LES TECHNOLOGIES DU WEB

1ère Partie : ***Introduction au Web***

1- Introduction à l'Hypertexte

2- Présentation du protocole HTTP

3- Principes de bases des CGI

4- Présentation du WEB2 (AJAX)

2ème Partie : ***Présentation de HTML & XHTML***

3ème Partie : ***Présentation de Javascript***

4ème Partie : ***Introduction à PHP***

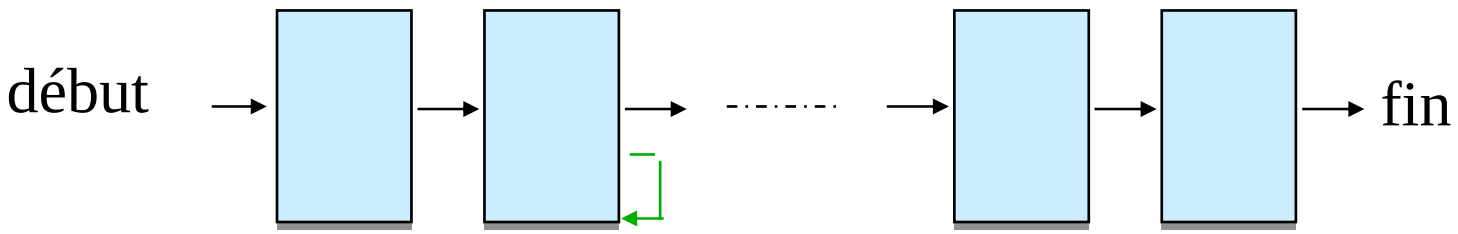
5ème Partie : ***Introduction à XML & XSLT***

LE WEB - *Introduction (1)*

Introduction : du livre à l'**hypertexte**

Une organisation essentiellement séquentielle

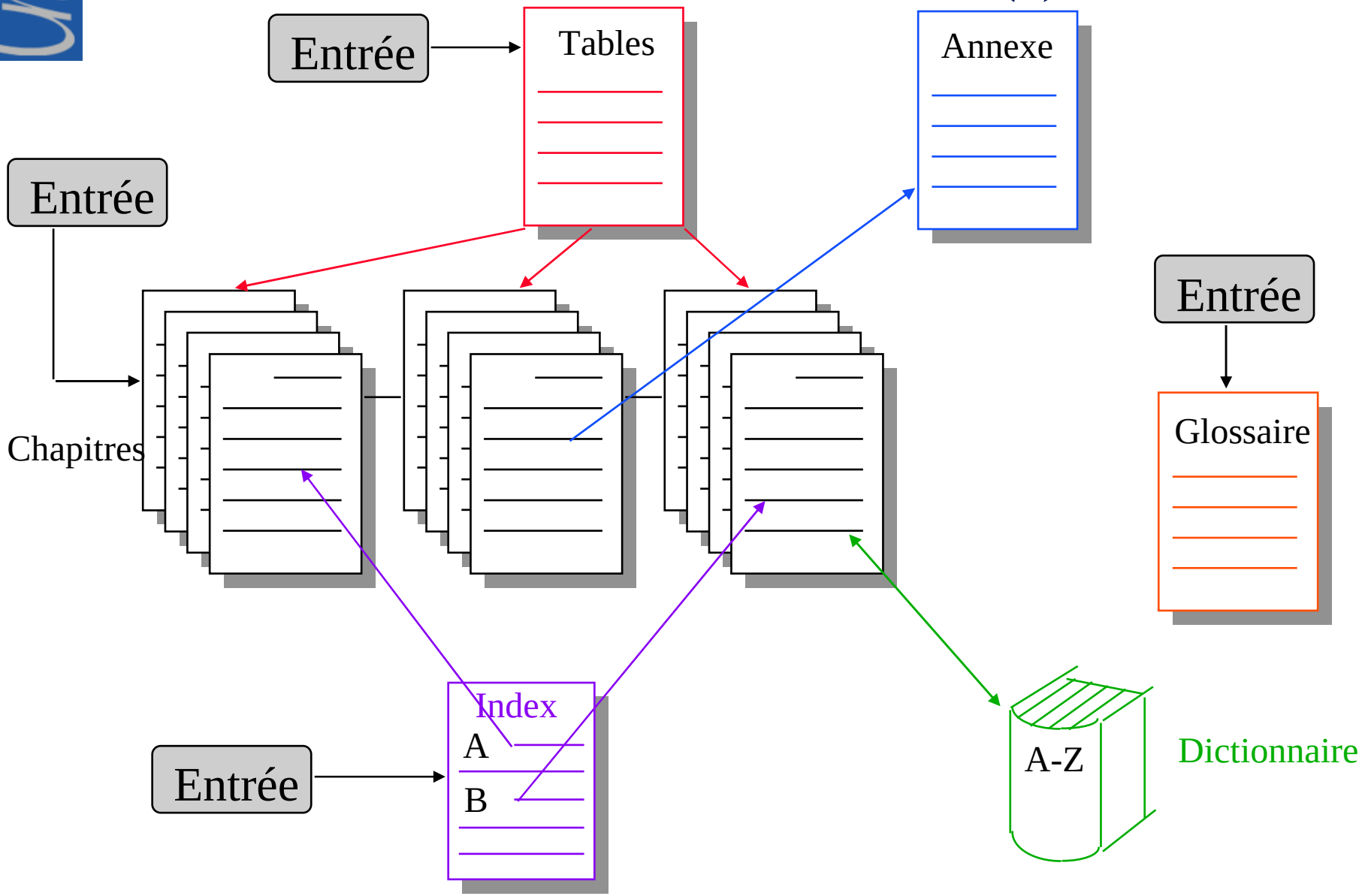
(seul déroutement : la note de bas de page, à priori, on lit dans l'ordre...



Un journal: rôle de la première page, sommaire…

une documentation technique: organisation arborescente, avec accès variés

LE WEB - *Introduction (2)*

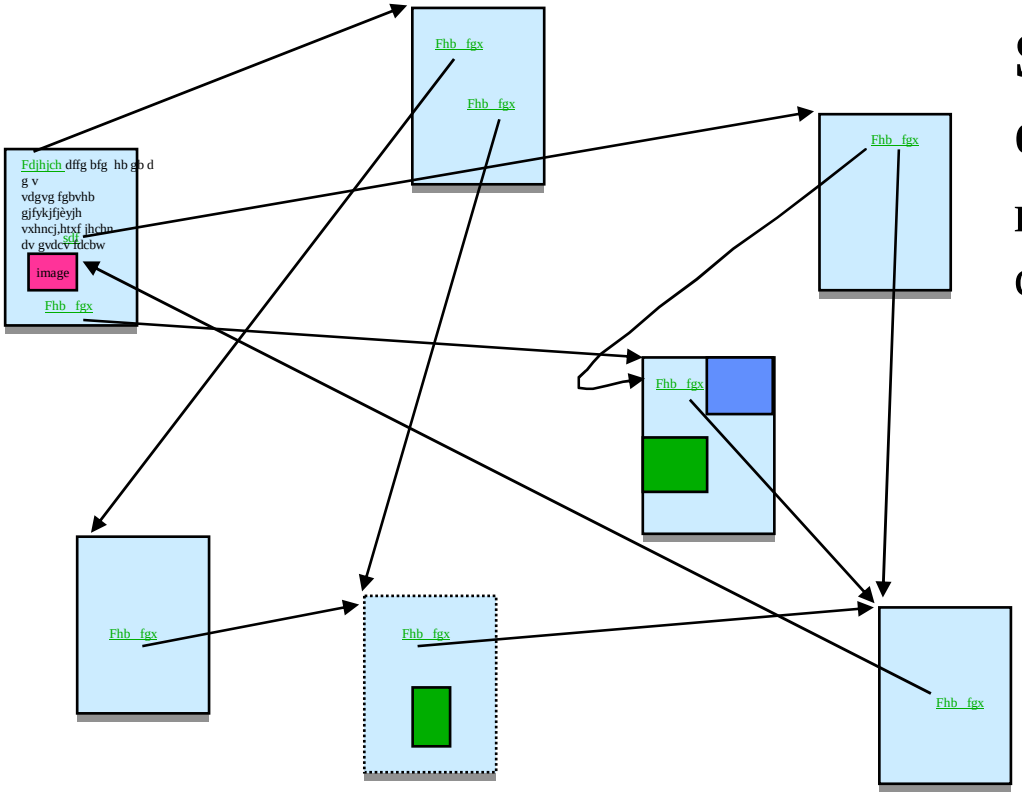


LE WEB - *Introduction (3)*

D'une façon générale, on peut dire qu'un hypertexte est un document numérique constitué de nœuds d'informations reliables entre eux par des liens.
Les définitions sont multiples car le mot désigne à la fois:

- un concept : l'ensemble des potentialités d'association de divers nœuds d'informations, association réalisée par des liens interactifs,
- un outil informatique, un logiciel : le générateur d'hypertexte, qui permet de produire aussi bien les nœuds que les liens,
- un produit : le document hypertextuel, dans lequel le lecteur/utilisateur pourra naviguer au travers des nœuds d'informations, et construire ses propres liens,
- chaque lien est mis en relation à partir d'un point d'ancrage (une ancre est un objet informatique de type pointeur, adresse …).

LE WEB - *Introduction (4)*



Stockage et représentation des connaissances :

réseaux sémantiques, graphes conceptuels,....

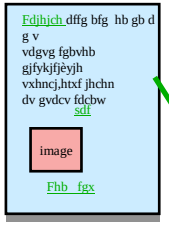
Hypertexte :

liens entre concepts

Toile d'araignée mondiale : World Wide Web

Un **document** : une **adresse : URL** "Uniform Ressource Locator"

LE WEB - *Architecture client-serveur*

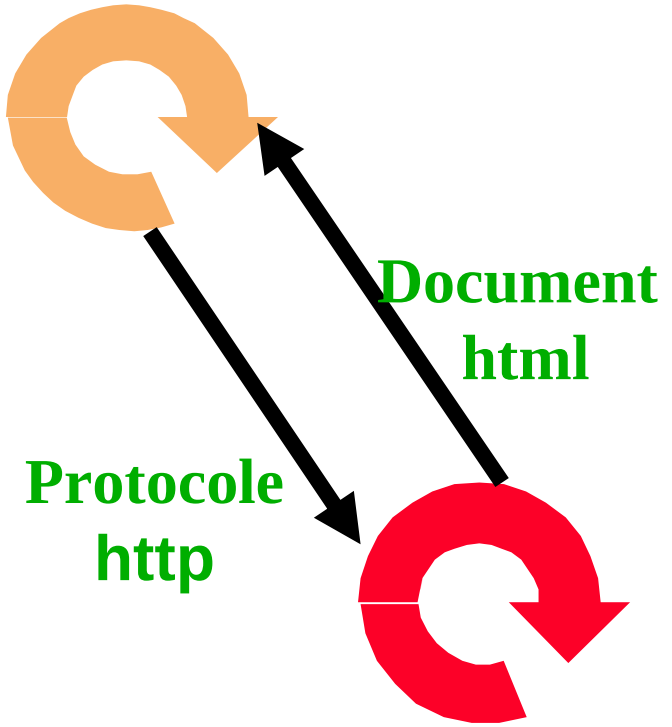


Un document:
du texte structuré,
des images fixes ou animés,
du son,
de la vidéo,
des liens vers d'autres documents
ou lien interne
ou lien vers programmes
et encore d'autres choses
(comportement, boutons, zones a remplir,…)

image

Autre lien

Le programme client (browser)
demande un document
situé sur un serveur (httpd)



LE WEB - *Les standards*

Service	Protocole	Langage
• FTP	FTP (File Transfert Protocol)	
• Messagerie	SMTP (Simple Mail Transfert Protocol)	
• News	NNTP (Network News Transfert Protocol)	
• WWW	HTTP (HyperText Transfert Protocol)	HTML (HyperText Markup Language)

- **D'autres standards :**

  SGML (XML) , Acrobat (Adobe), VRML (3D)

  et MIME (Multipurpose Internet Mail Extension)

LE WEB - *Le protocole HTTP*

- **HyperText Transfert Protocol : définit comment le browser envoie une requête et comment le serveur Web répond.**
  - **HTTP : au dessus de TCP/IP (transport fiable)**
  - **1990 : HTTP/0.9:** établissement de connexions, messages de type requête ou réponse en retour. Une réponse est un flot de caractères ASCII (méthode GET + réponse).
  - **HTTP 1.0 (RFC 1945)**
    - **Permet de transférer des messages avec des en-têtes décrivant le contenu du message en utilisant un codage de type MIME**
  - **HTTP 1.1 (RFC 2068 & 2616)**

LE WEB – *MIME (1)*

- **Permet de coder à l'intérieur d'un message plusieurs types de données à caractère multimédia** (graphique, voix, etc …)
- **Pour la description du type d'objet transféré et son traitement à la réception**
- **Nombre limité de média : types divisé en sous types**
  ex : video/mpeg, video/quicktime
- **Coté serveur :** objet relié à un type/sous-type MIME : c' est l'extension du nom de fichier qui définit l'étiquette
- **Coté client** : le type MIME récupéré est comparé avec la liste des types MIME connus du client:
  - certains sont traités directement ex: text/html,
  - d'autres par des logiciels plug-in ,
  - d'autres par des applications externes,
  - enfin, le client peut simplement sauvegarder sur le disque l'objet.

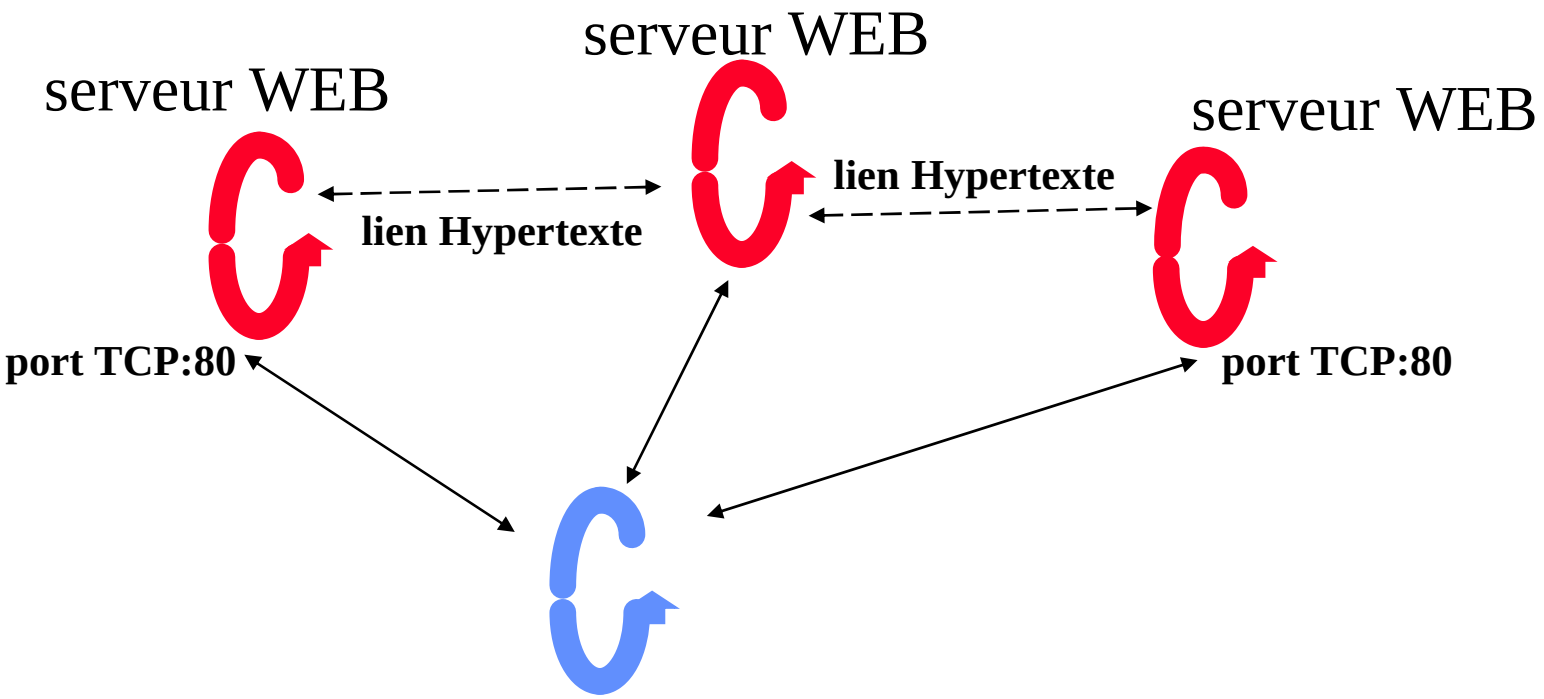
LE WEB – *MIME (2)*

Les MIME contents-types se définissent par un type et un sous type

Exemple de fichier de configuration serveur Machhtp (extrait)

