

# Unix et Programmation Web

## Cours 4

[kn@lri.fr](mailto:kn@lri.fr)

<http://www.lri.fr/~kn>

# Plan

## 1 Réseaux, TCP/IP ✓

## 2 Web et HTML

### 2.1 Internet et ses services

### 2.2 Fonctionnement du Web

### 2.3 Adressage des documents Web

### 2.4 Le protocole HTTP

### 2.5 HTML, le format des documents

### 2.6 Encodage des caractères (UTF-8) (digression)

# Bref historique d'Internet (1/2)

- 1959-1968 :** ARPA (*Advanced Research Project Agency*) crée un réseau de quelques machines capable de résister à une attaque.
- 1969 :** ARPANET. Interconnexion des ordinateurs centraux des grandes universités et institutions américaines. Première utilisation du concept de paquet d'information.
- 1970-1982 :** Interconnexion avec la Norvège et le Royaume-Uni.
- 1982 :** Passage au protocole TCP/IP. Naissance de l'Internet actuel.

# Bref historique d'Internet (2/2)

- 1986 :** « Autoroutes de l'information ». Des super-ordinateurs et les premières connexions à fibres optiques sont utilisées pour accélérer le débit d'Internet.
- 1987-1992 :** Apparition des premiers fournisseurs d'accès. Les entreprises se connectent.
- 1993-2000 :** Avènement du Web. Démocratisation du haut-débit (vers 2000 pour la France).
- 2000-présent :** Explosion des services en ligne, arrivée des réseaux sociaux, internet mobile, *Cloud* (stockage et calcul mutualisés accessible depuis internet).

# Internet

- Ensemble de logiciels et protocoles basés sur **TCP/IP**
- Modèle Client/Serveur
- Un serveur fournit un service:
  - courriel
  - transfert de fichier (ftp)
  - connexion à distance (ssh)
  - Web (http)
- Plusieurs services peuvent être actifs sur la même machine (serveur). Un **port (identifiant numérique)** est associé à chaque service. Sur Internet, un service est identifié par:
  - L'adresse IP de la machine sur lequel il fonctionne
  - Le numéro de port sur lequel le programme attend les connexions

# Exemples de services

<b>Service</b>	<b>Protocole</b>	<b>Port</b>	<b>Description</b>
<b>ftp</b>	<b>File Transfer Protocol</b>	<b>20,21</b>	<b>Transfert de fichiers</b>
<b>telnet</b>	<b>Network Virtual Terminal</b>	<b>23</b>	<b>Shell à distance</b>
<b>ssh</b>	<b>Secure Shell</b>	<b>22</b>	<b>Shell à distance crypté</b>
<b>mail</b>	<b>Simple Mail Transfer Protocol</b>	<b>25</b>	<b>Envoi de mail</b>
<b>pop</b>	<b>Post Office Protocol</b>	<b>110</b>	<b>Récupération de mail</b>
<b>imap</b>	<b>Internet Message Access Protocol</b>	<b>143</b>	<b>Synchronisation de mails</b>
<b>nslookup</b>	<b>Domain Name System</b>	<b>42</b>	<b>Serveur de noms</b>
<b>http</b>	<b>Hyper Text Transfer Protocol</b>	<b>80</b>	<b>Web</b>

# World Wide Web (1/2)

- Service de distribution de page **hypertexte**
- Une page **hypertexte** contient des références immédiatement accessibles à d'autres pages (pointeurs ou **liens hypertextes**)
- Les pages sont décrites dans le langage **HTML** (HyperText Markup Language)
- Architecture client/serveur:
  - Les pages sont stockées sur le serveur
  - Les pages sont envoyées au client (navigateur Web) qui en assure le rendu
- Utilise le protocole **HTTP** pour les échanges entre le client et le serveur

# World Wide Web (2/2)

## Concepts clé:

**URL :** localisation d'une page Web (« adresse de la page »)

**HTTP :** protocole de communication entre un client et un serveur Web

**HTML :** langage de description des pages Web

## Évolutions récentes (Web 2.0, internet mobile, *Cloud*, ...)

- Standardisation du contenu multimédia (images, vidéos et sons en *streaming*)
- Contenu interactif avancé (stockage de fichier coté client, rendu 3D, ...)
- Uniformisation de nombreuses extensions *ad-hoc*: HTML5

