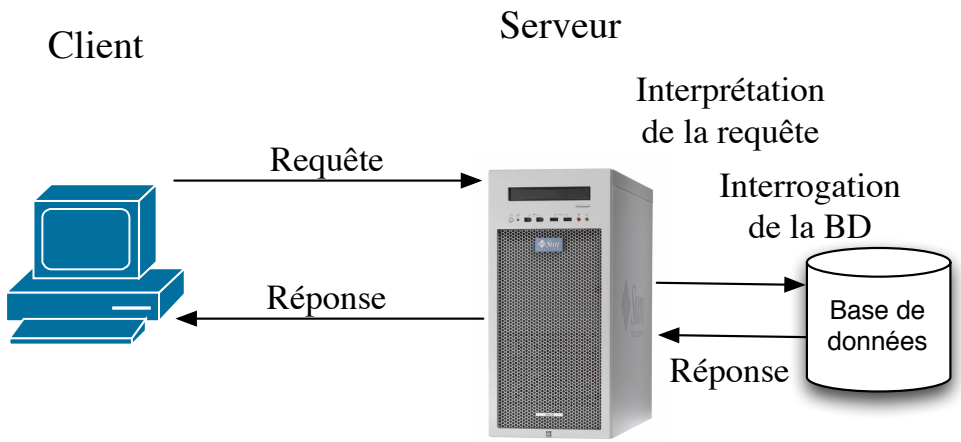
Requête simple

Exemple : affichage d'une page



Application serveur

Exemple : interrogation d'une base de données



Limites du modèle

Le client est toujours à l'initiative de l'échange

- Pas de push du serveur vers le client
- Pas de notifications d'évènements
  - Bricolage avec AJAX
  - Timer : toutes les X secondes, le client demande au serveur s'il y a quelque chose de nouveau (exemple : notifications Facebook)
- La connexion est fermée à la fin de l'échange
  - Le flux est coupé
  - HTML5 : WebSockets
  - Applets Java : connexion vers un servlet situé sur la machine serveur

Applications déployées

Soit centralisées

- tout se fait côté serveur : BD, blog, diffusion de contenu...

Soit entièrement côté client

- tout se fait côté client : animations / jeux Flash, applets Java...

Serveur Apache HTTP

Présentation

Serveur le plus répandu

- Environ 72,5% de parts de marché (décembre 2008)
- Deuxième : IIS, environ 18% (décembre 2008)

Bien intégré dans une combinaison open source

- LAMP : Linux Apache MySQL PHP

Mais *très* large compatibilité avec d'autres technologies

Modularité

Ajout de fonctionalités par modules

- Permet de proposer une très large gamme de fonctionalités
- Sans forcément les charger toutes (plus léger, failles de sécurité...)

Configuration d'Apache

Fichiers de configuration

Généralement dans `/etc/apache2/` (variable selon les versions et/ou la distribution Linux)

- httpd.conf et apache.conf

Modules

Configuration des modules

- mods-enabled/*.load
- mods-enabled/*.conf

Sites hébergés

Possibilité d'héberger plusieurs sites sur un serveur

- sites-enabled/
- sites-available/

Détermination : selon le port ou le nom d'hôte

```
<VirtualHost *:80>
```

Au niveau des communications

Possibilité de crypter les communications avec SSL ou TLS

- Permet d'éviter de faire passer des informations en clair (mots de passe, numéros de carte banquaire, de sécurité sociale...)
- Utilisation d'un certificat signé par une autorité de certification
- Port 443 par défaut
- Protocole HTTPs

Restrictions d'accès

Utilisation d'un fichier `.htaccess`

- Définition de règles par répertoire et pour les sous-répertoires
- Global (tout le serveur ou tout le site) ou local (un répertoire donné et ses sous-répertoires)
- Définit les restrictions : par adresse IP du client, par type de ficheir, authentification du client...

Traces et logs

Traces d'accès au serveur

Disponibles dans `/var/log/apache2/`

- `access.log` : toutes les requêtes arrivées au serveur

```
127.0.0.1 - - [22/Jan/2011:17:51:17 +0100] "GET
/server-status?auto HTTP/1.1" 200 633 "-" "libwww-perl/5.836"
```

- `errors.log` : toutes les erreurs

```
[Sun Jan 16 07:43:37 2011] [notice] Apache/2.2.16 (Debian)
PHP/5.3.2-2 with Suhosin-Patch configured -- resuming normal
operations
```

Autres traces

- Possibilité de mettre en place une trace par Virtual Host
- PID du serveur Apache : `/var/run/apache2.pid`

Fichiers sur le serveur

Site principal

Localisation définie dans la configuration du serveur ou du virtual host

```
/etc/apache2/sites-available/default:  DocumentRoot /var/www
```

Comptes utilisateurs

Localisation dans un répertoire particulier du répertoire utilisateur, généralement `~/public_html` ou `~/WWW` (défini dans la configuration du serveur ou du virtual host)

```
/etc/apache2/mods-available/userdir.conf:  UserDir public_html
```

Droits d'accès

Le serveur tourne généralement sous un utilisateur particulier : www-data, apache, root (peu sécurisé donc peu fréquent)...

- Les fichiers doivent être lisibles par cet utilisateur
- Les répertoires doivent être traversables par cet utilisateur
- Les scripts doivent être exécutables par cet utilisateur
- etc