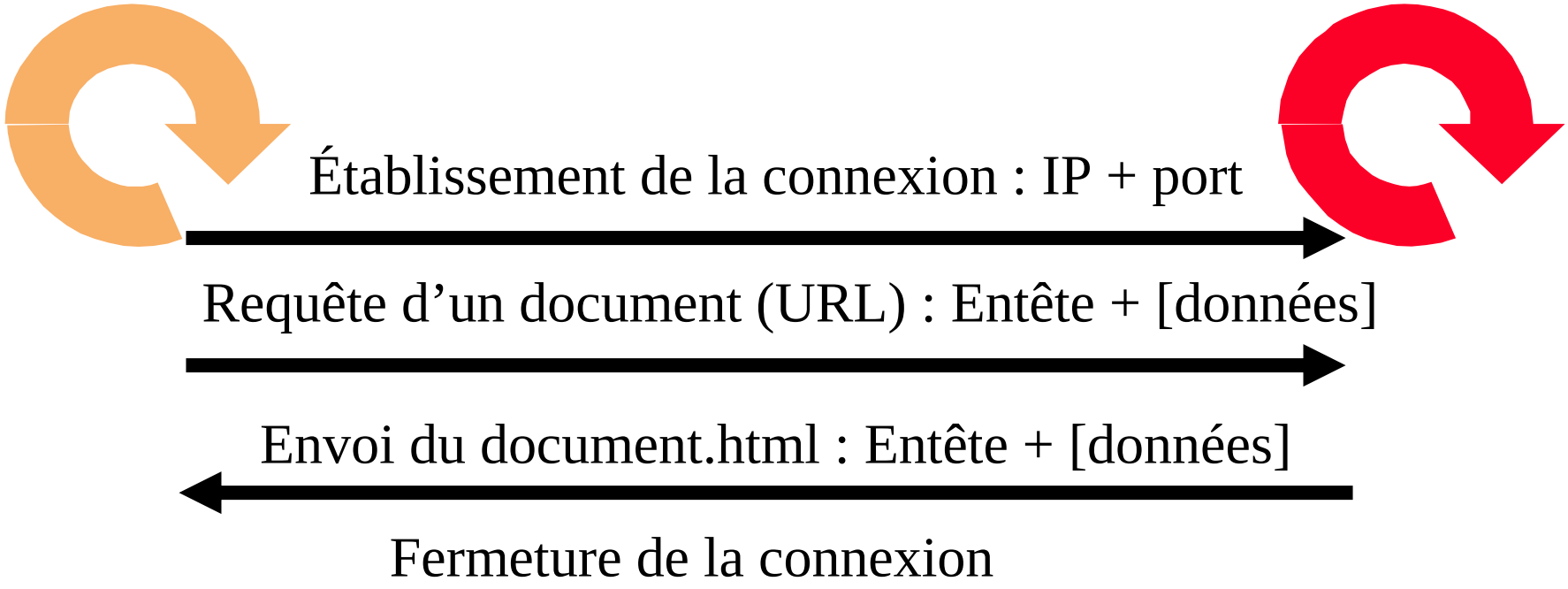
```
TEXT    .HTML TEXT * text/html
BINARY  .GIF  GIFf * image/gif
CGI .CGI APPL * text/html
ACGI .ACGI APPL * text/html
SCRIPT  .SCRIPT TEXT * text/html
SCRIPT  * TEXT ToyS text/html
APPL    .EXE APPL * text/html
TEXT    .TEXT TEXT * text/plain
TEXT    .TXT TEXT * text/plain
TEXT   .HQX TEXT * application/mac-binhex40
BINARY  .JPG JPEG * image/jpeg
BINARY  .JPEG JPEG * image/jpeg
BINARY  .PICT PICT * image/pict
BINARY .AU * * audio/basic
BINARY .AIFF * * audio/x-aiff
BINARY .XBM * * image/x-xbm
BINARY  .MOV MOOV * video/quicktime
BINARY .MPEG MPEG * video/mpeg
BINARY .XL XLS3 * application/excel
BINARY .SIT SITD * application/x-stuffit
BINARY .PDF PDF * application/pdf
BINARY .DIR TEXT * application/x-director
BINARY .DCR TEXT * application/x-director
BINARY .DXR TEXT * application/x-director
```

LE WEB - *HTTP/1.0 - les bases*



client WEB (browser) : **demandant un document hypertexte pouvant contenir d'autres liens hypertexte ou des liens sur des sessions TELNET ou FTP,...**

LE WEB - *HTTP/1.0 - schéma de base*



Si le document comporte des images, un nouveau transfert complet est à faire pour chaque image….sauf si….connexion persistante

LE WEB - *HTTP/1.1 - Introduction*

- **RFC 2068 puis 2616 (Janvier 1997)**
- **proposé comme standard Internet**
- **prise en compte des effets de hiérarchie de cache, proxy et connexion persistante**
- **3 niveaux de recommandations**
  - ✓ **MUST: obligatoire**
  - ✓ **SHOULD**
  - ✓ **MAY**
- **méthodes nouvelles: PUT, DELETE, TRACE,OPTIONS**
- Entête **host:**www.host.com:80 (obligatoire si le client utilise http1.1)

LE WEB - *HTTP/1.1 - Définitions (1)*

origin server : **serveur WWW sur lequel réside la ressource**

user agent : **qui envoie les requêtes(browser, éditeur, spider,..)**

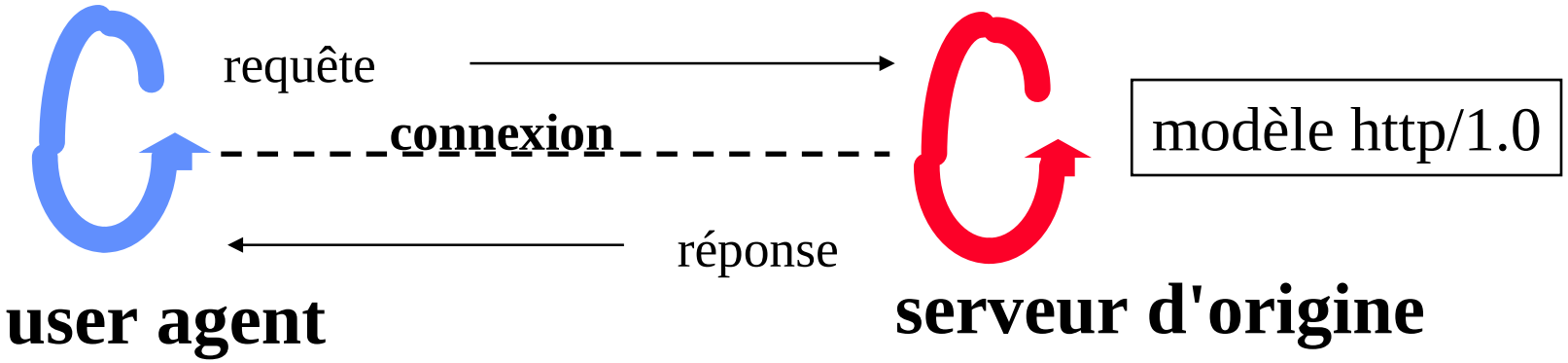
proxy : **programme intermédiaire(relais)qui agit à la fois comme serveur et client. Les requêtes sont servies soit directement (interne) soit passées à d'autres serveurs**

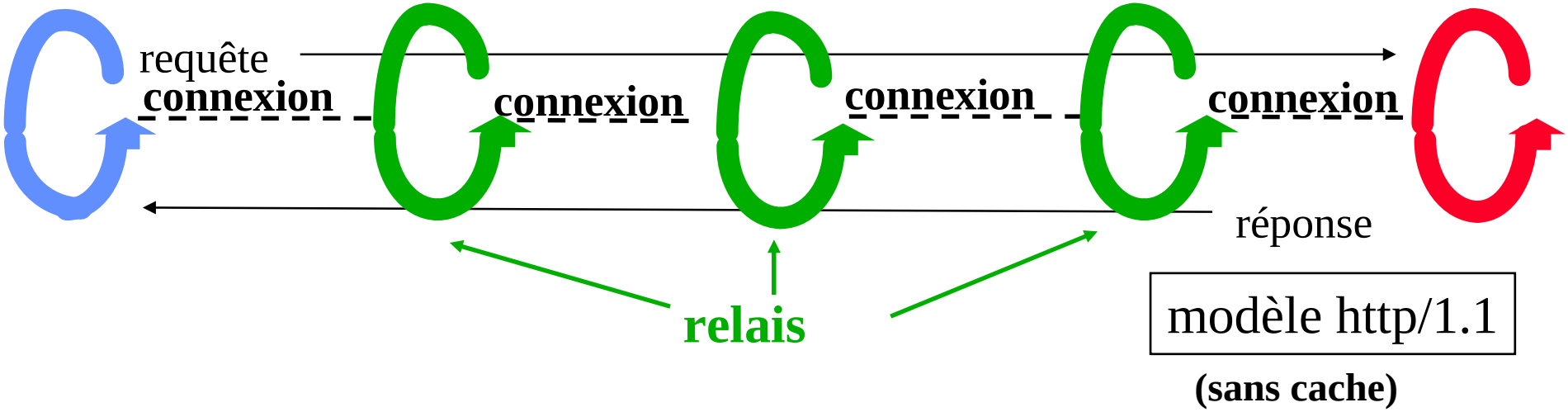
gateway : **relais qui agit comme serveur à la place d'un autre. Le client faisant la requête n'a pas à être averti qu'il communique avec la gateway (différence de proxy)**

tunnel : **agit comme un relais aveugle entre deux connexions. une fois activé (initié par une requête http, il ne doit plus être considéré comme un élément de la communication)**

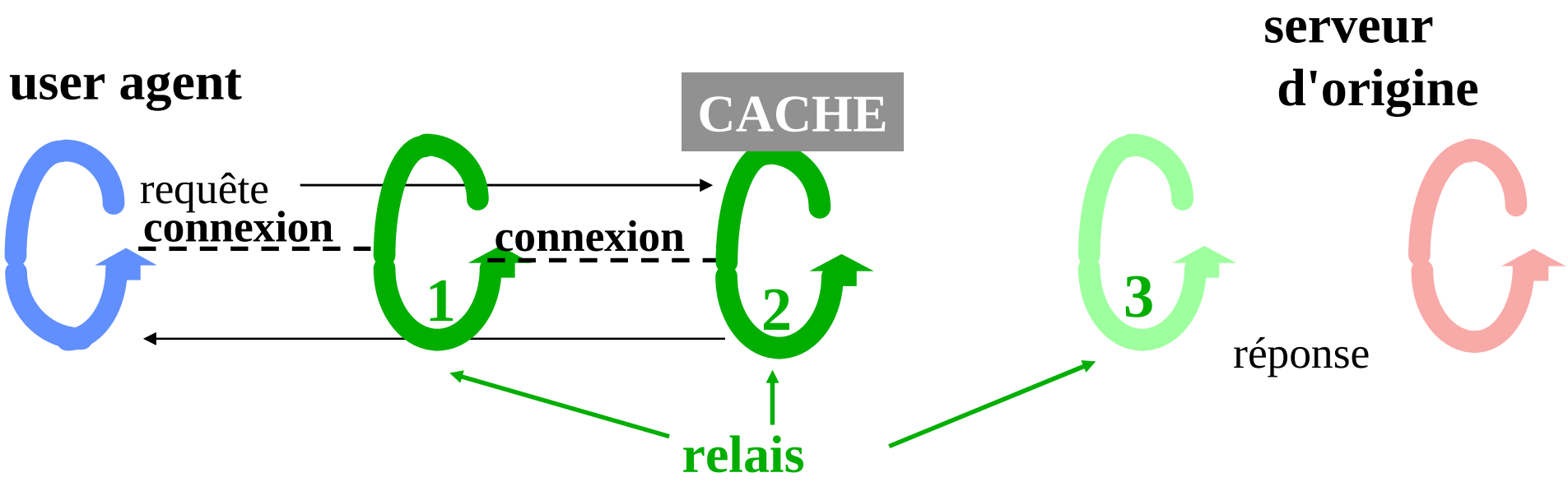
cache : **stock local de messages réponses et système de gestion de ceux-ci. tout client ou serveur peut être cache (sauf un tunnel)**

LE WEB - *HTTP/1.1 - Définitions (2)*





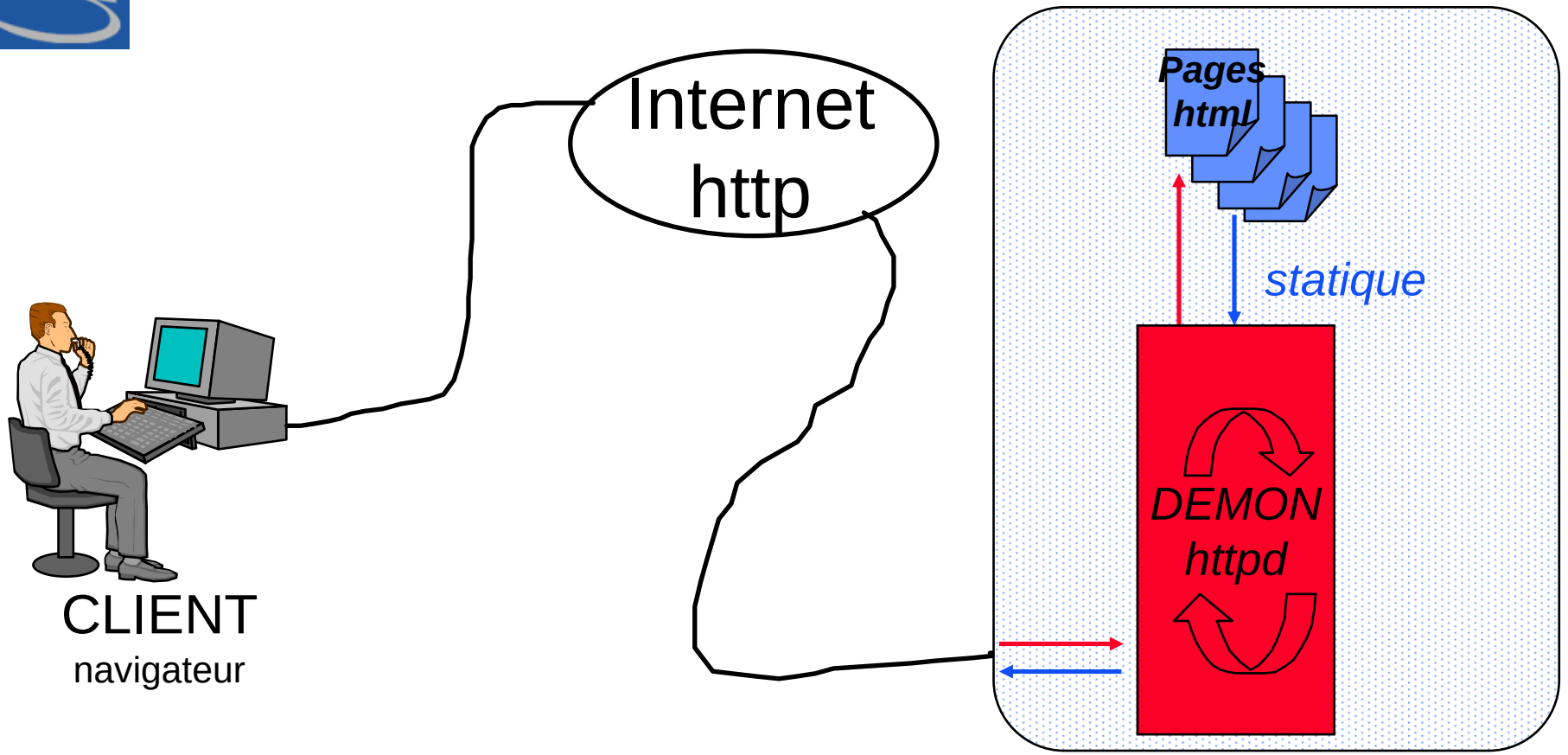
LE WEB - *HTTP/1.1 - Définitions (3)*



modèle http/1.1

(avec cache)

Le Serveur WEB



En fait :

➔ **Le navigateur effectue une requête HTTP (envoi d'entêtes HTTP)**

➔ **Le serveur traite la requête puis envoie une réponse HTTP (envoi des entêtes HTTP de réponses)**

LE WEB - *Les méthodes*

HTTP 1.0

- **METHOD : ressource demandée, version du protocole**
- **GET : télécharger le contenu d'un document**
- **HEAD : récupérer uniquement l'entête d'un document**
- **POST : envoyer des informations au serveur pour traitement (e-mail ou formulaires/CGI)**
- **CORPS (optionnel : méthode POST)**

HTTP 1.1

- **OPTIONS : requête d'informations sur les options de communication possibles, connaître les options d'une ressource**
- **PUT : pour que le client écrive au serveur la ressource citée**
- **DELETE : pour que le client supprime la ressource citée sur le serveur**
- **TRACE : pour invoquer un retour sur ce qu'a reçu le serveur (diagnostic et test)**

LE WEB - *HTTP/1.1 – Les entêtes*

Une requête HTTP est un ensemble de lignes envoyé au serveur par le navigateur. Elle comprend :

- Une ligne de requête: c'est une ligne précisant le type de document demandé, la méthode qui doit être appliquée, et la version du protocole utilisée. La ligne comprend trois éléments devant être séparés par un espace :

  La méthode, L'URL et La version du protocole utilisé par le client

- 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 (Navigateur, système d'exploitation, ...). 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

```
GET http://www.cnam.fr HTTP/1.1
Accept : text/html
If-Modified-Since : Saturday, 15-January-2004 14:37:11 GMT
User-Agent : Mozilla/4.0 (compatible; MSIE 5.0; Windows XP)
```

- Le corps de la requête: c'est un ensemble de lignes optionnelles devant être séparées des lignes précédentes par une ligne vide et permettant par exemple un envoi de données par une commande POST lors de l'envoi de données au serveur par un formulaire

LE WEB - *L'entête HTTP - Client*

User-Agent : *pour spécifier le type du client*

Host *: hôte[:port]*

Accept : *pour spécifier des types de média acceptés en réponse*

Referer *: le client spécifie l'adresse de la ressource qui a fourni l' URL demandée (le serveur peut connaître et maintenir les liens...)*

