Web, HTTP and HTML

Dritan Nace

Web, HTTP and HTML

- The Web : general presentation
- URL and the HTTP protocol
- HTML, examples
- CSS style pages, examples

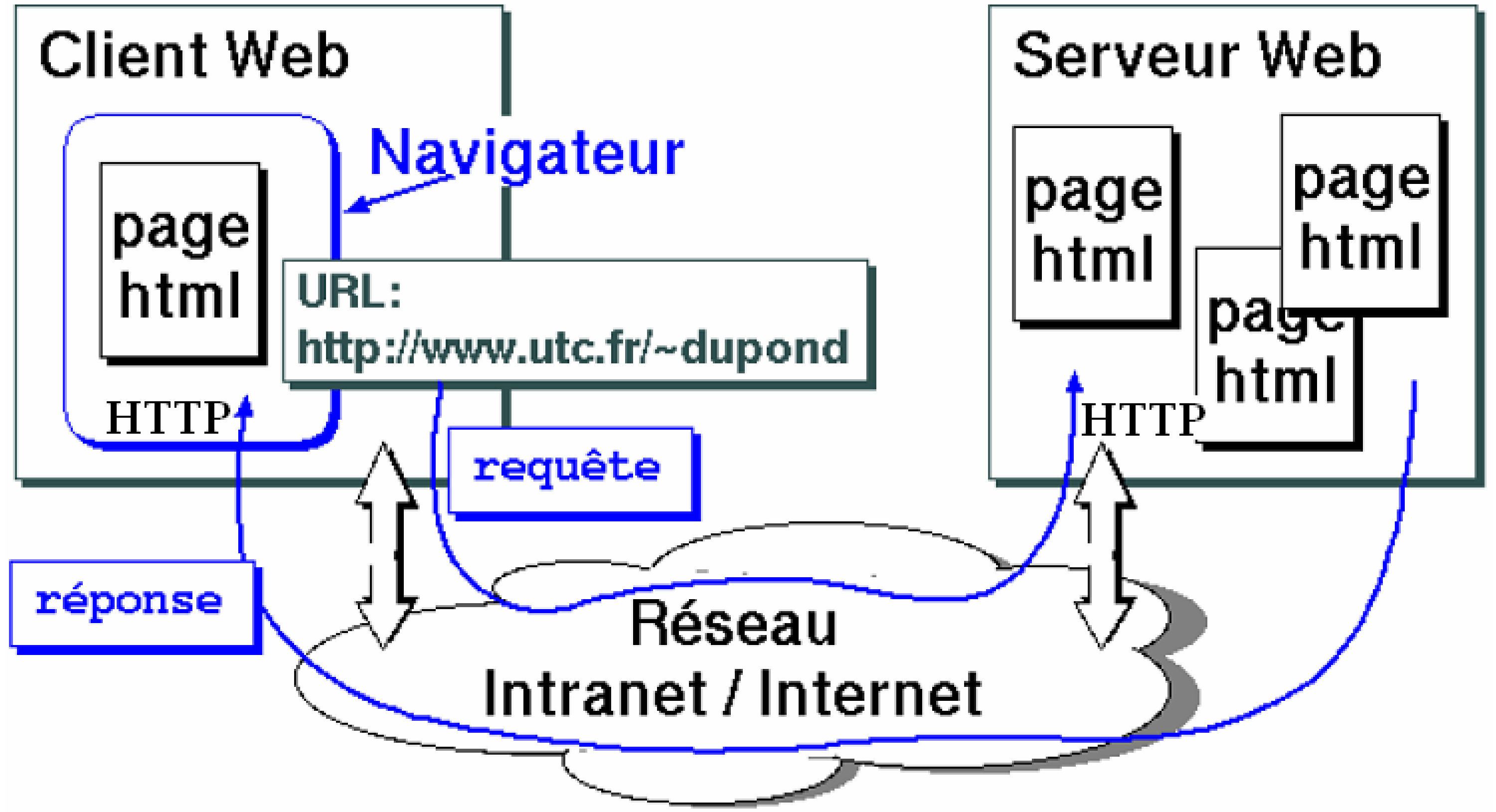
General introduction to the Web

- **Web = shared hyper-media information system :**
  - **text, images, animated images, sound …**
- **Information is stored on servers :**
  - **www.orange.fr**
  - **www.cisco.fr**
  - **..**
- **To interact witht the server, clients such as Netscape, Internet Explorer, …**
- **These are the clients which translate the person’s requests into HTTP requests, to interact with the servers**

General web introduction

- The Web is composed of the following general parts:
  - HTML (**HyperText Markup Language**)
  - URL (**Uniform Resource Locator)**
  - HTTP (**HyperText Transfer Protocol**)

Path of a connection (I)