# Plan

## 1 Réseaux, TCP/IP ✓

## 2 Web et HTML

### 2.1 Internet et ses services ✓

### 2.2 Fonctionnement du Web

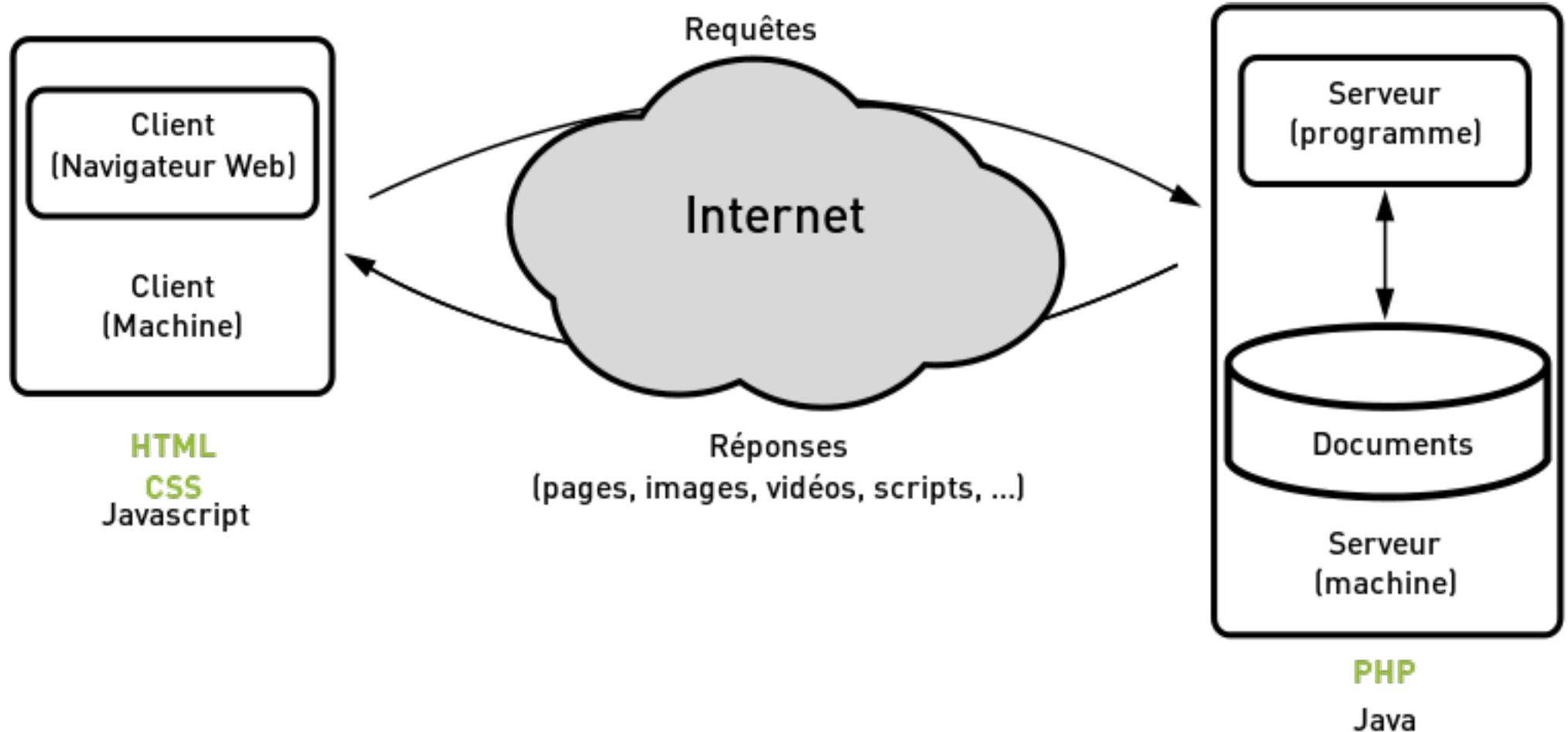
### 2.3 Adressage des documents Web

### 2.4 Le protocole HTTP

### 2.5 HTML, le format des documents

### 2.6 Encodage des caractères (UTF-8) (digression)

# Fonctionnement du Web



# Côté client

## Le navigateur :

- Analyse l'URL demandée
- Obtient l'adresse IP auprès du serveur DNS
- Établit une connexion (potentiellement sécurisée) avec le serveur
- Envoie une **requête HTTP** au serveur
- Récupère la page envoyée par le serveur dans sa **réponse**
- Analyse la page et récupère les éléments référencés : images, sons, ...
- Effectue le traitement du code client
- Met en forme le contenu et l'affiche dans la fenêtre

# Côté serveur

- Un *listener* (*thread* particulier) attend les connexions sur un port par défaut (80 dans le cas de HTTP)
- À chaque nouvelle connexion, le *listener* crée un *thread* de traitement et se remet en attente
- Le *thread* de traitement vérifie la validité de la requête :
  - le document demandé existe ?
  - le client est autorisé à accéder au document ?
  - ...
- Le *thread* de traitement répond à la requête :
  - Exécution de code côté serveur, récupération de données dans une BD, ...
  - Envoi de la page au client

# Plan

## 1 Réseaux, TCP/IP ✓

## 2 Web et HTML

### 2.1 Internet et ses services ✓

### 2.2 Fonctionnement du Web ✓

### 2.3 Adressage des documents Web

### 2.4 Le protocole HTTP

### 2.5 HTML, le format des documents

### 2.6 Encodage des caractères (UTF-8) (digression)

# Adressage des documents Web (1/3)

**URL** : *Uniform Resource Locator* identifie un document sur internet

Une URL se décompose en 3 partie

- **protocole** (comment ?)
- **adresse** (où ?)
- **document** (quoi ?)

Syntaxe (simplifiée) :

`protocole://adresse/document`

Exemple :

`http://www.lri.fr/~kn/teach_fr.html`

# Adressage des documents Web (2/3)

On peut aussi préciser **un numéro de port**, des paramètres et un **emplacement** :