Authorization : *pour identification/autorisation de l'utilisateur*

Connexion : *pour spécifier les options à appliquer à la connexion*

If-Modified-Since : *le corps du document n'est pas envoyé si il n' a pas été modifié depuis la date indiquée*

From *: pour identifier la requête (depuis http1.1)*

Accept-Encoding *: codage*

Accept-Language *: langage*

Range *: bytes = début-fin*

NB : la ressource demandée peut être un fichier ou la sortie d'un programme exécuté par le serveur

LE WEB - *L'entête HTTP - Serveur*

Allow *: type/sous-type de la ressource*

Content-encoding *: mécanisme de décodage a utiliser*

Content-lenght *: taille de données que le browser doit lire*

Content-type *: type de données (MIME)*

Date *: date d'envoi*

Last-modified *: date de dernière modification du fichier*

Server *: nom du logiciel serveur / version*

WWW-authenticate *: demande d'authentification par le serveur*

Date, Expire , Location *: localisation explicite pour redirection automatique*

Retry-After *: date / secondes*

LE WEB - *HTTP - les réponses (1)*

Une réponse HTTP est un ensemble de lignes envoyées au navigateur par le serveur. 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 texte explicatif. La ligne comprend trois éléments devant être séparés par un espace :

- ✓ La version du protocole utilisé
- ✓ Le code de statut
- ✓ La signification du code

```
HTTP/1.1 200 OK
Date : Sat, 15 Jan 2004 14:37:12 GMT
Server : Microsoft-IIS/2.0
Content-Type : text/HTML
Content-Length : 1245
Last-Modified : Fri, 14 Jan 2000 08:25:13 GMT
```

• **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

LE WEB - *HTTP - les réponses (2)*

Réponse à la requête par le serveur

1) ***status line*** **: HTTP/1.1 code-réponse explication pour l'humain**

code : 5 types(1er digit) XX :

1XX : requête reçue, process continue

2XX : succès (action reçue, comprise et acceptée)

3XX : redirection (une action est nécessaire pour répondre à la requête)

4XX : erreur client (syntaxe requête ou requête non traitable ex:404 :-)

5XX : erreur coté serveur

2) *en-tête requête*

3) *ligne blanche*

4) *data (fichier) ou sortie du programme exécuté par le serveur(dans cgi-bin/ ou server-java/ ou ....DLL ou ....)*

LE WEB - *Une transaction HTTP*

Requête du client :

get /machin/default.html HTTP/1.1
host: lion.cnam.fr
accept: image/gif,image/x-xbitmap,image/jpeg,*/*
user-agent: Mozilla/3.01Gold(WinNT;I)"
connection:Keep-Alive

Réponse du serveur :

HTTP/1.1 200 OK
Date: Mon, 14 Feb 2001 11:08:41 GMT
Server: Apache/1.3.6 (Unix) (Red Hat/Linux) PHP/3.0.8
Last-Modified:Wed,12 Feb 2001 23:34:45 GMT
Accept-Ranges: bytes
Content-lenght:12789
Content-type:text/html
<html>
……..
</html>

LE WEB - *HTTP - Les caches*

- point important (performances ,saturation réseau)
- nécessité de mécanismes plus complets
- transparence sémantique des caches :
  - le client doit recevoir la même chose que si il avait eu accès directement au serveur d'origine
  - d'où de nouvelles en-têtes pour la validation, la notion d'expiration (date de péremption..)
- but du *caching* : diminuer le trafic
  - en éliminant l'envoi de requêtes par des relais (mécanisme d'expiration)
  - en éliminant le besoin d'envoyer la réponse complète (mécanisme de validation)

LE WEB - *HTTP - L'expiration (1)*

- Expiration explicitement spécifiée par le serveur :

  en tête *Expires:date*

  - évite au cache de transmettre la requête, qui peut ainsi fournir une réponse fraîche sans contacter le serveur d'origine
  - si un serveur origine veut toujours être contacté: il met une date d' expiration passée, ce qui force le cache à revalider ses entrées
  - les dates : format GMT

LE WEB - *HTTP - L'expiration (2)*

- ***Age:****en secondes* (en-tête de réponse) permet de **savoir si l'entrée du cache est "fraîche"**

= somme des durées de résidence dans les caches (successifs) + temps de transit dans le réseau

➔ **permet donc d'estimer le temps depuis lequel ce document a été envoyé depuis le serveur d'origine.**

Les serveurs d'origine **doivent** envoyer pour toute réponse un header ***date*** donnant la date d'envoi.

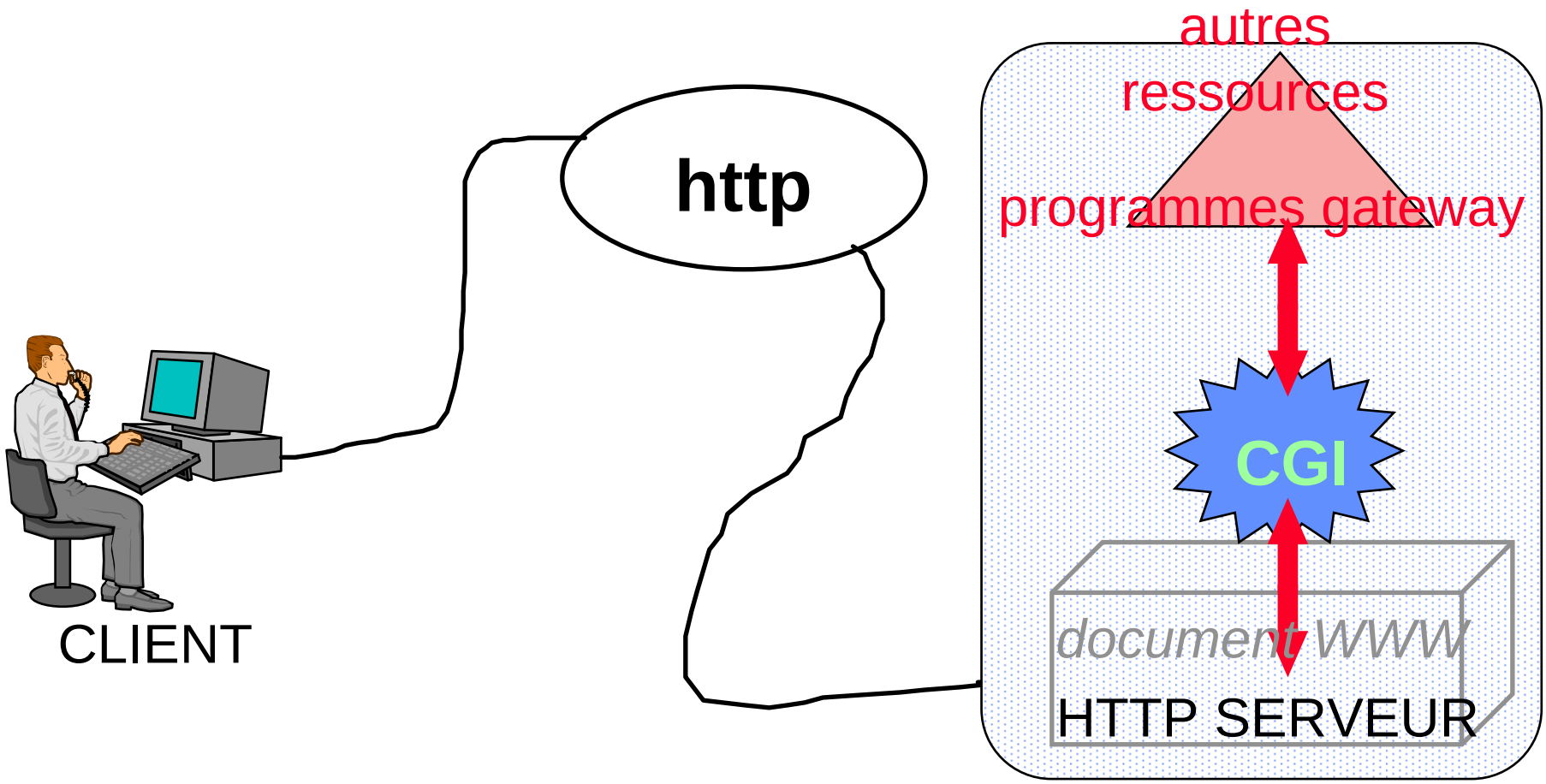
Les caches et serveurs d' origine : doivent disposer d'un protocole de synchronisation pour synchroniser leurs horloges (type NTP)

LE WEB – *Les CGI*

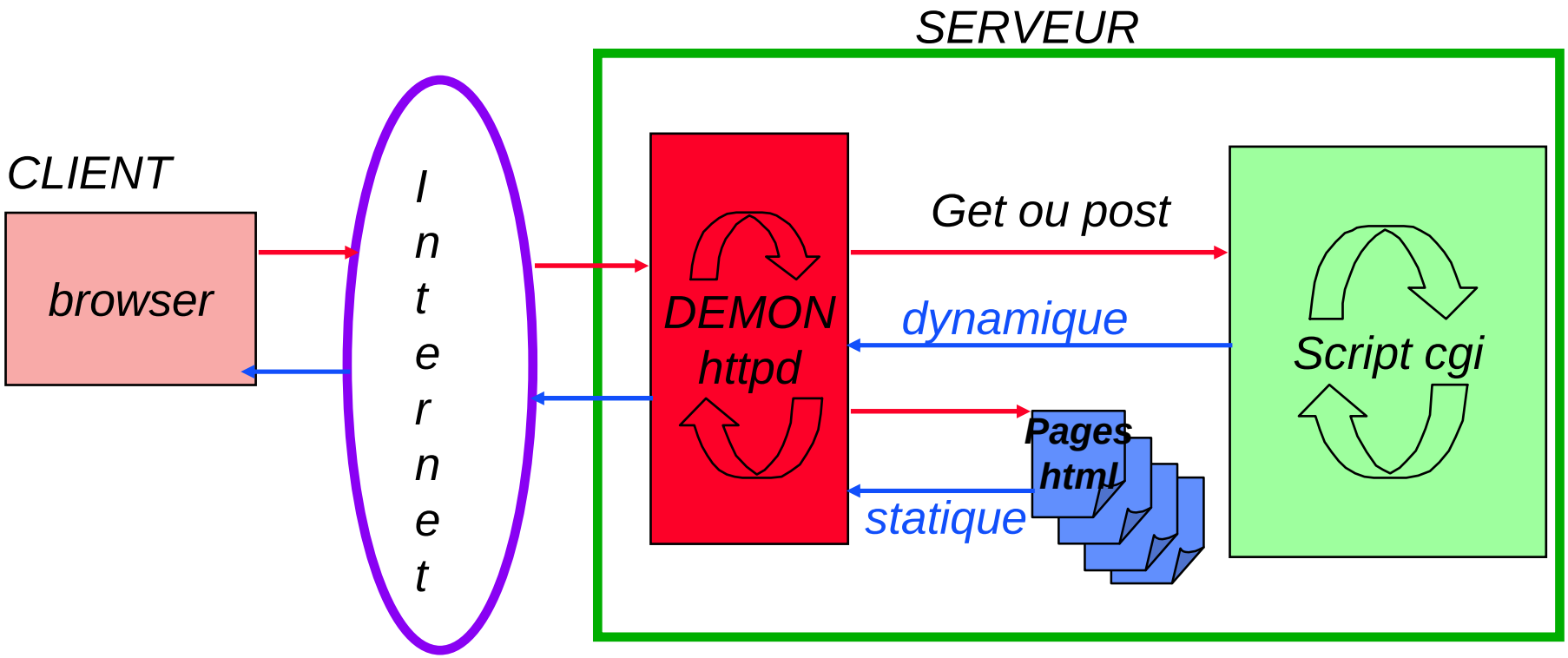
Common Gateway Interface

- **Le mécanisme de communication**
- **L'encodage des données**
- **Un exemple de programme CGI**
- **La description des variables**

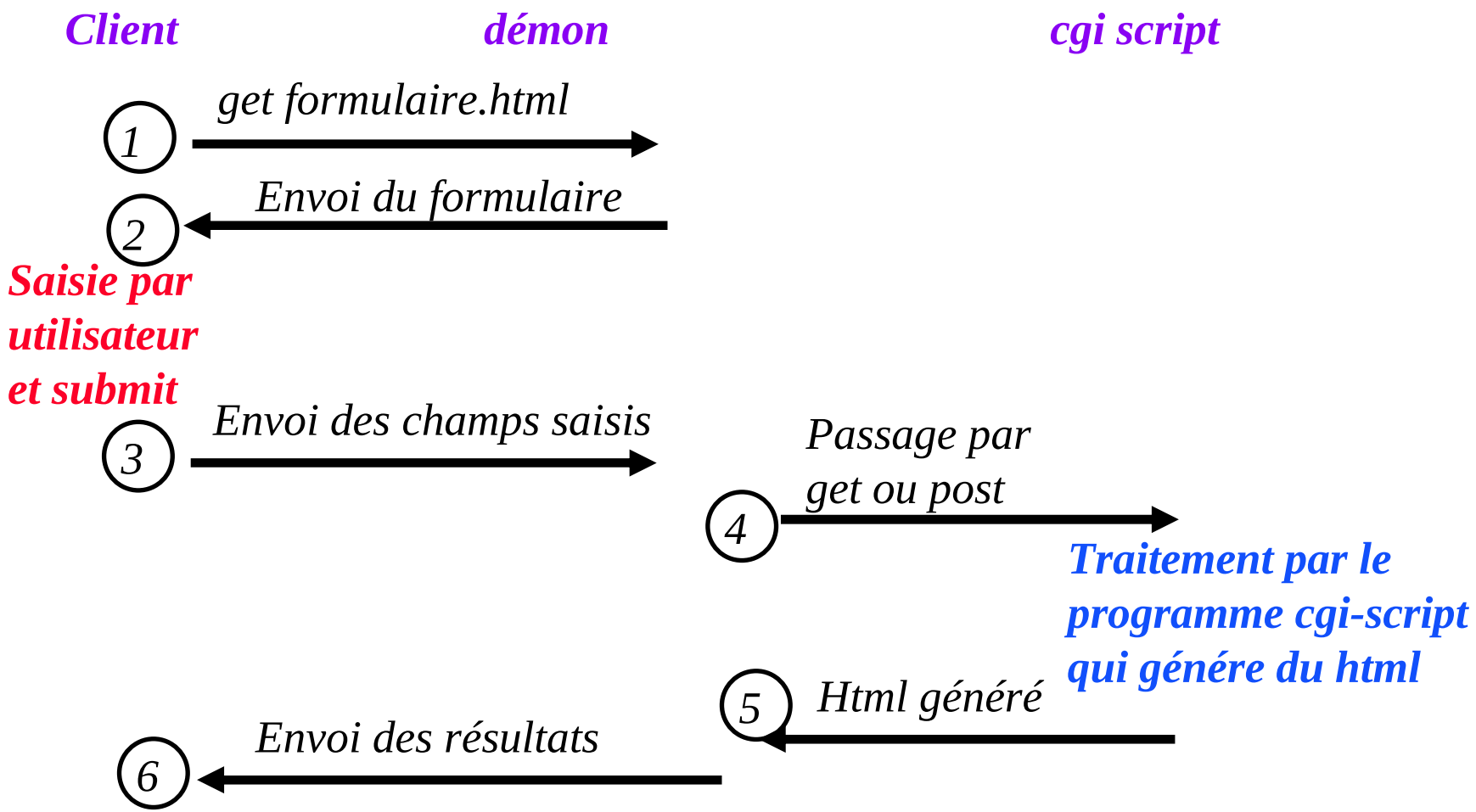
Le Web - Common Gateway Interface



CGI - *Interactivité WWW : CGI*



CGI - *Démarche d'un formulaire CGI*



CGI - *Communication (GET & POST)*

- **GET :**
  - **le au client envoie serveur les données du formulaire avec l'URL, encodées dans une chaîne. *Le serveur reconnaît l'URL comme un appel à un programme du fait du répertoire indiqué (cgi-bin par exemple)***
  - **le programme gateway reçoit les données encodées du formulaire par la variable d'environnement QUERY_STRING.**

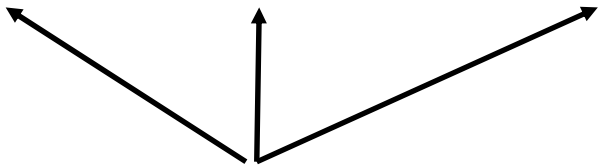
- **POST :**
  - **le client envoie au serveur les données encodées du formulaire dans un message body, avec sa requête.**
  - **le serveur les transmet au programme gateway sur l'entrée standard de celui-ci, (sans EOF).**

  CONTENT_LENGHT donne la taille des données à lire

CGI - *L'encodage des données (1)*

- **GET : l'appui sur SUBMIT envoie au serveur une URL définie par**

action*?*name1=value1*&.....&*namen=valuen



spécifié par ACTION du formulaire

n couples name=value si n champs dans le formulaire séparés par &

name : nom du champ,
value : valeur entrée *(peut être vide)*

- **POST : même encodage,**
  mais ACTION : URL et message BODY : données encodées

CGI - *L'encodage des données (2)*

L'encodage est le suivant:

Toute chaîne de caractères est encodée de façon à ne comporter que des caractères ASCII:

- **les codes >128 étant encodés par %xx ,**
- **les caractères spéciaux de ponctuation interdits (sauf @ * _ - et .),**
- **les blancs sont remplacés par des +,**
- **les couples (issus des formulaires) étant séparés par le caractère &.**

CGI - *Exemple d'encodage (1)*

Aidez nous en remplissant le questionnaire ci -dessous

Votre societe: Conservatoire National des Arts et Métiers

Domaine informatique? ☑ Nombre d' employes: 1400

Navigateur utilise

- Internet Explorer ☑
- Mosaic ☐
- Netscape Communicator ☑
- Autres: win machin, wintruc

votre nom: Dupont@cnam.fr

merci de votre collaboration.