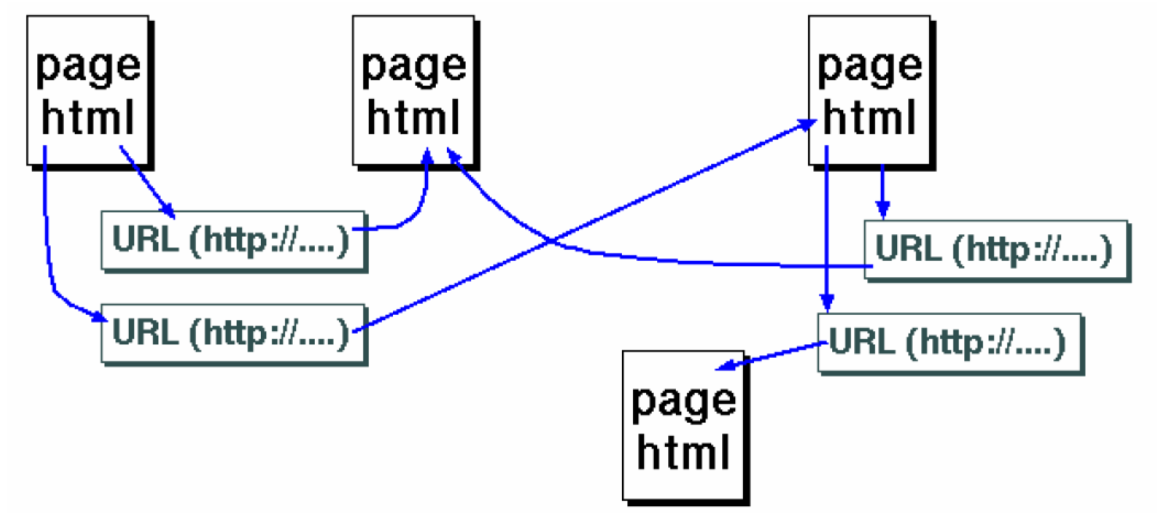
Path of a connection (II)

How is a Web page obtained ?

- **The user clicks on a web link or types a web address;**
- **The browser (which is the HTTP client) then creates a request to send to the Web server;**
- **The HTTP application passes this request to TCP;**
- **TCP adds its header in which the client port number is present, and the port server number (here, 80), and then transmits all of this (segment) to the network layer, or IP;**
- **IP adds its header which contains the source and destination;**
- **IP transmits all of this (packet) to the physical;**
- **The physical layer adds its header (eg. Ethernet) to get a frame, and sends the data (bits) on the physical support**;

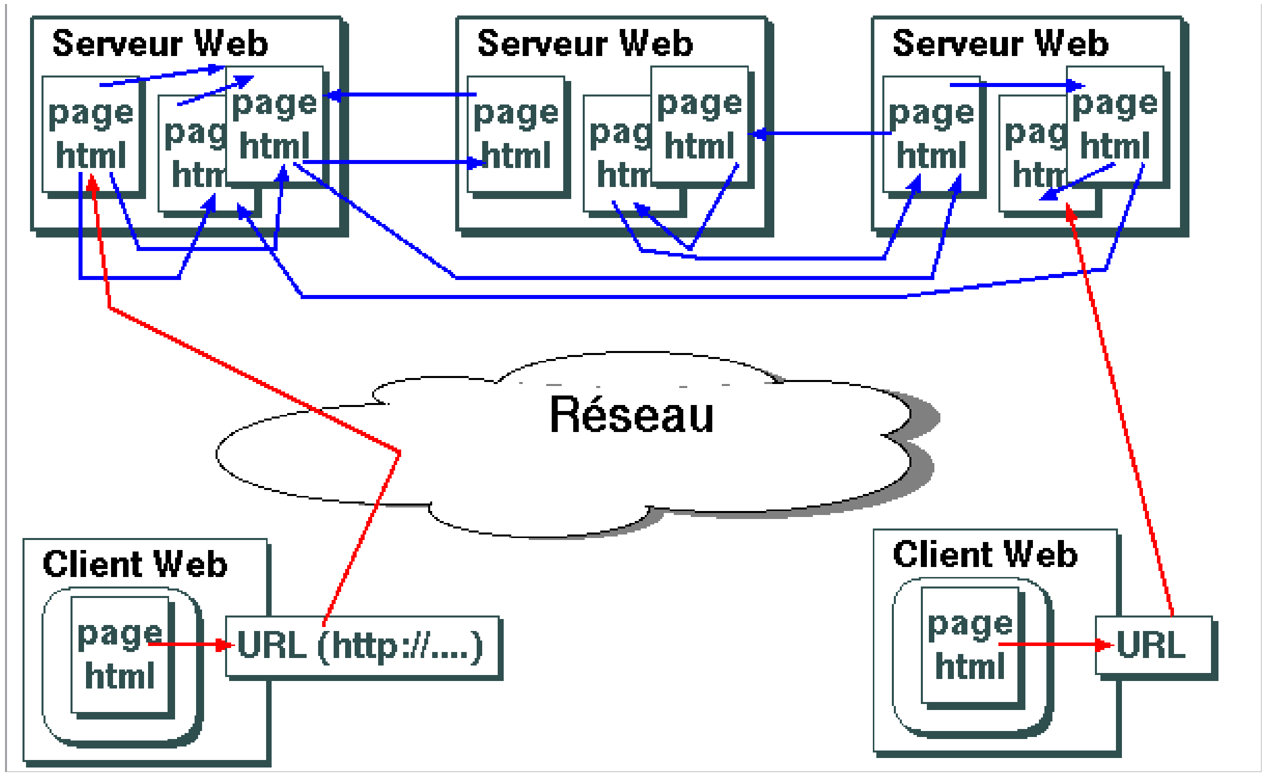
Web: general aspects, navigation between web pages