`protocole://adresse:port/document?p1=v1&p2=v2#empl`

**Exemple :**

`http://www.youtube.com:80/results?search_query=tbbt#search-results`

**Le serveur utilise les paramètres passés par le client dans l'URL pour *calculer* le contenu de la page (changer la chaîne «**

tbbt

**» ci-dessus et essayer)**

# Adressage des documents Web (3/3)

La **racine** d'un site Web (ex:

`http://www.lri.fr/`

) correspond à un répertoire sur le disque du serveur (ex:

`/var/www`

). Le fichier

`http://www.lri.fr/index.html`

se trouve à l'emplacement

`/var/www/index.html`

Le serveur Web peut aussi effectuer des **réécritures d'adresses** :

`http://www.lri.fr/~kn/index.html`

devient

`/home/kn/public_html/index.html`

# Plan

## 1 Réseaux, TCP/IP ✓

## 2 Web et HTML

### 2.1 Internet et ses services ✓

### 2.2 Fonctionnement du Web ✓

### 2.3 Adressage des documents Web ✓

### 2.4 Le protocole HTTP

### 2.5 HTML, le format des documents

### 2.6 Encodage des caractères (UTF-8) (digression)

# Caractéristiques du protocole HTTP

- **Sans connexion permanente:**
  - Le client se connecte au serveur, envoie sa requête, se déconnecte
  - Le serveur se connecte au client, envoie sa réponse, se déconnecte
- **Indépendant du contenu :** permet d'envoyer des documents (hyper) texte, du son, des images, ...
- **Sans état:** chaque paire requête/réponse est indépendante (le serveur ne maintient pas d'information sur le client entre les requêtes)
- **Protocole en mode texte**

# Format des messages HTTP

Les messages ont la forme suivante

- Ligne initiale CR LF
- zéro ou plusieurs lignes d'option CR LF
- CR LF
- Corp du message (document envoyé, paramètres de la requête, ...)
- **Requête** la première ligne contient un nom de **méthode** (GET, POST, HEAD, ...), le paramètre de la méthode et la version du protocole
- **Réponse** la version du protocole, le code de la réponse (200, 404, 403, ...) et un message informatif