Envoi | RAZ

CGI - *Exemple d'encodage (2)*

```
<FORM METHOD="POST" ACTION="http://cocker.cnam.fr/cgi-bin/post-query"
ENCTYPE="x-www-form-encoded">
<P>Aidez nous en remplissant le questionnaire ci -dessous</P>
<P>Votre societe <INPUT NAME="org" TYPE="text" SIZE="48"></P>
<P>Domaine informatique? <INPUT NAME="informatiq" TYPE="checkbox" VALUE="on">
Nombre d' employes<INPUT NAME="emplois" TYPE="text" SIZE="6"></P>
<P>Navigateur utilise</P>
<UL>
<LI>Internet Explorer <INPUT NAME="browsers" TYPE="checkbox" VALUE="explorer">
<LI>Mosaic <INPUT NAME="browsers" TYPE="checkbox" VALUE="mosaic">
<LI>Netscape Communicator<INPUT NAME="browsers" TYPE="checkbox" VALUE="navigator">
<LI>Autres <INPUT NAME="autres" SIZE="40" TYPE="text">
</UL>
<P>votre nom <INPUT NAME="contact" SIZE="42" TYPE="text"></P>
<P>merci de votre collaboration.</P>
<P><INPUT TYPE="submit" VALUE="Envoi"><INPUT TYPE="reset" VALUE="RAZ">
</FORM>
```

CGI - *Exemple d'encodage (3)*

POST /cgi-bin/post-query HTTP1.1
Accept=www/source
Accept:text/html
Accept:image/jpeg.....
liste des types acceptés par le client
User-Agent: Netscape...
From ....
Content-Type:application/x-www-form-urlencoded
content-length:n *(nombre d'octets du message)*
**********ligne blanche**********
org=Conservatoire+National+des+Arts+et+Métiers
&informatiq=on
&emplois=1400
&browsers=explorer
&browsers=navigator
&autres=win+machin%2C+wintruc
&contact=Dupond@cnam.fr

CGI - *Les variables CGI (1)*

1) décodage de l'information (ces fonctions sont souvent fournies)

2) traitement de la requête en fonction de la chaîne reçue

3) Envoi du document HTML (créé par le script cgi) (idem1)

Pour créer ce document, le programme cgi dispose des variables d'environnement :

SERVER_SOFTWARE : nom et version du logiciel serveur

SERVER_NAME : nom de la machine ou numéro IP.

GATEWAY_INTERFACE : version des spécifications CGI

SERVER_PROTOCOL : nom et numéro version du protocole de requête

SERVER_PORT : port utilisé pour la requête (80 par défaut)

REQUEST_METHOD : GET ou POST

CGI - *Les variables CGI (2)*

PATH_INFO : information contenue dans l'URL mais qui suit la partie identifiant le programme passerelle.

PATH_TRANSLATED permet une version translatée de PATH_INFO pour traduire le chemin logique en nom de fichier

SCRIPT_NAME : URL du programme en cours d' exécution

REMOTE_HOST : nom de la machine émettrice de la requête

REMOTE_ADDR : adresse IP de la machine distante

AUTH_TYPE : si contrôle d'identité par le serveur et programme passerelle protégé, décrit la méthode d'identification

REMOTE_IDENT : si système d' identification(RFC931),nom d'utilisateur

CONTENT_TYPE : spécifie la nature de la donnée associée (méthodes PUT et POST).

CGI - *Les variables CGI (3)*

et surtout :

QUERY_STRING : information sur laquelle est basée la recherche dans l'URL qui appelle le programme passerelle (méthode GET)

CONTENT_LENGTH : longueur en octets de la donnée associée, utilisé avec PUT et POST. Le programme passerelle doit utiliser cette variable pour connaître la quantité d' informations disponible

CGI - *En conclusion*

- en fonction des données reçues, le programme cgi retourne le document créé par sa sortie standard.

Il doit impérativement renseigner respecter les en-têtes du type :
Content_type, *la ligne blanche puis générer du html*

- le serveur est alors en mesure d'envoyer le document html au client.

➔ début de l'interaction....

LE WEB2 – *Problématique*

➢ **La technologie Web entière est basée sur le modèle de l'aller-retour : pour une requête serveur, vous avez un retour qui se traduit par un rafraîchissement des données (la page Web affichée).**

➢ **Ce modèle de fonctionnement est fiable car existant depuis très longtemps mais il pose aussi des problèmes d'interaction homme machine et de performances.**

- ✓ **D'un point de vue utilisateur, le rafraîchissement de toute la page au moindre clic est synonyme de temps d'attente et de scintillement qui n'est pas toujours du meilleur effet dans une application professionnelle.**

- ✓ **Du point de vue des performances, à la moindre modification, vous rechargez une page entière avec toutes ses balises HTML, ce qui génère un trafic important.**

LE WEB2 – *Introduction à AJAX (1)*

➢ **AJAX (*Asynchronous Javascript And XML*, traduisez *Javascript asynchrone et XML*) est une méthode de développement web basée sur l'utilisation d'un script Javascript pour effectuer des requêtes web à l'intérieur d'une page web sans recharger la page.**

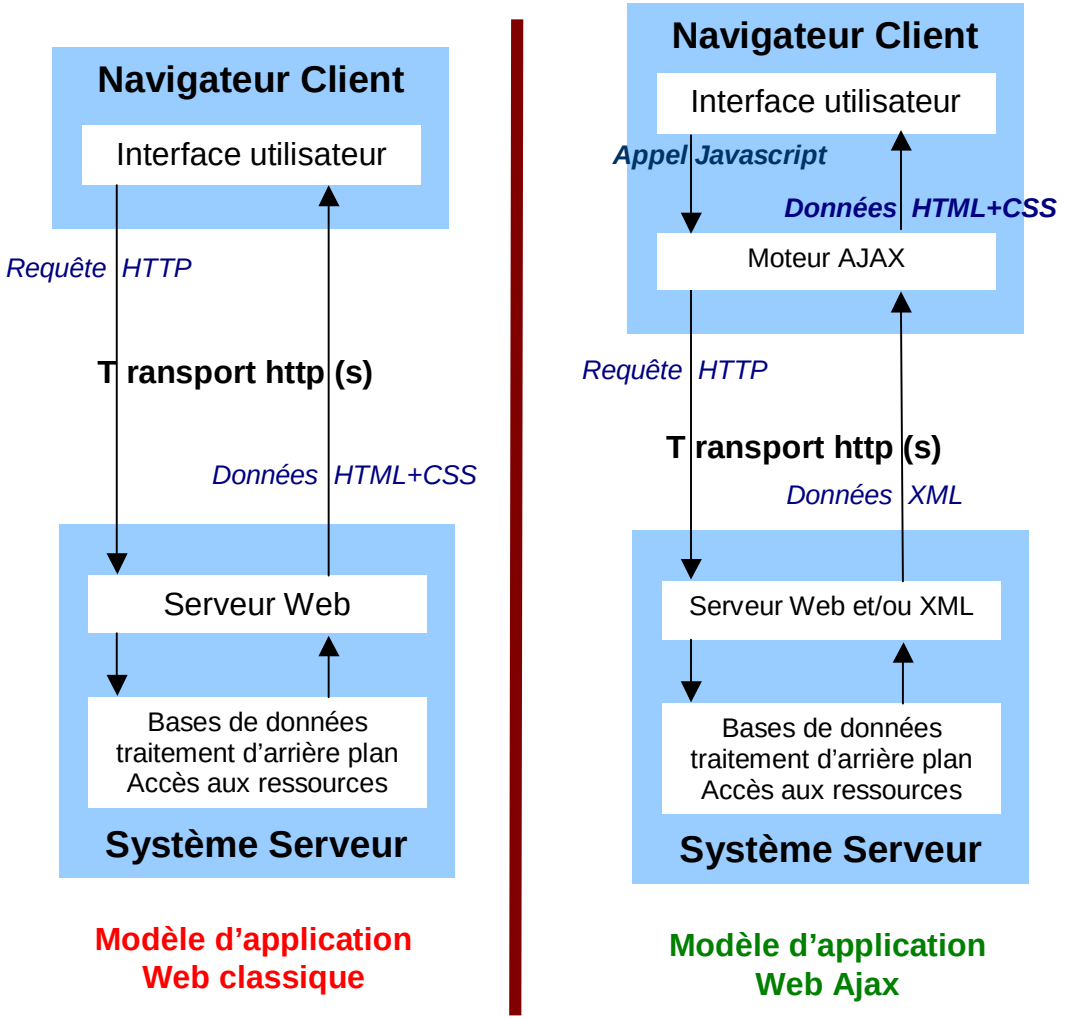
➔AJAX rend plus interactifs les sites web et offre une meilleure ergonomie ainsi qu'une réactivité amélioré en permettant de modifier interactivement une partie de l'interface web seulement.

➔La nouveauté d'AJAX est bel et bien de tirer parti des trois outils pour faire des applications dynamiques.

➢ En effet, le modèle web traditionnel est basé sur une suite de requêtes et de réponses successives, c'est-à-dire une navigation séquentielle de page web en page web. AJAX permet de ne modifier que la partie de la page web qui nécessite d'être mise à jour en créant une requête HTTP locale et en modifiant tout ou partie de la page web en fonction de la requête HTTP récupérée.

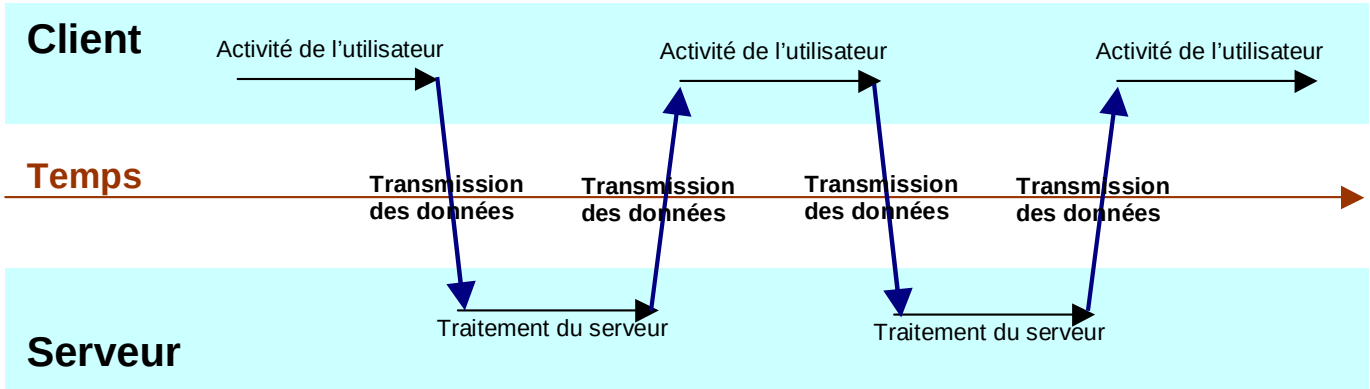
➢ AJAX est basé sur l'objet XMLHttpRequest qui permet de faire une requête via Javascript à un serveur HTTP. Le but est donc, comme dans le "Web 1.0", de faire une requête au serveur et d'en attendre le retour. Cependant, dans notre cas, le navigateur du client n'est pas nécessairement rafraîchi et tout est transparent pour l'utilisateur.

LE WEB2 – *Introduction à AJAX (2)*

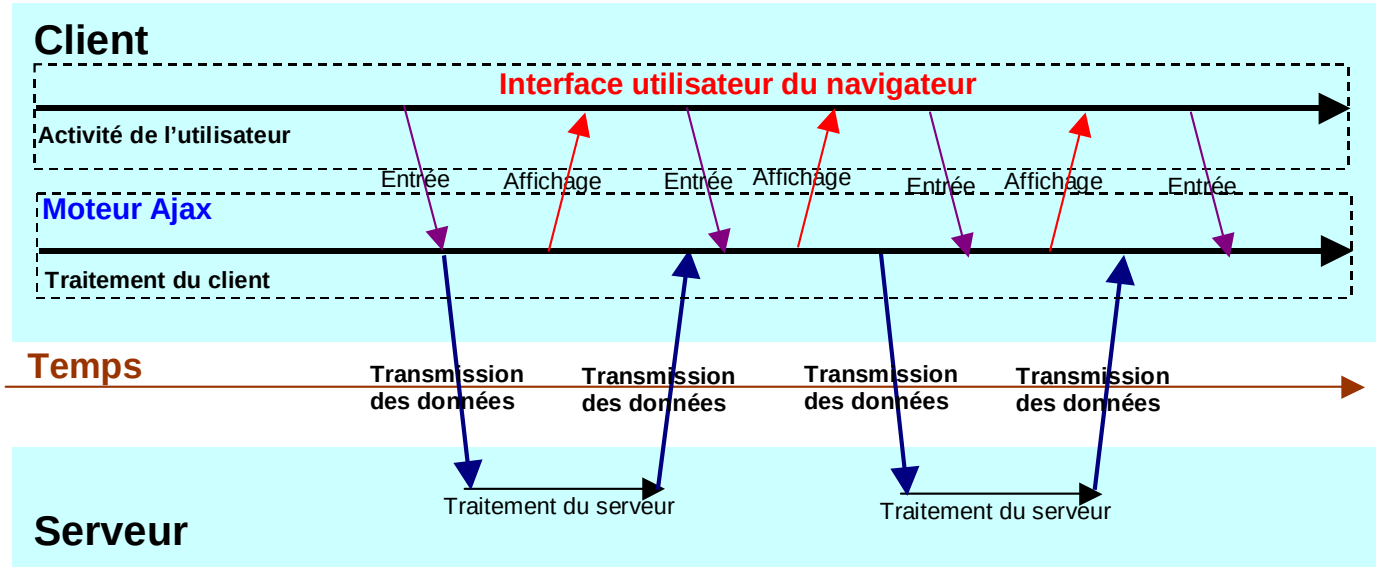


LE WEB2 – *Introduction à AJAX (3)*

Modèle d'application Web classique (synchrone)



Modèle d'application Web Ajax (asynchrone)



LE WEB2 – *Les principes d'AJAX*

➢ **il n'y a pas vraiment de moteur AJAX, de la même manière qu'il n'est pas possible de télécharger AJAX, ou de trouver un site référençant les fonctions AJAX, tout simplement car AJAX est plus une 'manière' de programmer (pas une méthode), voire carrément un mode de programmation, dans le sens ou la programmation est assimilable a la pensée**

➔**AJAX est plus une vision idéaliste de la programmation, que de la programmation en soit.**

➢**AJAX n'est pas un nom ou une marque déposée, c'est un acronyme inventé. Il permet de faire interagir le client encore plus que ne le permettent jusqu'ici les pages Web (même avec flash et/ou Javascript)**
le principe étant d'être en constante interaction avec l'utilisateur et de communiquer avec le serveur et les bases de données de manière transparente (sans recharger la page)

Le principe technique est simple :

➔ **à l'aide de javascript, on envoie une requete au serveur (genre :**

OnClick="javascript:runscript.URL(blabla.php?paramètres)")

➔ **le serveur renvoie une page xml de structure connue par le 'moteur' AJAX , ce dernier utilise le contenu XML pour restructurer une zone ou la totalité de l'interface du client, et ceci, sans rechargement de page.**

ce principe permet une grande rapidité (pas de rechargements d'images, headers, etc...) et une grande fluidité (pas de saccades ou de chargements progressifs, tout dépend de la machine du client)

donc le 'moteur' AJAX est en fait un ensemble de fonctions Javascript basées sur la prise en charge des chaînes, et des balises, ainsi que sur les interactions serveur/client java par le biais de la classe XMLHttpRequest

LE WEB2 – *Les limites d'AJAX*

♦ Un débogage cauchemardesque.

Il existe très peu d'outils de test et d'aide à la conception. Pour créer une interface riche, il faut connaître sur le bout des doigts les différences entre les navigateurs et tester sur chaque plate-forme pour s'assurer que le résultat soit correct.

♦ Javascript n'est pas un langage de haut niveau.

- Il n'est pas totalement orienté objet, et les différences entre les plates-formes posent un problème.
- Si les lignes de code sont nombreuses, des problèmes de performance peuvent se poser.

♦ Une technique encore peu outillée.

Son succès passera par le développement de frameworks standards. Les premiers sont apparus, capables de masquer les détails de chaque navigateur.

LES TECHNOLOGIES DU WEB

1ère Partie : **Introduction au Web**

- 1- **Introduction à l'Hypertexte**
- 2- **Présentation du protocole HTTP**
- 3- **Principes de bases des CGI**
- 4- **Présentation du WEB2** *(AJAX)*

2ème Partie : **Présentation de HTML & XHTML**

3ème Partie : **Présentation de Javascript**

4ème Partie : **Introduction à PHP**

5ème Partie : **Introduction à XML & XSLT**

2ème Partie : Présentation de HTML

HyperText Markup Language

- Introduction à HTML
- Les règles générales
- Présentation du Titre
- Description du Document
- Etude des Tables et des formulaires
- Présentation des feuilles de styles
- Présentation de XHTML

HTML - *Introduction*

Inventé par Tim Berners-Lee (CERN)

- Application de SGML (ISO8879)
- Recommandation W3 Consortium (IBM, Microsoft, Netscape, Novell, Softquad, Spyglass, Sun)
  - ✓ HTML 3.2 (Janvier 97) (tables, images cliquables, applets ... pas de frames)
  - ✓ HTML 4.0 (retour des frames...)
    - draft du consortium (Septembre 97) puis standard fin 97
    - CSS1 Cascading style sheets, expressions mathématiques, internationalisation ,objets multimédias, multilingue, animation(couches)…
    - ➔ supporté par les tous les navigateurs courants
  - ✓ XHTML (HTML 5.0)

- **SGML: 4 éléments**
  - ✓ déclaration SGML (définit les caractères et délimiteurs)
  - ✓ DTD : Document Type Définition : syntaxe de construction des marques
  - ✓ Spécification : sémantique des marques
  - ✓ Instance du document

HTML - *Règles générales (1)*

- **Jeu de caractères: ISO8859-1(Iso latin1)/unicode**
  - **problèmes compatibilité Dos,MACIntosh et Windows....**
- **tag HTML :**

  < délimiteur début

  / fin de

  > délimiteur fin

  tag *: nom du tag (case insensitive) en général : nom*

  + attributs optionnels: align="center"
- **la plupart des tags vont par paires :(conteneur)**

<h1> texte de niveau H1</h1>

élément HTML

HTML - *Règles générales (2)*

- **Pour tout tag : ensemble d'attributs optionnels definis par** *NOM_attribut="valeur"*

(quote simple ou double , omis si lettres,chiffres, . et -)

(syntaxe simplifiée possible : COMPACT="COMPACT" peut s' écrire COMPACT=COMPACT, ou COMPACT)

- **Les *URL* Uniform Ressource Locator:adresse de document WWW :**

scheme://hostname[:port]/path/filename

protocoles :

http, ftp, news, telnet, ...

mailto (avec adresse électronique dupont@machin.dom)

nom de machine (domaine) et port (optionnel si défaut)

HTML - *Structure générale*

<! DOCTYPE> déclaration de la version HTML utilisée

<html> début du html

<head> collection non ordonnée d'éléments
TITLE,SCRIPT,STYLE,ISINDEX,META,LINK,BASE

</head>

<body>

corps du document proprement dit

</body>

</html> fin du document

HTML - *HEAD (1)*

<head>

<title> tiens du html</title >

pour utilisation

voir HTML 4.0

<script>langscript</script>

<style>infos style </style>

presque plus utilisé car FORM

<isindex prompt="message">

obligatoire, définit le titre du document
-titre dans la fenêtre browser, -nom dans la liste des signets (bookmark)

langage de script par défaut
feuille de style par défaut

utilisation dans la recherche simple (avant les formulaires)
l'utilisateur entre une chaîne (envoyée au serveur qui renvoie le résultat de recherche)

HTML - *HEAD (2)*

- **Méta-information sur le document :**

  <meta name="nom de l'info" content="info">

  attribut: NAME ou HTTP-EQUIV

  - les moteurs de recherche reconnaissent et utilisent en général: name=keywords , description

  on trouve aussi author, generator, robots, reply-to, rating

  - **HTTP-EQUIV**: relation avec HTTP:

<meta http-equiv=Expires content="Wed,29 Oct 1997 18:30:24 GMT">

générera l'en-tête Expires (de http)

Expires: Wed,29 Oct 1997 18:30:24 GMT (utilisé par les caches pour la fraîcheur du document)

Citons aussi: **<meta http-equiv=" Pragma " content= " no-cache" >**

HTML - *Le document (1)*

<body attr1="val-1" ... attrn="val-n">

- **Les attributs possibles:**
  - ***bgcolor*: couleur de fond #RRGGBB (en hexa)**
  - ***text*: couleur du texte**
  - ***background*: spécifie l' URL de l'image utilisée en motif de fond**
  - ***link*: couleur des liens hypertexte non encore visités**
  - ***vlink*: couleur des liens hypertexte déjà visités**
  - ***alink*: couleur des liens quand l'utilisateur clique dessus**

- **Si aucune couleur précisée : valeur par défaut du navigateur pour chacun des attributs**

HTML - *Le document (2)*

Exemple :

```
<! DOCTYPE HTML PUBLIC "-//W3C/DTD HTML 4.0 Final/EN">
<html>
<head>
<title> Exemple de tag body </title>
<body bgcolor=#00FFFF link=#000080 vlink=#800080
    alink=#FF0000 text=#808080>
texte quelconque
</body>
</html>
```

NB : les attributs de BODY : souvent non définis car style sheet....

HTML - *Le document (3)*

Les éléments du BODY sont de type BLOCK ELEMENT ou de type TEXT ELEMENT:

- **BLOCK ELEMENT: qui produit des ruptures de paragraphes**
  - **TITRES :(HEADERS) six niveaux possibles <H1> à <H6> , avec attribut ALIGN=center, right ou left**
  - **ADDRESS est en général en fin de document pour indiquer des informations aux lecteurs (par exemple URL du document, auteur avec e-mail et adresse postale).**
  - **P : paragraphes, Listes (UL,OL,DL), PRE(préformatté), DIV (une division), CENTER (pour centrage), BLOCKQUOTE(citation), TABLE(présentation structurée), HR(règle horizontale), FORM (formulaire)....**

HTML - *Le niveau TEXT*

Marquages spéciaux :

A - ancre hypertexte**, attributs : HREF** (URL de destination) **et NAME** (destination dans le document).

BASEFONT - taille de fontes par défaut, nécessite **l'attribut SIZE *de 1 à 7***

IMG - Image. nécessite l'attribut SRC localisation de l'image (GIF or JPEG format) à insérer**. attributs : ALIGN = LEFT, RIGHT, TOP, MIDDLE, BASELINE and BOTTOM** indiquant l'alignement de texte suivant l'image par rapport à l'image**. WIDTH et HEIGHT attributs en pixels de largeur et hauteur , VSPACE et HSPACE :**espaces autour de l'image (H et V) **BORDER :** épaisseur de bordure si ancre, **USEMAP et ISMAP** attributs si l'image est cliquable(réactive coté client **USEMAP** ou coté serveur **ISMAP), ALT :** alternative à l'image si l'image ne peut être chargée.

APPLET - pour insérer une applet java, les paramètres sont passées par **PARAM**

FONT - modification de Fontes indiquée par **SIZE** (1 à 7 ou +n,-n) et **COLOR (en hexa).**

BR - Line break.

HTML

La table (1)

```
<table cellspacing="2" cellpadding="0" border="4" width="450"
    height="61">
<tr>
   <th valign="bottom" width="116" height"30">en t&ecirc;te 1
     </th>
    <th width="137" height"50">en t&ecirc;te 2 (centree par defaut) et en
    gras</th>
    <th align="right" width="144">titre de cellule cadr&eacute;e a
    droite</th>
</tr>
<tr>
   <td nowrap height="17">contenu d'une cellule avec NOWRAP</td>
   <td width="173" height"20">contenu sans l'attribut NOWRAP</td>
    <td></td>
</tr>
<tr>
    <td height"17"> </td>
    <td></td>
    <td bgcolor="#D7FB23">couleur de fond</td>
</tr>
</table>
```

HTML - *La table (2)*

en tête 1	en tête 2 (centree par defaut) et en gras	titre de cellule cadrée a droite
contenu d'une cellule avec NOWRAP	contenu sans l'attribut NOWRAP	
		couleur de fond

attributs de tABLE :
ALIGN *(Hor. de la table)*,
WIDTH et HEIGHT ,BORDER *(par défaut pas de bordure),*
CELLSPACING : *esp entre cellules en pixels,*
CELL SPADING *:esp. entre contenu et bordure*

attributs de CAPTION : légende **ALIGN:TOP/BOTTOM**
attributs de TR : rangée de la table contient des **TH** *titre et* **TD** *data, avec* **ALIGN** *(left, center,right) et* **VALIGN** *(top,middle,bottom)*

attributs de TH (titre de cellule)et de TD (cellule de données):
NOWRAP*: ne pas casser les mots* **ROWSPAN***: nombres de lignes(1)*
COLSPAN*: nombre de colonnes(1) et* **ALIGN,VALIGN,WIDTH ,HEIGTH**

HTML - *Les formulaires (1)*

Permet à l'utilisateur d'entrer des données (attribut INPUT) **et de les envoyer à un** programme **sur le serveur WWW.**

Ce programme peut alors répondre à la requête du client/utilisateur.

- **C'est par l'élément FORM que l'on peut spécifier le programme, (attribut ACTION) et la METHODE pour envoyer les données : GET ou POST** (plutôt conseillée).
- **Form définit un conteneur:**

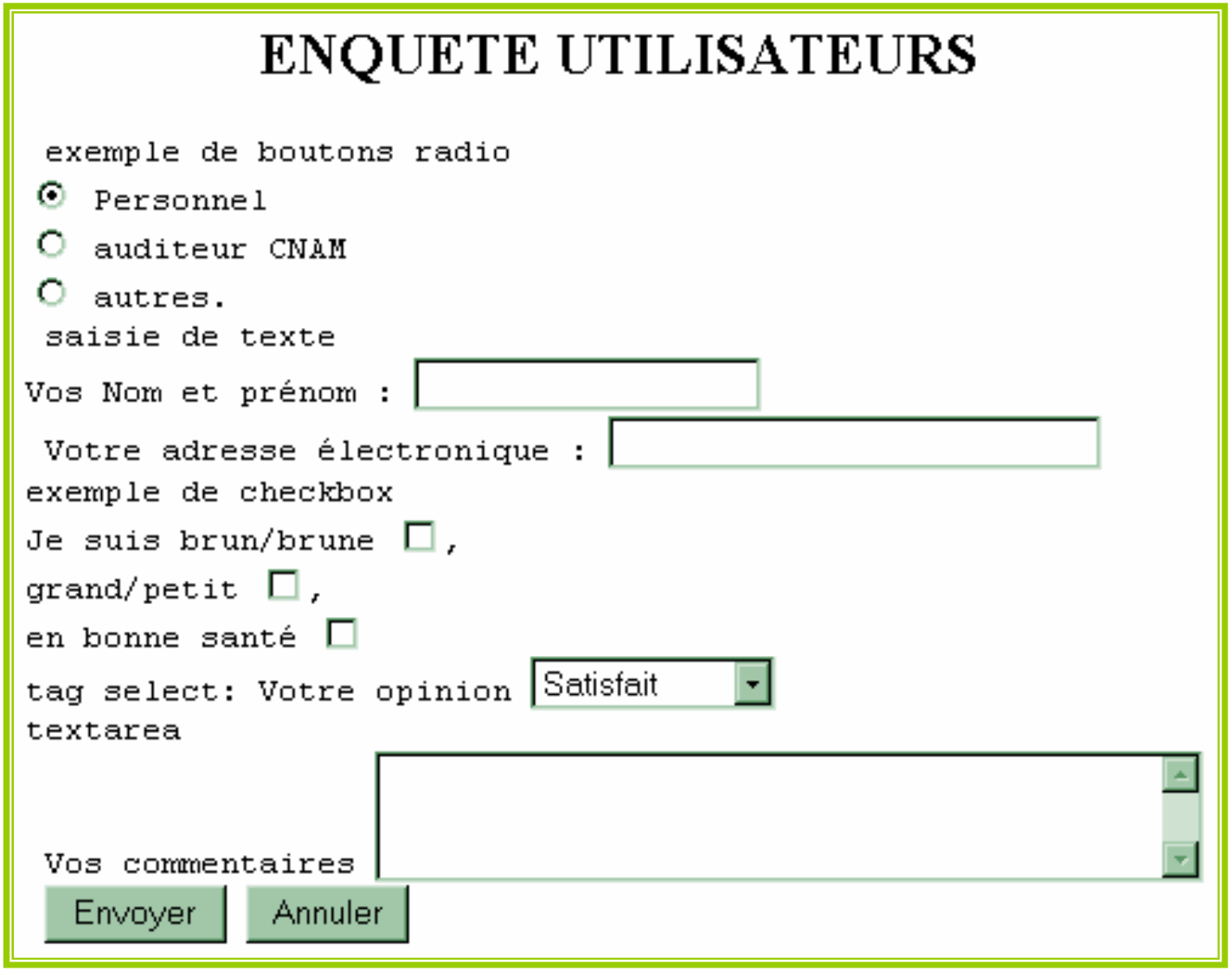
<form method="POST" action="http://site/path/prog/">
contenu du formulaire ici </form>

- données à entrer par INPUT, SELECT (alternative de checkbox) ou TEXTAERA(pour champs d'entrée multilignes)

HTML - *Les formulaires (2)*

```
<p> <h2 align="center">ENQUETE UTILISATEURS</h2></p>
<form action="/cgi-bin/cgiprog" method= "POST">
<pre> exemple de boutons radio
<input type="RADIO" name="SECTEUR1" value="CNAM" checked> Personnel
<input type="RADIO" name="SECTEUR2" value= "ELEVES"> auditeur CNAM
<input type="RADIO" name="SECTEUR3" value= "NQP"> autres.
<p> saisie de texte
Vos Nom et prénom : <input type="TEXT" name="NOM1" size="20"></p>
<p> Votre adresse électronique : < input type="TEXT" name="ADR1" size="30"></p>
<p>exemple de checkbox
Je suis brun/brune<input type=" CHECKBOX " name="BRUN" value="brun">,
grand/petit< input type=" CHECKBOX " name="GRAND" value="grand">,
en bonne santé< input type=" CHECKBOX " name="SANTE" value="en bonne
santé">
tag select: Votre opinion <select name="OPINION1">
<option value="tresbien">Très satisfait
<option value="bien" SELECTED>Satisfait
<option value="bof">Indifférent
<option value="beurk">C'est nul !!
</select>
textarea<p> Vos commentaires <textarea name="COM1" rows="3"
cols="40"></textarea></p><p> <input type="SUBMIT" value="Envoyer"> <input
type="RESET" value="Annuler"></p> </pre> </form>
```

HTML - *Les formulaires (3)*



HTML - *Les types d'ancres*

- **lien vers document distant**

*** nom de l'ancre ***

URL: http:// nommachine:port/path/fichier

- **lien vers fichier local**

*** nom quelconque***

- **lien vers une partie d'un document local**

une étiquette est définie dans le document par

nom et le lien est donné par

nomancre

- **lien vers partie du document courant**

nom et le lien est donné par

nomancre

HTML - *Les images cliquables*

exemple: une barre de navigation

code:

```
<P><MAP NAME="t_pagingButtons.map">
  <AREA SHAPE="rect" COORDS="507,1,528,24" HREF="pp.html"
  ALT=""> <AREA SHAPE="rect" COORDS="483,0,504,24"
  HREF="ps.html" ALT=""> </MAP>

et l'ancre A;

<IMG  SRC="band_sup.gif" USEMAP="#t_pagingButtons.map" HEIGHT="24"
  WIDTH="528" ALIGN="BOTTOM" ISMAP ALT="">
```

L'attribut USEMAP="nom de la carte" est positionné dans IMG et la carte est ensuite décrite dans MAP.

L'élément MAP : attribut NAME="nom de la carte" et les descriptions des attributs AERA décrivant les zones sensibles: AERA:

4 formes (SHAPE)de zones sensibles :

SHAPE=RECT,CIRCLE,POLY,DEFAULT (rectangle, cercle, polygone, et le reste qui n'a pas été spécifié précédemment)

HTML - *Les feuilles de styles*

WEB STYLE SHEETS

Idée de base:

séparer la présentation d'un document du document lui-même, **sans sacrifier l'indépendance inter-plateformes**

– **CSS Cascading Style Sheet 1** (né en 1994, recommandation W3C en 96 et **CSS Cascading Style Sheet 2** (fin 1998)

Ensemble de règles de style indiquant à un browser comment présenter un document:

– **une règle:**

s'applique à un selector (élément HTML comme BODY, P, EM,ou class selector ou ID selector...)

syntaxe: *selector (propriété1:val1;....;propriétén:valn)*

– **exemple:**

```
<head>
<title> CSS Exemple</title>
<style>
H1 (font-size:large;color:red)
H2, H3 (font-size:14pt;font-style:oblique)
</style>
.....
```

HTML - *CSS : les styles & cascades*

- **style interne**
  - **par les attributs des éléments HTML(ex: BODY)**
  - **dans le document :dans HEAD par**

```
<style>
<!--
La ou les feuilles de style
-->
</style>
```

- **Style externe au document:**
  - **LINK (élément de HEAD) par**

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

- **Cascades : possibilité de définir plusieurs style sheets :**

```
<link rel=StyleSheet href="basic.css" title="contemporain">
< link rel =StyleSheet href="table.css" title="contemporain">
< link rel =StyleSheet href="form.css" title="contemporain">
```

  - **les 3 fichiers css définissent un style nommé "contemporain" (importance de l'ordre des déclarations)**
  - **on peut spécifier que des règles sont prioritaires par *!important***
  - **en cas de conflit, les règles des auteurs sont prioritaires sur celles des lecteurs**

HTML - *CSS : Class Selector*

- **Tout élément HTML peut être associé à une CLASS :**
  - **le stylesheet précisant les propriètés des éléments de la classe :**

  p.avertissement (color:red;font-weight:bold)
  - **dans le document (body):**

  *** attention ceci est essentiel.... ***
  - **ainsi seuls les Paragraphes de classe** avertissement **recevront ce style (rouge et très gras)**
- **On peut déclarer une classe sans élément html associé :**

  exemple :

  .warning {font-size:small;}

  la classe warning pouvant alors être associée à divers éléments html

HTML - *CSS : Héritages*

- On peut grouper les sélecteurs :
  - H1,H3,H5{color:red;}
- héritage des propriétés :
  - exemple: les propriétés de body s'appliquent au contenu sauf si surcharge
- commentaires /* */
- pseudo-classes
  - A:link,A:active,A:visited pour les ancres
  - first-line et first-letter pour la 1ère ligne et la 1ère lettre d'un bloc (P,Hi,...