# Démo

# Plan

## 1 Réseaux, TCP/IP ✓

## 2 Web et HTML

2.1 Internet et ses services ✓

2.2 Fonctionnement du Web ✓

2.3 Adressage des documents Web ✓

2.4 Le protocole HTTP ✓

2.5 HTML, le format des documents

2.6 Encodage des caractères (UTF-8) (digression)

# HTML

**HyperText Markup Language** : langage de mise en forme de documents hypertextes (texte + liens vers d'autres documents). Développé au CERN en 1989.

**1991** : premier navigateur en mode texte

**1993** : premier navigateur graphique (mosaic) développé au NCSA (National Center for Supercomputing Applications)

# Document HTML

- est un document **semi-structuré**
- dont la structure est donnée par des **balises**

## Exemple

## Rendu par défaut

Un texte <code>&lt;b&gt;en gras&lt;/b&gt;</code>	Un texte <b>en gras</b>
<code>&lt;a href="http://www.u-psud.fr"&gt;Un lien&lt;/a&gt;</code>	<a href="http://www.u-psud.fr">Un lien</a>
<code>&lt;ul &gt;   &lt;li&gt;Premièrement&lt;/li&gt;   &lt;li&gt;Deuxièmement&lt;/li&gt; &lt;/ul&gt;</code>	<ul style="list-style-type: none"><li>■ Premièrement</li><li>■ Deuxièmement</li></ul>

On dit que `<toto>` est une balise **ouvrante** et `</toto>` une balise **fermante**. On peut écrire `<toto/>` comme raccourci pour `<toto></toto>`.

# Historique du langage HTML

- 1973** : GML, Generalised Markup Language développé chez IBM. Introduction de la notion de balise.
- 1980** : SGML, Standardised GML, adopté par l'ISO
- 1989** : HTML, basé sur SGML. Plusieurs entreprises (microsoft, netscape, ... ) interprètent le standard de manière différente
- 1996** : XML, eXtensible Markup Language norme pour les documents semi-structurés (SGML simplifié)
- 2000** : XHTML, version de HTML suivant les conventions XML
- 2008** : Première proposition pour le nouveau standard, HTML5
- 2014** : Standardisation de HTML5

# XHTML vs HTML

On utilise XHTML dans le cours. Différences avec HTML:

- Les balises sont **bien parenthésées** (<a> <c> </c> </b> est interdit)
- Les balises sont en minuscules

Les avantages sont les suivants

- HTML autorise les mélanges majuscule/minuscule, de ne pas fermer certaines balise ... Les navigateurs corrigent ces erreurs de manières **différentes**
- Le document est **structuré** comme un programme informatique (les balises ouvrantes/fermantes correspondent à { et }). Plus simple à débbuger.

# Rôle d'(X)HTML

Séparer la **structure** du document de son **rendu**. La structure donne une **sémantique** au document :

- ceci est un titre
- ceci est un paragraphe
- ceci est un ensemble de caractères importants

Cela permet au navigateur d'assurer un rendu en fonction de la sémantique. Il existe différents types de rendus:

- graphique interactif (Chrome, Firefox, Internet Explorer, ...)
- texte interactif (Lynx, navigateur en mode texte)
- graphique statique (par ex: sur livre électronique)
- rendu sur papier
- graphique pour petit écran (terminal mobile)