HTML - *CSS : Les propriétés (1)*

- **Font:**
  - **font-family:nom de famille de fonte ou nom générique**
    - ex: font-family: “Times New Roman”, serif, monospace)
  - **font-style:normal, italique, oblique**
  - **font-weight:bold,normal, bolder, lighter...**
  - **font-size : en absolu:** 14pt**, en % :** 90%(base de la taille du parent),..
- **Color**
  - **background-color, background-image, background-attachement, background-repeat, background-position**
  - **color**
- **Texte**
  - **text-align, text-indent, word-spacing, letter-spacing, text-align, text-transform(min,MAJ), text-decoration, width, height, line-height, white-space....**

HTML - *CSS : Les propriétés (2)*

- **Les marges (margin)**
  - **margin-top,right,bottom,left en unité de longueur**
- **Les bords et enrobages**
  - **border-style, color,top-width, right-width,bottom-width,left-width,**
  - **padding-top,-right,-bottom,-left(remplissage)**
- **Les listes (style-image, style-position,style-type) pour les puces des listes**
- **Les unités utilisables:**
  - **en pixels (px), inches (in), centimètres (cm) et pourcentage(%) et points pour les caractères (pt)**

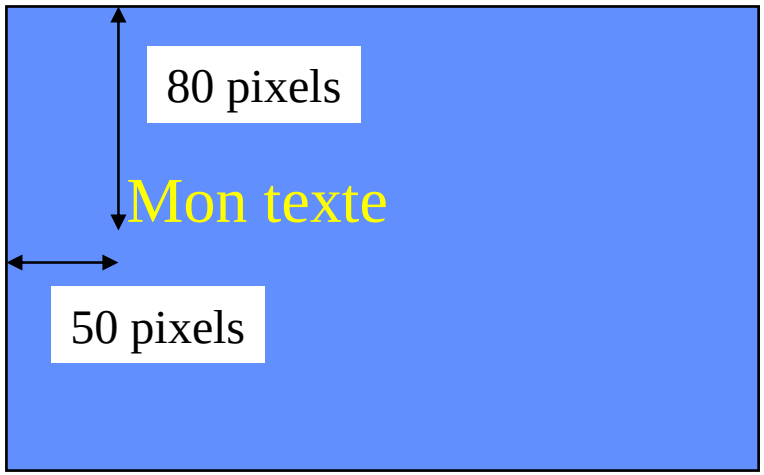
HTML - *CSS : En complément*

On peut positionner des blocs de contenu grâce à CSS:

Exemple :

```
<style type =‘ text/css ’>
.pub{position:absolute;top:80px;left:50px; color:yellow;font-size:x-large;font-wight:bold}
</style>

<div class=pub>
 mon texte</div>
```



– **Dans le body:**

```
<span style=‘position:absolute;top:80px;left:100px; idth:128px;height:200px’>
<img src=‘image.gif ’>
</span>
<span style=‘position:absolute;top:120px;left:160px’>
Mon texte sur l’image
</span>
```

Permettra de positionner du texte sur l’image au pixel prés, mais les layers permettent aussi cela.

HTML - *CSS : En résumé*

- **Séparation contenu et mise en forme**
- **Factorisation de style entre documents (style externe)**
- **Rendre html plus lisible, moins fouillis**
- **Réduire les temps de chargement**

XHTML - *Introduction (1)*

Un langage de balises extensible (XML) est une syntaxe générale, lisible par l'homme et compréhensible par la machine, ayant pour finalité de décrire des données hiérarchisées.

XML n'est pas un langage de balises prédéfinis : c'est un langage de description d'autres langages qui offre la possibilité de définir ses propres balises. Un langage de balises prédéfinis comme HTML ne décrit l'information qu'à partir d'une seule classe de documents. XML quant à lui, permet de définir ses propres langage de balises personnalisés pour différentes classes de documents

XHTML - *Introduction (2)*

- **XML est un sous-ensemble simplifié d'SGML** L'idée maîtresse d'XML est de séparer le contenu de la forme
- **SGML est une description de langage de balises**
- **HTML est un langage définit en SGML**
- **SGML est un standard international de méthodes de représentation de texte au format électronique, c'est un métalangage.** C'est à dire qu'il est une méthode de description formelle de langage, dans le cas présent, de langage de balise.
- Le but est de faciliter la transition depuis HTML vers XML. la solution consiste à **reformater HTML en terme de XML**. **Le résultat est XHTML**, une application particulière de XML, déstinée à l'expression de pages Web.
- XHTML est en fait la version qui va suivre HTML 4. On peut l'envisager comme un HTML 5 que l'on aurait appelé XHTML 1.0. **En XHTML, toutes les balises et attributs d'HTML 4 continuent d'être supportées**.

XHTML - *Eléments clés*

Quelques éléments clés d'Xhtml :

- **Les balises Xhtml sont toujours en minuscule**
- **Xhtml est strictement une version plus " disciplinée " d'Html**
- **Des pages écrites en Xhtml fonctionnent correctement avec la majorité des navigateurs**
- **Toutes les balises, même celles comportant des éléments vides doivent être closes**
- **Xhtml est une reformulation d'Html 4 en tant qu'application d'Xml**
- **Les éléments (tags) ainsi que les attributs sont presque tous identiques à ceux d'Html**

XHTML - Les règles

- **Balises obligatoires :**

  Les éléments <head> et <body> ne peuvent être omis.

- **Les balises et leurs attributs doivent être écrits en minuscule**
- **Les éléments ne doivent pas se chevaucher**
- **Tous les éléments non-vides doivent être fermés**
- **Les éléments vides doivent être “terminés”**
- **Les valeurs des attributs doivent être écrites entre guillemets**
- **Les valeurs des attributs ne peuvent plus être réduits**

XHTML - *HTML ↔ XHTML*

HTML	XHTML
<TD BGCOLOR=″#ffcc33″>	<**td bgcolor**=″#ffcc33″>
engras !	engras !** **
Premier paragraphe	** **Premier paragraphe** **
<hr> <input … > <img src=″valid.gif″>	<hr /> <input … />
<img … border=0>	
<td nowrap>texte</td> <input type=″radio″ … checked>	<td **nowrap=″nowrap″**>texte</td> <input type=″radio″ **checked=″checked″** />

XHTML - Structure du document

```
<!DOCTYPE html PUBLIC ″-//W3C//DTD XHTML 1.0 Transitional//EN″
    ″DTD/xhtml11-transitional.dtd″>
<html>
    <head>
        <title>Document minimal</title>
        <link rel=″stylesheet″ href=″/style.css″ type=″text/css″ />
    </head>
    <body>
        <p> <a href=″http://validator.w3.org/check/refer″>valide</a> </p>
        <dd> les éléments de la liste sont ordonnés, c'est à dire par numéros :
            <ol>
                <li>le premier   </li>
                <li>le second    </li>
            </ol>
        </dd>
    </body>
</html>
```

LES TECHNOLOGIES DU WEB

1ère Partie : **Introduction au Web**

- 1- **Introduction à l'Hypertexte**
- 2- **Présentation du protocole HTTP**
- 3- **Principes de bases des CGI**
- 4- **Présentation du WEB2** *(AJAX)*

2ème Partie : **Présentation de HTML & XHTML**

3ème Partie : **Présentation de Javascript**

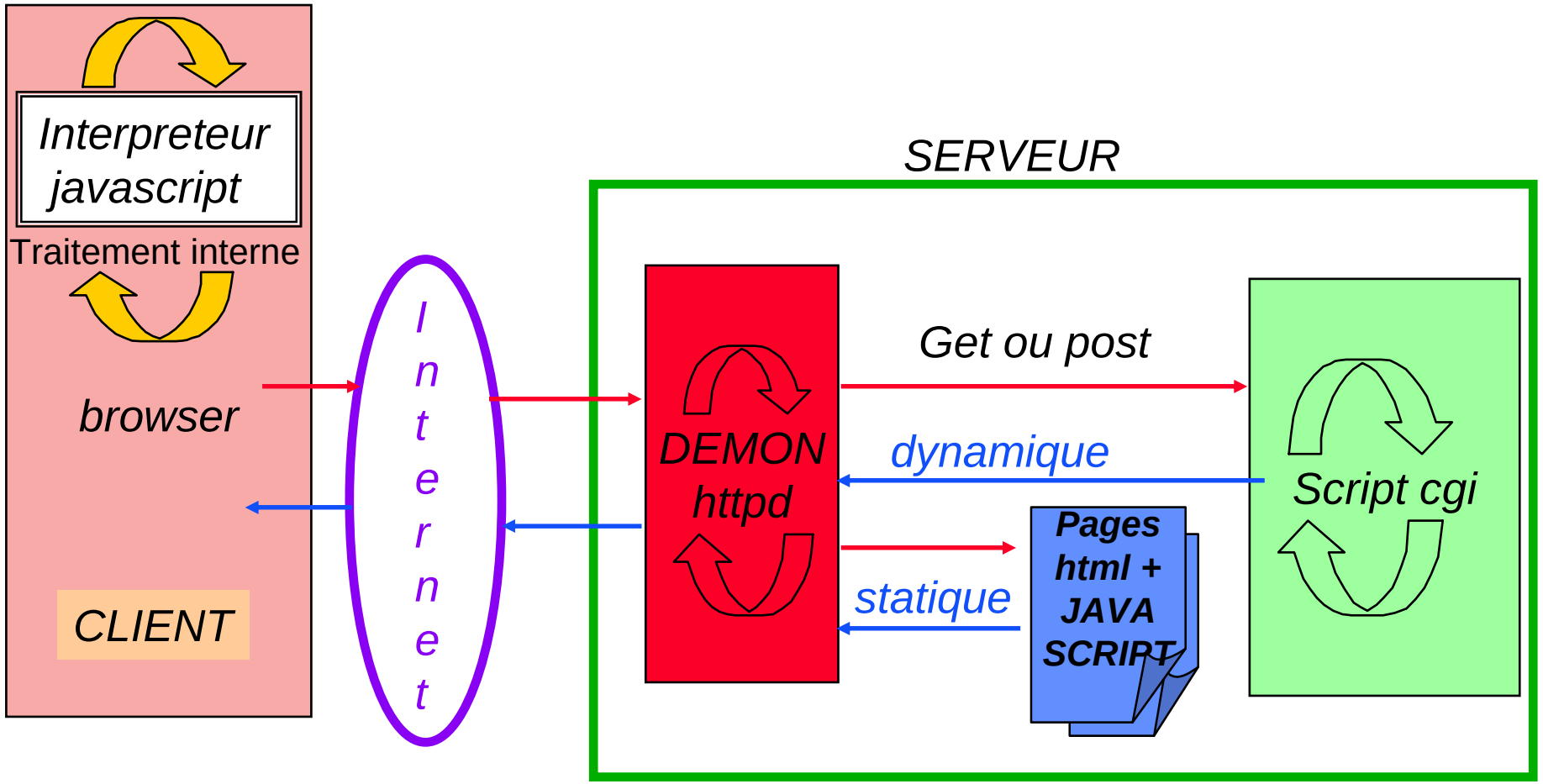
4ème Partie : **Introduction à PHP**

5ème Partie : **Introduction à XML & XSLT**

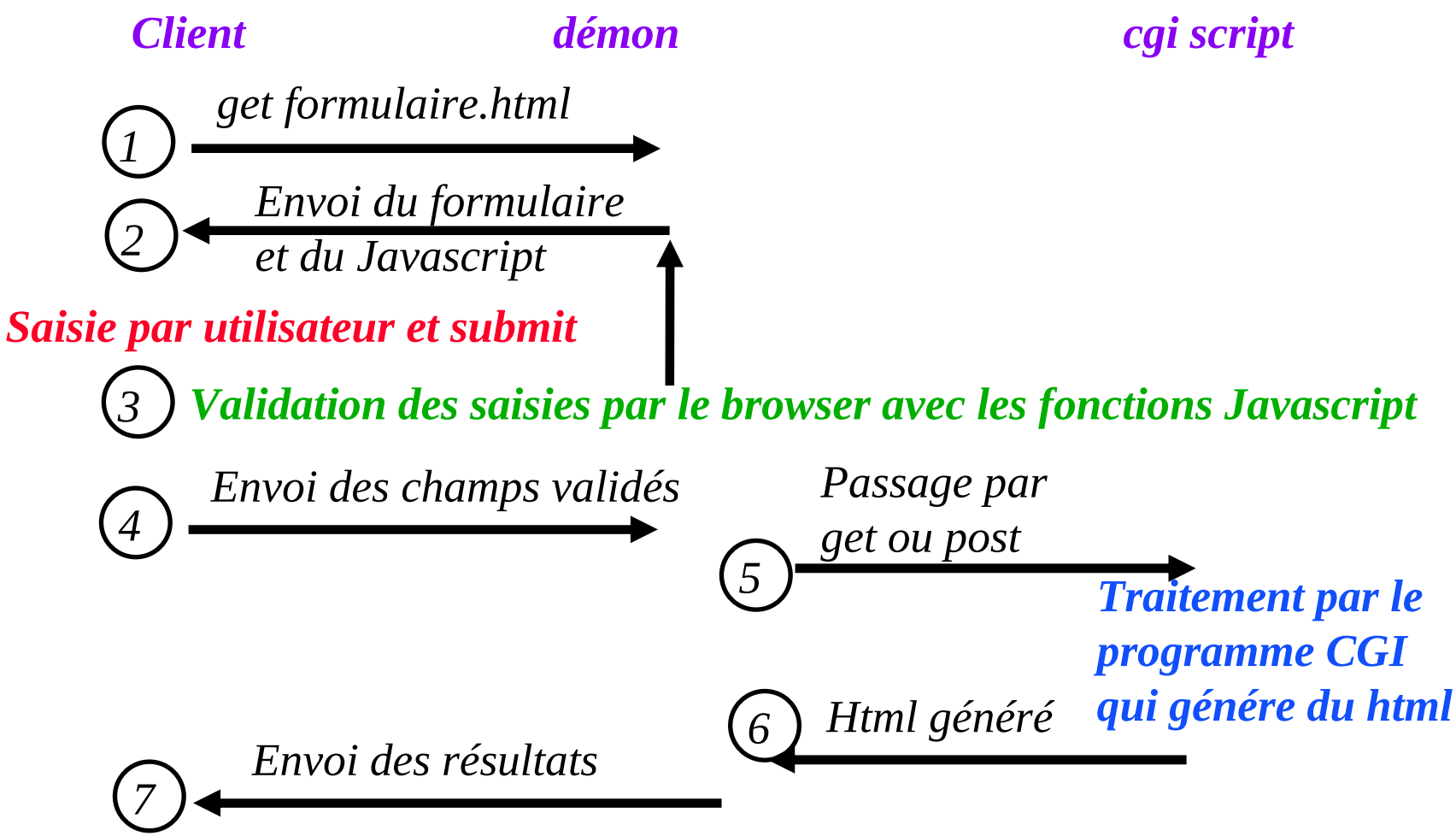
3ème Partie : Présentation de JAVASCRIPT

- **Le mécanisme de communication**
- **L'encodage des données**
- **Un exemple**
- **La description des variables**

JAVACRIPT - *Interactivité*



JAVACRIPT - *Un client actif*



JAVACRIPT - *Introduction*

- **Javascript : développé par Netscape (à partir de Netscape 2.0)**
  - langage de scripts (programmes interprétés) contenus dans le document html, exécutés sur la machine du client (extension de html)

- **javascript ≠ Java**
  - java: véritable langage de programmation, compilé, code protégé,typage fort, objet (classes, instances, héritage)
  - javascript: facile à utiliser, typage faible, intégré dans html,interprété basé sur les objets html

- **Implémenté par le tag SCRIPT et attribut LANGUAGE**

  exemple:

  <SCRIPT LANGUAGE="JavaScript1.1">

  code javascript (version 1.1)

  </SCRIPT>

JAVACRIPT - *Implémentation du code (1)*

Exemples d'insertion de Javascript

```
<SCRIPT LANGUAGE="JavaScript">
   // Définit une fonction clickmulot()
</SCRIPT>
……
……
……
……
……
……..
   <FORM ...> <INPUT
     TYPE="button"
     onClick="clickmulot(this)" ...>
   ...
  </FORM>
```

- **un document devient réactif à un événement si à cet événement est associée une fonction javascript**
- **un événement est déclaré dans un formulaire ou sur un lien hypertexte.**
- **Exemple :**

```
<FORM>
<input type="button" value ="exemple test"
                          onClick="alert('ouah!')">
</FORM>
```

un click souris sur le bouton provoque l'exécution du handler onClick , ici l'affichage d'une fenêtre pop-up **alert** de contenu ouah!

(**alert** : fonction existant dans javascript)

JAVACRIPT - *Implémentation du code (2)*

- **On peut aussi placer le source javascript dans un fichier externe par :**

  <SCRIPT LANGUAGE="Javascript" SRC=source.js></SCRIPT>

  (le serveur devant être configuré pour servir un type MIME "application/x-javascript")

- **on peut définir des variables ou expressions javascript pour définir des attributs HTML :**

  exemple: **<HR width="&(nomvar);" ALIGN="LEFT">**

  définit une barre horizontale cadrée à gauche de 50% si nomvar:variable Javascript vaut 50

JAVACRIPT - *Les événements (1)*

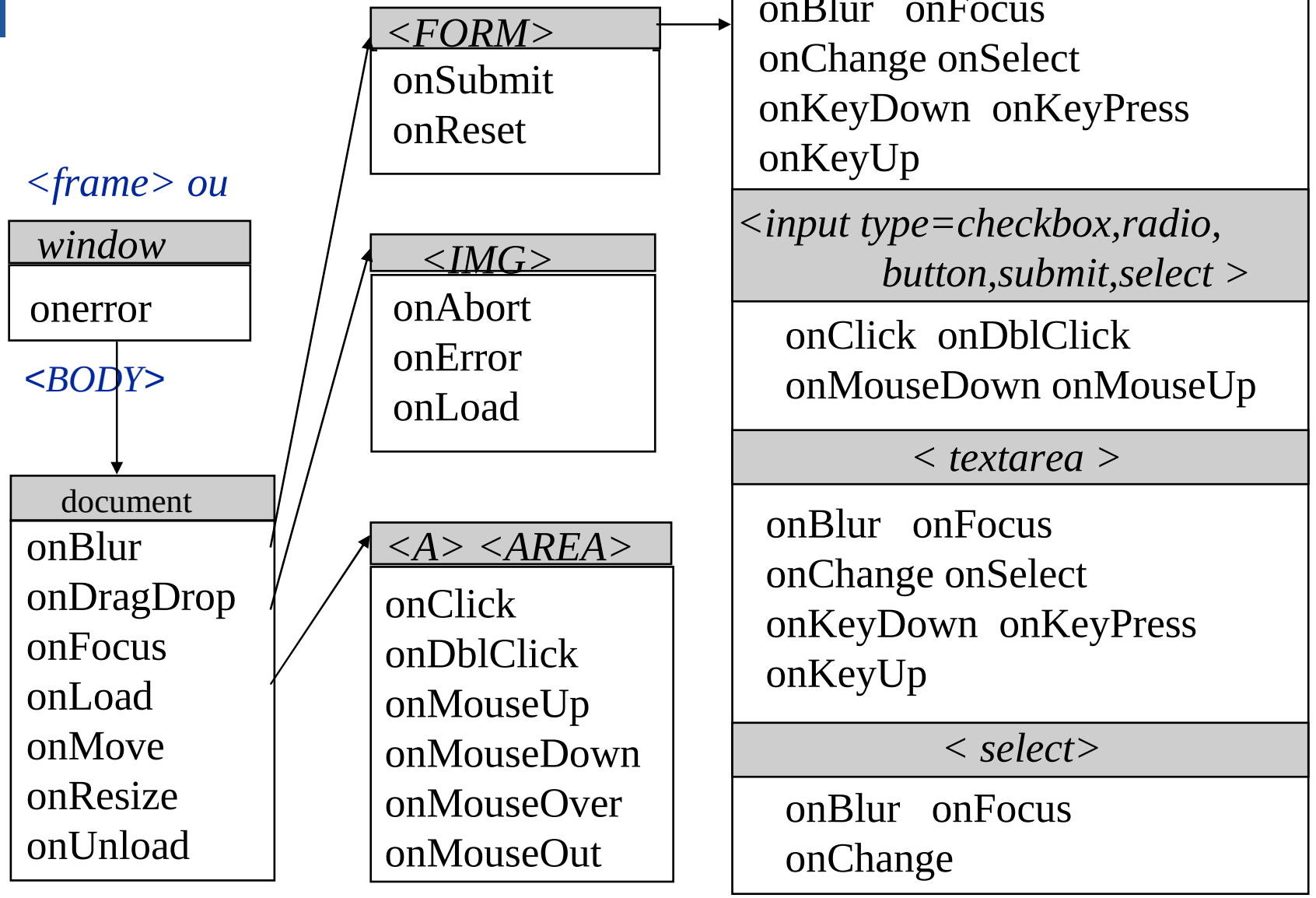
- **Souris**
  - **Click (handler: onClick)**
    - **sur un bouton de form (radio, checkbox, button, submit, reset)**
    - **un lien**
  - **MouseOver (onMouseOver) sur un lien**
  - **MouseOut (onMouseOut) sur un lien ou area d'image cliquable**
  - **MouseDown (onMouseDown) et MouseUp (onMouseUp) bouton appuyé (relaché) sur un document bouton ou lien**
  - **MouseMove (on MouseMove) : mouvement de souris**
  - **DblClick (onDblClick)**

- **Clavier**
  - **KeyPress (onKeyPress) l'utilisateur a appuyé sur une touche**
  - **KeyDown (onKeyDown) l'utilisateur enfonce une touche**
  - **KeyUp (onKeyUp) relâchement sur une touche dans un document, image, liens ou textarea**
  - **Change (onChange) changement de la valeur d'un élément type text, textarea et select list de form**

JAVACRIPT - *Les événements (2)*

- **Focus**
  - Focus (onFocus) l'utilisateur donne le focus input dans une fenêtre ou un élément de formulaire
  - Blur (onBlur) l'utilisateur enlève le focus input dans une fenêtre ou un élément de formulaire
- **Formulaires**
  - Select (onSelect) sélection d'un élément de formulaire (type text et textarea)
  - Submit (onSubmit) et Reset (onReset) soumission (ou annulation) des données d'un formulaire
- **Document**
  - Load (onLoad) et Unload (onUnload) : chargement et sortie de page
  - Resize (onResize) et Move (onMove) d'une fenêtre
  - Abort (onAbort) arrêt du chargement d'une image
  - Error (onError) de chargement d'image ou document
  - DragDrop (onDragDrop) l'utilisateur fait glisser un objet (ex:fichier) dans la fenêtre du browser

Les objets et les handlers



JAVACRIPT - *Exemples (1)*

le code :

```
<A HREF="#fin" onClick="alert('onClick sur le lien')">onClick
</A>l'utilisateur clique sur une zone hypertexte.<BR>
[<I>cliquez sur onClick</I>]
```

donne:

onClick l'utilisateur clique sur une zone hypertexte.
[cliquez sur onClick]

JAVACRIPT - *Exemples (2)*

le code:

```
<FORM><INPUT VALUE=onFocus onFocus="alert('onFocus
    sur la   zone')">l'utilisateur  clique sur une zone.<BR>
[<I>cliquez sur onFocus</I>] </FORM>
```

donne:

on focus **l'utilisateur clique sur une zone.**

[cliquez sur onFocus]

JAVACRIPT - *Basé sur des objets*

- **Javascript est basé sur des objets, entités informatiques de base, avec des propriétés et un comportement défini par des méthodes (fonctions associées à des objets).**

- **les objets de Javascript :**

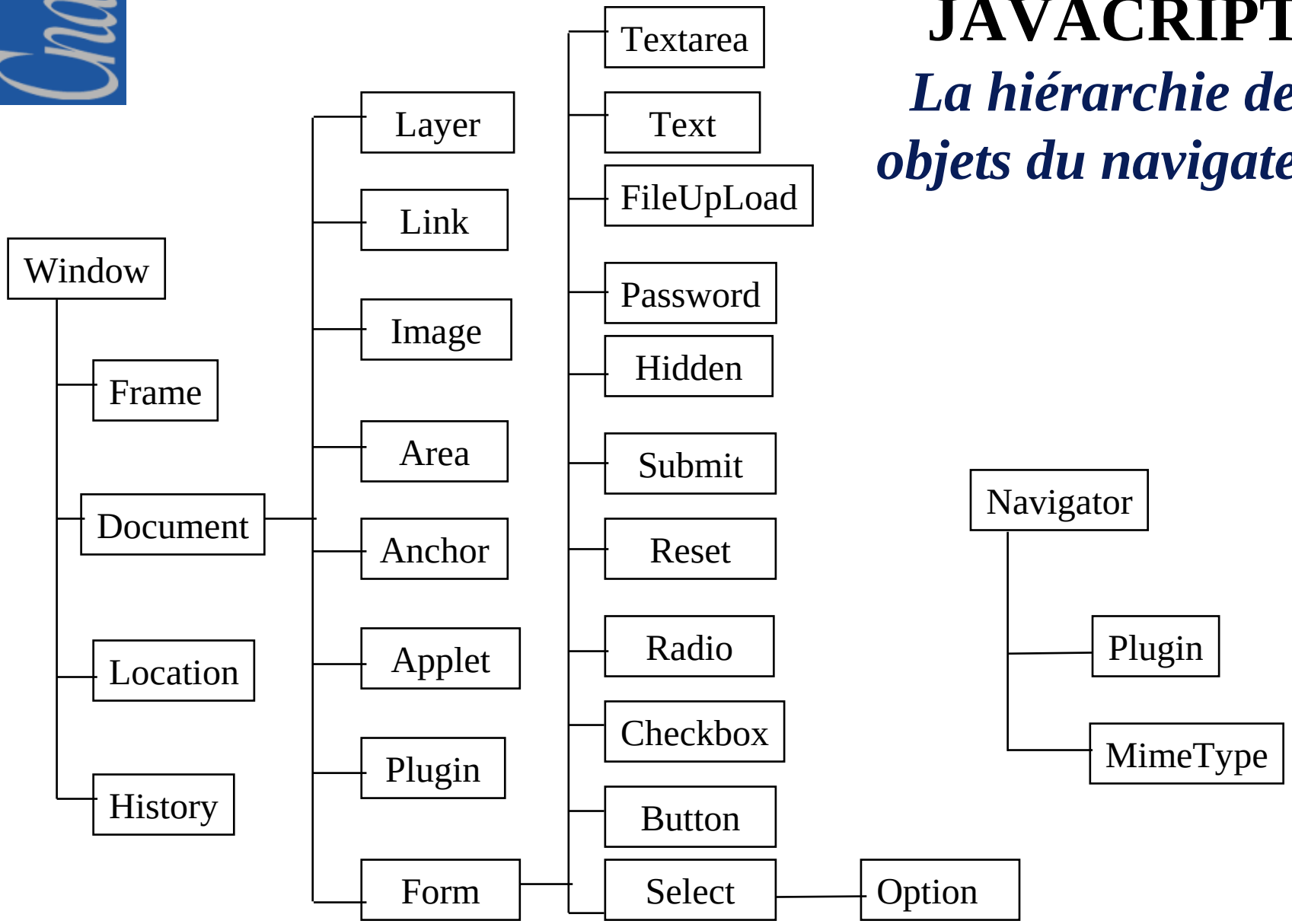
  - **les objets issus du document html : ou objets du navigateur**
  - **les objets prédéfinis dans Javascript**
  - **les objets créés par l'utilisateur**

JAVACRIPT - *Les objets du navigateur (1)*

- **Javascript permet de travailler sur des objets :**
  - issus du document html (les objets du navigateur)
    - **organisés en une hiérarchie**
  - avec en général
    - **un nom (attribut NAME)**
    - **des propriétés**
      - qui ont une valeur (VALUE)
      - accessibles par des tableaux (array)
    - **des méthodes**

JAVACRIPT

La hiérarchie des objets du navigateur



JAVACRIPT - *Les objets du navigateur (2)*

- **navigator: name, version, MIMEtype supportés et plugins**
- **window : propriétés s'appliquant sur la fenêtre entière**
  - **il y a des objets window pour chaque fenêtre window fille dans un document à frames**
- **document : propriétés basées sur le contenu du document (title, background,color,link,forms)**
- **location : propriétés basées sur l'URL courante**
- **history :propriétés représentant la navigation (URL) du client**

on se réfère à une propriété par le nom de l'objet et des ancêtres :

document.myform.texte1.value : *valeur du texte (input de NAME=texte1) de la form (de NAME=myform) du document*

JAVACRIPT - *Exemples d'objets*

Soit le code :

```
<HTML> <HEAD> <TITLE>exemple d'objets</TITLE>
<SCRIPT>.....</SCRIPT>   </HEAD>
<BODY>
<FORM NAME="maforme "ACTION=npq.cgi" METHOD="get">
 <INPUT  TYPE="text" NAME="texte1" VALUE="blablabla" SIZE=20>
 <INPUT  TYPE="text" NAME="texte2" VALUE="machina" SIZE=20>
.....
</FORM> </BODY></HTML>
```

D'où les propriétés et leur valeur :

document.title	*"exemple d'objets"*
document.bgcolor	*#ffffff*
document.maforme.method	*"get"*
document.maforme.texte1.value	*"blablabla"*
document.maforme.texte1.name	*"texte1"*

si fond blanc défini par le browser

JAVACRIPT - *L'objet de base WINDOW (1)*

Les méthodes de l'objet window (frame idem)

- open et close : ouverture et fermeture d'une fenêtre dans le browser
- alert : boîte de dialogue avec message d'alerte
- confirm : boîte avec bouton OK et Cancel
- prompt : boîte avec champ texte pour entrer une valeur
- blur et focus : enlève ou donne le focus à une fenêtre
- scrollTo : scrolling jusqu'à une valeur spécifiée de coordonnée
- setInterval : appel de fonction ou évaluation d'expression périodiquement
- setTimeout : appel de fonction ou évaluation d'expression après un temps donné

JAVACRIPT - *L'objet de base WINDOW (2)*

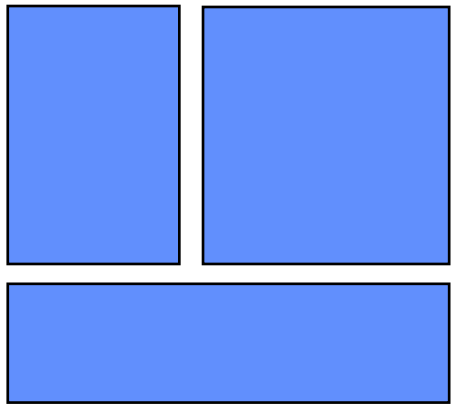
Javascript permet enfin de manipuler les fenêtres

- **ouverture de fenêtre : méthode open**
  window.open("docu.hml") ou
  window.open("http://lion.cnam.fr","fenaffiche");
  « fenaffiche » sera le nom de TARGET utilisable par une autre fenêtre

- **on peut aussi fermer une fenêtre par close**
  mafenetre.close(), self.close() ou ***window.close()***

- **un FRAMESET est composé d'un ensemble de FRAMES**

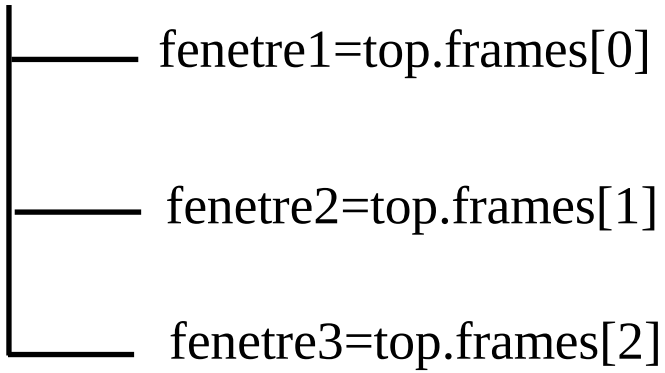
```
<FRAMESET ROWS="60%,40%">
  <FRAMESET COLS="30%,70%">
    <FRAME SRC="fen1.html"NAME="fenetre1">
    <FRAME SRC="fen2.html" NAME="fenetre2">
  </FRAMESET>
<FRAME SRC="fen3.html" NAME="fenetre3"> </FRAMESET>
```



JAVACRIPT - *L'objet de base WINDOW (3)*

top ou parent:fenetre mère du navigateur

Ceci correspond à:



On pourra ainsi d'une fenêtre accéder à d'autres, par exemple :

<A HREF="URL" TARGET="nomde frame"> ancre </A>
pour charger un nouveau document URL dans une frame

<FORM Action="URL-CGI" TARGET="nomde frame">..</FORM>
pour afficher les résultats dans une frame particulière

JAVACRIPT - *L'objet de base WINDOW (4)*

on peut (à la création ou en modification par Javascript) agir sur le look :

fen1= window.open ("url.html", "fenimage", "scrollbars=yes, toolbar=no, menubar=yes, location=no, directories=no, status=no")

crée une nouvelle fenêtre du browser de nom fen1, contenu url.html accessible en TARGET:fenimage, avec des ascenseurs, barre de menus, sans barre d'outils ni status, ni zône location

fen1.toolbar = visible permettra de rajouter la barre d'outils.