# Exemple de document

**(liste des balises données sur la feuille de TD 4!)**

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" >
  <head>
    <title>Un titre</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  </head>
  <body>
    <h1>Titre de section</h1>
    <p> premier paragraphe de texte. On met
    un <a href="http://www.lri.fr">lien</a> ici.
    </p>
    <!-- on peut aussi mettre des commentaires -->
  </body>
</html>
```

# Structure d'un document XHTML

Pour être *valide* un document XHTML contient **au moins** les balises suivantes :

- Une balise `html` qui est la **racine** (elle englobe toutes les autres balises). La balise `html` contient deux balises filles: `head` et `body`
- La balise `head` représente l'en-tête du document. Elle peut contenir diverses informations (feuilles de styles, titre, encodage de caractères, ...). La seule balise **obligatoire** dans `head` est le titre (`title`). C'est le texte qui est affiché dans la barre de fenêtre du navigateur ou dans l'onglet.
- la balise `body` représente le contenu de la page. On y trouve diverses balises (`div`, `p`, `table`, ...) qui formatent le contenu de la page

# Plan

## 1 Réseaux, TCP/IP ✓

## 2 Web et HTML

2.1 Internet et ses services ✓

2.2 Fonctionnement du Web ✓

2.3 Adressage des documents Web ✓

2.4 Le protocole HTTP ✓

2.5 HTML, le format des documents ✓

2.6 Encodage des caractères (UTF-8) (digression)

# Représentation des textes

**Avant de représenter des documents complexes, on s'intéresse aux textes (sans structure particulière)**

**Problématique: comment représenter du texte réaliste ?**

**Exemple de texte réaliste:**

**" و عليكم السلام,Здравей,¡Hola!, 你好,Góðan daginn,... "**

# Historiquement...

**Encodage 1 caractère = 1 octet (8 bits) :**

- **Encodage ASCII sur 7 bits (128 caractères)**
- **ASCII étendu 8 bits (256 caractères, dont 128 de « symboles »)**
- **Latin 1 : ASCII 7 bits + 128 caractères « ouest-européens » (lettres accentuées française, italienne, ...)**
- **Latin 2 : ASCII 7 bits + 128 caractères « est-européens » (Serbe, Hongrois, Croate, Tchèque, ...)**
- **Latin 3 : ASCII 7 bits + 128 caractères turques, maltais, esperanto,**
- **Latin 4 : ASCII 7 bits + 128 caractères islandais, lituanien, ...**
- **...**
- **Latin 15 : Latin 1 avec 4 caractères « inutiles » remplacés (par exemple pour « € » à la place de « ¤ »)**

# ... et pendant ce temps là, ailleurs dans le monde

## Encodage multi-octets:

- Encodages spécifiques pour le Chinois (Big5, GB, ...)
- Encodages spécifiques pour le Japonais (Shift-JIS, EUC, ...)

Impossibilité de mettre plusieurs « alphabets » dans un même texte

Chaque logiciel « interprétait » les séquences d'octet de manière prédéfinie

# UTF-8

## Universal (Character Set) Transformation Format 8 bit

- Encodage à taille variable « universel » (contient tous les alphabets connus)
- Un organisme (ISO) donne un code à chaque symbole
- Compatible avec ASCII 7 bits

### Encodage

Nombre d'octets	Octet 1	Octet 2	Octet 3	Octet 4	Octet 5	Octet 6
1	0xxxxxxx					
2	110xxxxxx	10xxxxxx				
3	1110xxxxx	10xxxxxx	10xxxxxx			
4	11110xxxx	10xxxxxx	10xxxxxx	10xxxxxx		
5	111110xxx	10xxxxxx	10xxxxxx	10xxxxxx	10xxxxxx	
6	1111110xx	10xxxxxx	10xxxxxx	10xxxxxx	10xxxxxx	10xxxxxx

## Exemples

A  $\rightarrow 65_{10} \rightarrow 0100\ 1010_2$  (représenté sur un seul octet)

Ë  $\rightarrow 7877_{10} \rightarrow 0001\ 1110\ 1100\ 0101_2$  (représenté 3 octets) :

11100001 1011 10 11 1000 0101  $\equiv$  225 187 133

  $\rightarrow 128053_{10} \rightarrow \dots \equiv 240\ 237\ 220\ 181$

### Avantages

- compatible ASCII 7 bits (d'anciens documents texte en anglais sont toujours lisibles)
- pas d'espace gaspillé (à l'inverse d'UTF-32 ou tous les caractères font 32 bits)

### Inconvénients

- Caractères à taille variable: il faut parcourir le texte pour trouver le n<sup>ème</sup> caractère
- Les vieux logiciels doivent être adaptés