JAVACRIPT - *L'objet de base DOCUMENT*

write et writeln : méthodes générant du HTML

- **propriétés:**
  - **bgColor, fgColor, linkColor, alinkColor, vlinkColor**
  - **LastModified (cf http)**
  - **referrer: URL précédemment visitée par le client**
  - **Cookie permettra de positionner des valeurs ou de les récupérer.**

cookie : ***mécanisme pour stocker des données persistantes sur le client (fichier cookies.txt)***

JAVACRIPT - *L'objet de base FORM*

- **chaque form d'un document créé un objet form.**
- **Les objets form d'un document sont stockés dans un tableau forms (à partir de forms[0])**
- **les éléments du formulaire sont eux-même dans un tableau elements (à partir de de elements[0])**

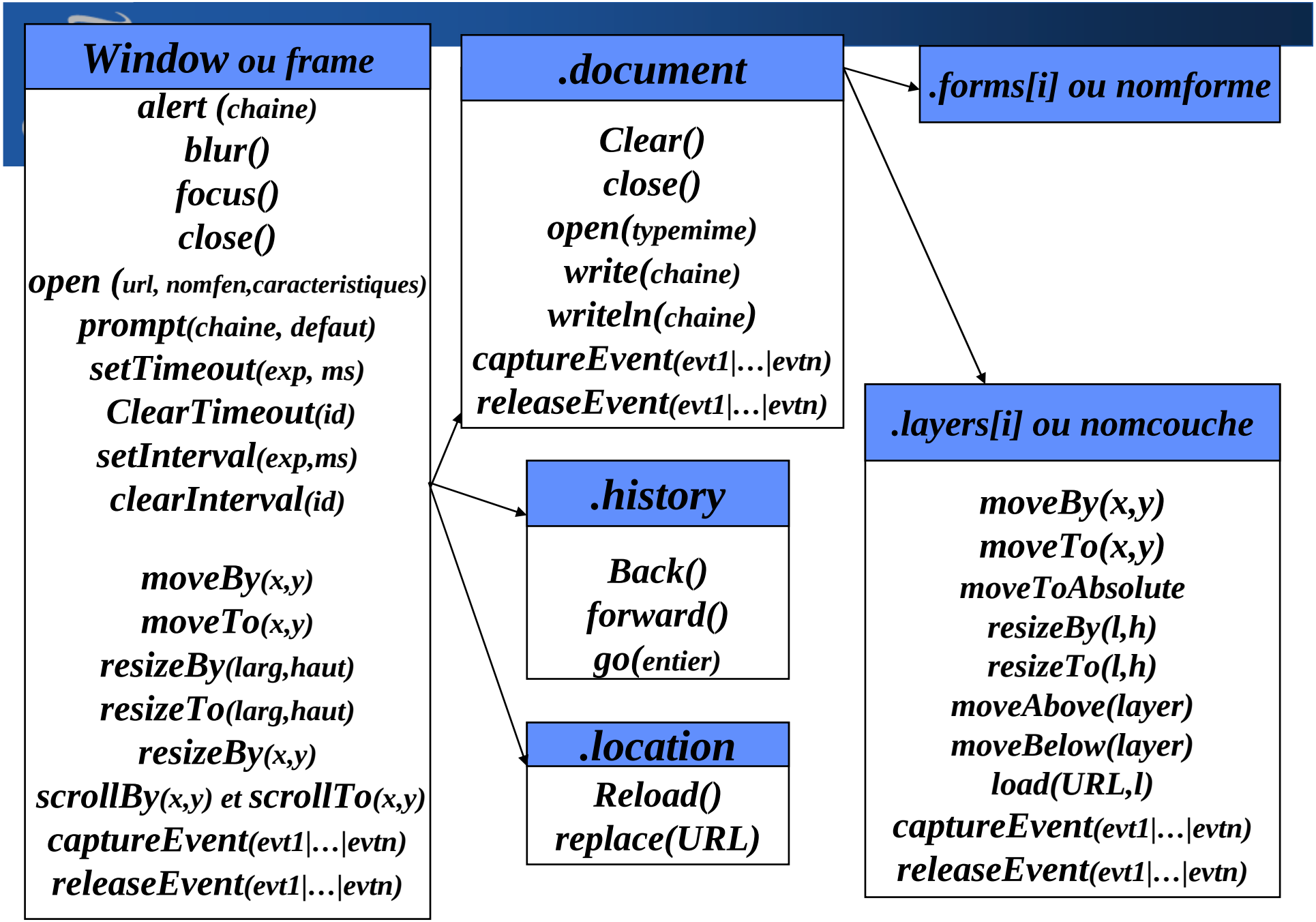
  méthodes des objets form : submit() et reset()

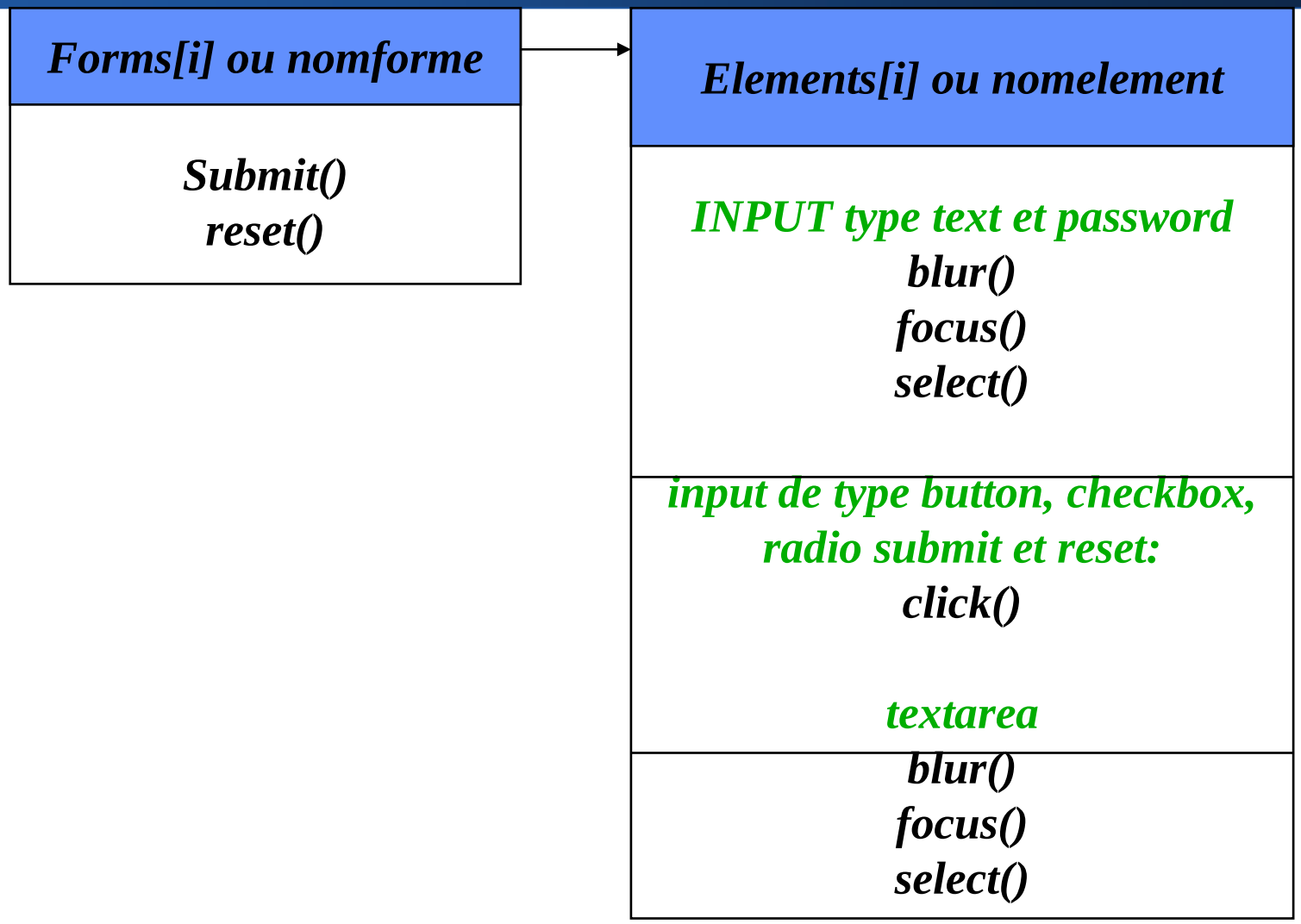
JAVACRIPT - *Objets de base*

- **Objet de base LOCATION :**
  - **propriétés basées sur l'URL courante:**
  - **par exemple : hostname**
    - méthodes :
      - replace() : pour charger une URL
      - reload() : force le rechargement du document

- **Objet de base HISTORY :**
  - **contient la liste des URL visitées. (tableau history)**
  - **propriétés: current , next , previous**
  - **méthode: go** ***ex : history.go(-2)***

- **Objet de base NAVIGATOR :**
  - **informations sur le navigateur : appName**
    appVersion
    - propriétés: javaEnabled
    - mimeType et plugin (dans tableau plugins et mimeTypes)


Window ou frame
alert (chaine)
blur()
focus()
close()
open (url, nomfen,caracteristiques)
prompt(chaine, defaut)
setTimeout(exp, ms)
ClearTimeout(id)
setInterval(exp,ms)
clearInterval(id)
moveBy(x,y)
moveTo(x,y)
resizeBy(larg,haut)
resizeTo(larg,haut)
resizeBy(x,y)
scrollBy(x,y) et scrollTo(x,y)
captureEvent(evt1|…|evtn)
releaseEvent(evt1|…|evtn)
.document
Clear()
close()
open(typemime)
write(chaine)
writeln(chaine)
captureEvent(evt1|…|evtn)
releaseEvent(evt1|…|evtn)
.forms[i] ou nomforme
.history
Back()
forward()
go(entier)
.location
Reload()
replace(URL)
.layers[i] ou nomcouche
moveBy(x,y)
moveTo(x,y)
moveToAbsolute
resizeBy(l,h)
resizeTo(l,h)
moveAbove(layer)
moveBelow(layer)
load(URL,l)
captureEvent(evt1|…|evtn)
releaseEvent(evt1|…|evtn)



window.document.forms[1].elements[1].focus() met le focus sur le 2eme élément du deuxième formulaire de la page

JAVACRIPT - *Les tableaux d'objets du navigateur*

document	**anchors**	**les tags <A> du document ayant un NAME, dans l' ordre**
	applets	**les tags <APPLET> dans l'ordre (du source)**
	embeds	**les tags <EMBED> dans l'ordre (du source)**
	forms	**les tags <FORM> dans l'ordre (du source)**
	images	**les tags <IMG> dans l'ordre (du source)** *(pas celles créées par image)*
	links	les tags <AERA HREF=....>,<A HREF=....> et les objets créés par link , **dans l'ordre (du source)**
Function	**arguments**	les arguments d'une fonction
Form	**elements**	les éléments (objets *checkbox,radio et text* **dans l'ordre (du source)**
select	**options**	les tags <OPTIONS> dans l'objet Select dans l'ordre
window	**frames**	les tags <FRAME> d'une fenêtre contenant un FRAMESET dans l'ordre du source
navigator	**mimeTypes**	tous les MIME types supportés par le browser de façon interne, via helpers ou par plugins
	plugins	les plugins installés sur le browser client, dans leur ordre

JAVACRIPT - *Fonctions et objets prédéfinis*

Il existe des méthodes prédéfinies sur des objets prédéfinis dans Javascript.

- **objets prédéfinis avec méthodes associées :**
  - **Array**
  - **Boolean**
  - **Date**
  - **Function**
  - **Math**
  - **Number**
  - **RegExp**
  - **String**

JAVACRIPT - *Les fonctions (1)*

- **Ensemble d'instructions correspondant à une tâche spécifique, appelable par leur nom.**
- **Définition:**

  function nomdefonction(paramètres,arguments) {

  bloc d'instructions

  }
- **Passage de paramètres:**

  function ecrirenom(name) {

  document.write("<HR votre nom est<B>");

  document.write(name);

  document.write("</B></HR>");

  }

JAVACRIPT - *Les fonctions (2)*

- **Une fonction peut retourner un résultat par *return***
- **NB:**
  - on déclarera les fonctions dans la partie HEAD pour être sur que le navigateur les connaissent.
  - ***name*** est une variable locale à la fonction et n'existe que lors de l'appel par opposition aux variables déclarées à l'extérieur qui sont globales et accessibles partout.
  - possibilités de fonctions récursives
  - on pourra ainsi définir un handler d'événement

- **Javascript fournit aussi un jeu de base de fonctions prédéfinies (comme un ensemble de méthodes de base (méthode = fonction associée à un objet))**

JAVACRIPT - *Les fonctions (3)*

Les fonctions prédéfinies:

- **eval(expression):** *eval permet l'évaluation de la chaine expression*
  par exemple si expression:instructions Javascript, eval les exécute.
  ==> inutile pour les expressions arithmétiques
- **isNaN(valeurtest) : NaN:not a number donne true *(si NaN)* ou false *(si Number)***
- **parseFloat(string) et parseInt(string[,base]) transforme une chaine en valeur numérique (flottant ou entier selon la base demandée)**
- **escape(string) : encodage de chaine d' ISOlatin character set en hexadécimal et unescape(string) retourne l'ASCII**
- **taint et untaint: marquage pour sécurisation**
- **number(objet) et string(objet) : convertissent un objet en nombre ou en chaine lisible :** ***<SCRIPT> D= new Date(430054663215);***
  ***document.write(string(D)+"
");***
  </SCRIPT> donne **THU Aug 18 04:37:43 Pacific Dayligt Time 1983**

- enfin,....il existe aussi des objets Function créés par new...(compilés) à partir de javascript 1.1

JAVACRIPT - *Les objets de l'utilisateur (1)*

- **Outre les objets prédéfinis de javascript, les objets du navigateur, on peut créer de nouveaux objets manipulables par javascript**

- **Il importe de définir l'objet, ses propriétés et les méthodes (fonctions associées) portant sur cet objet**

- **les méthodes et propriétés sont accessibles par :**

  nomobjet.nommethode **et** *nomobjet.nompropriete*

JAVACRIPT - *Les objets de l'utilisateur (2)*

Création d'un objet par une fonction *constructor function* qui définit les noms et propriétés:

exemple: function constructor:

```
fonction stagiaire (nom,prenom,age) {
   this.nom=nom;
   this.prenom=prenom;
   this.age=age;                          this:objet courant
}
```

on créera une instance de l'objet par :

stagiaire1=new stagiaire ("dupond","charles",24) *où l'objet stagiaire1 a les propriétés:* stagiaire1.nom,stagiaire1.prenom et stagiaire1.age.

On peut créer d'autres instances par un autre new qui créera un autre objet:
stagiaire2=new stagiaire ("martin","louis",19)

JAVACRIPT - *Les objets de l'utilisateur (3)*

Pour définir une méthode pour les objets, on va créer une fonction que l'on insérera dans le constructor:

```
function total(){
total=eval(this.notes.info+this.notes.math);
document.write("<HR><P>total"+total); }
```

- ***et on définira la function constructor***

```
function stagiaire (nom,prenom,notes) {
  this.nom=nom;  this.prenom=prenom;  this.notes=notes;
 this.total=total; }
```

- ***et on pourra appeler la méthode :***

```
stagiaire1.total
```

NB: on peut définir dynamiquement les méthodes sur un objet créé par *stagiaire1.nommethode=nomfonction*

JAVACRIPT – *Compléments (1)*

JAVASCRIPT permet de modifier les propriétés des objets d'un document :

document.tags.H1.color="red" ***met la propriété color de l'objet de propriété H1 de l'objet de propriété tags du document à red*** *(document peut être omis)*

ou

<STYLE TYPE="text/javascript"> tags.P.fontSize="18pt";
tags.P.marginLeft="20pt";
<STYLE>

définit un style pour les paragraphes **P**

les noms de propriétés:margin-right (css)devient marginRight(javascript)

JAVACRIPT - *Compléments (2)*

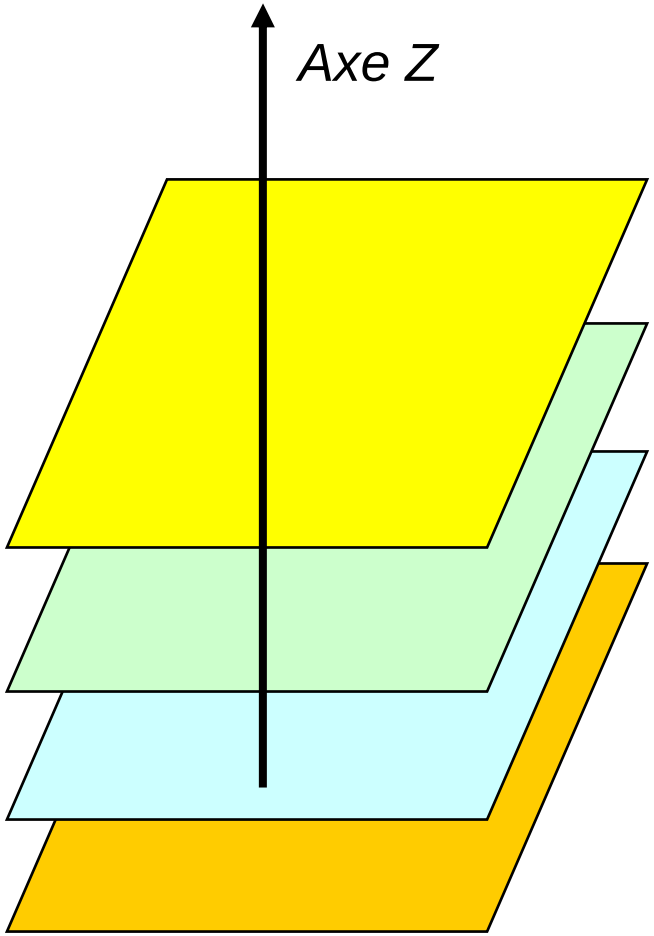
Pour positionner les couches de manière dynamique :

L'affichage devient un empilage de couches dont on peut spécifier la position dans la fenêtre du browser et l'ordre d'empilement.

On pourra ensuite réaliser par la

- **modification des propriétés *(visibility, bgcolor,src,zindex…)***
- **l'appel aux méthodes des couches (*move,resise, moveAbove et moveBelow, load)***
- **aux événements *(focus, blur, load, mouseout et mouseover)***

de véritables animations s'exécutant sur le poste browser



JAVACRIPT - *Compléments (3)*

Les fontes téléchargeables

Javascript permet d'incorporer des fontes dans un document :

- **grâce à un fichier de fontes (comme une image gif ou jpeg) par LINK avec REL=FONTREF et SRC : accès au fichier**
- **un fichier de fonte : type MIME (paramétrage httpd nécessaire)**