- WEB : link documents via a global network (1989)
- The web pages are written in HTML = *Hypertext Markup Language*
- Hypertext : all the documents linked by « hyperlinks »

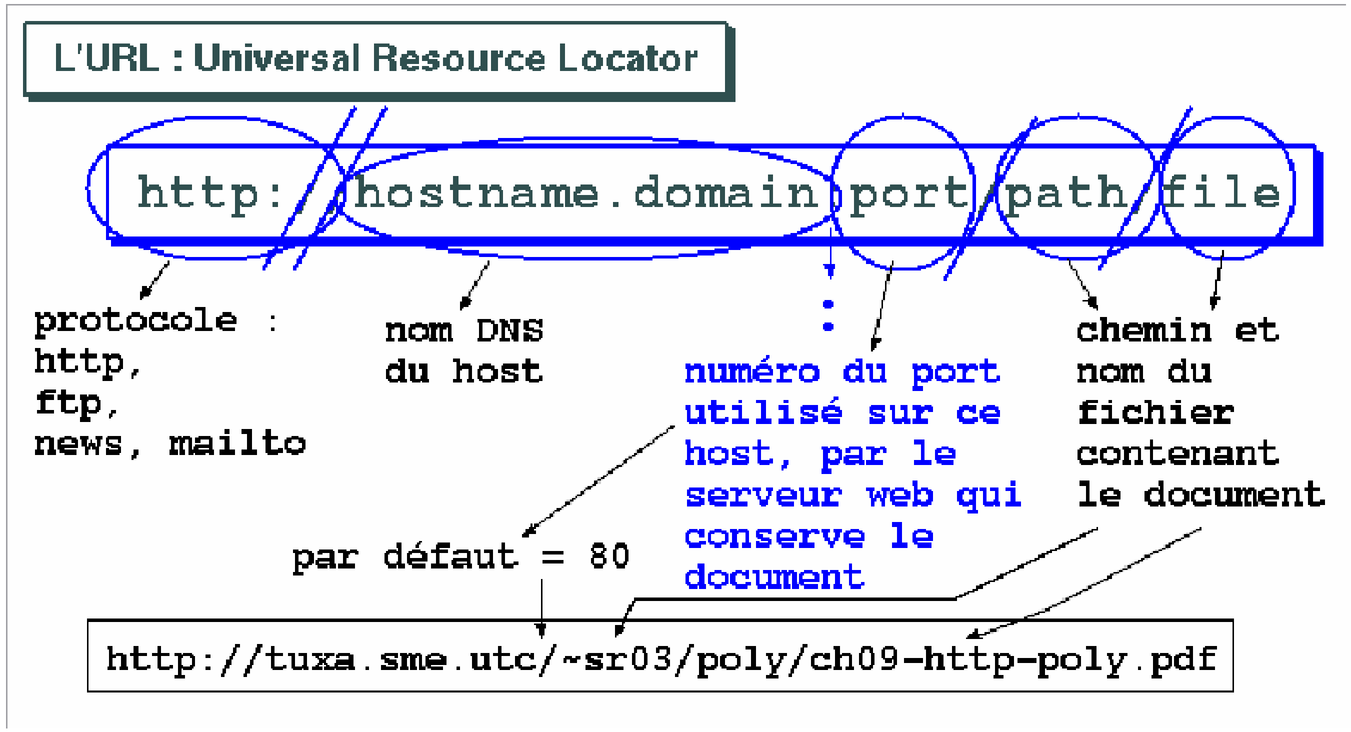


Partager les documents en les reliant entre eux et en cachant la méthode d'accès et la localisation.

Web


Serveur Web
page html
pag htm
page html
Serveur Web
page html
pag htm
page html
Serveur Web
page html
pag htm
page html
Réseau
Client Web
page html
URL (http://....)
Client Web
page html
URL

URL, URI, URN…


L'URL : Universal Resource Locator
http://hostname.domain:port/path/file
protocole : http, ftp, news, mailto
nom DNS du host
:
numéro du port utilisé sur ce host, par le serveur web qui conserve le document
par défaut = 80
chemin et nom du fichier contenant le document
http://tuxa.sme.utc/~sr03/poly/ch09-http-poly.pdf

URL : Uniform Ressource Locator
URI : Uniform Ressource Identifier
URN : Uniform Ressource Name

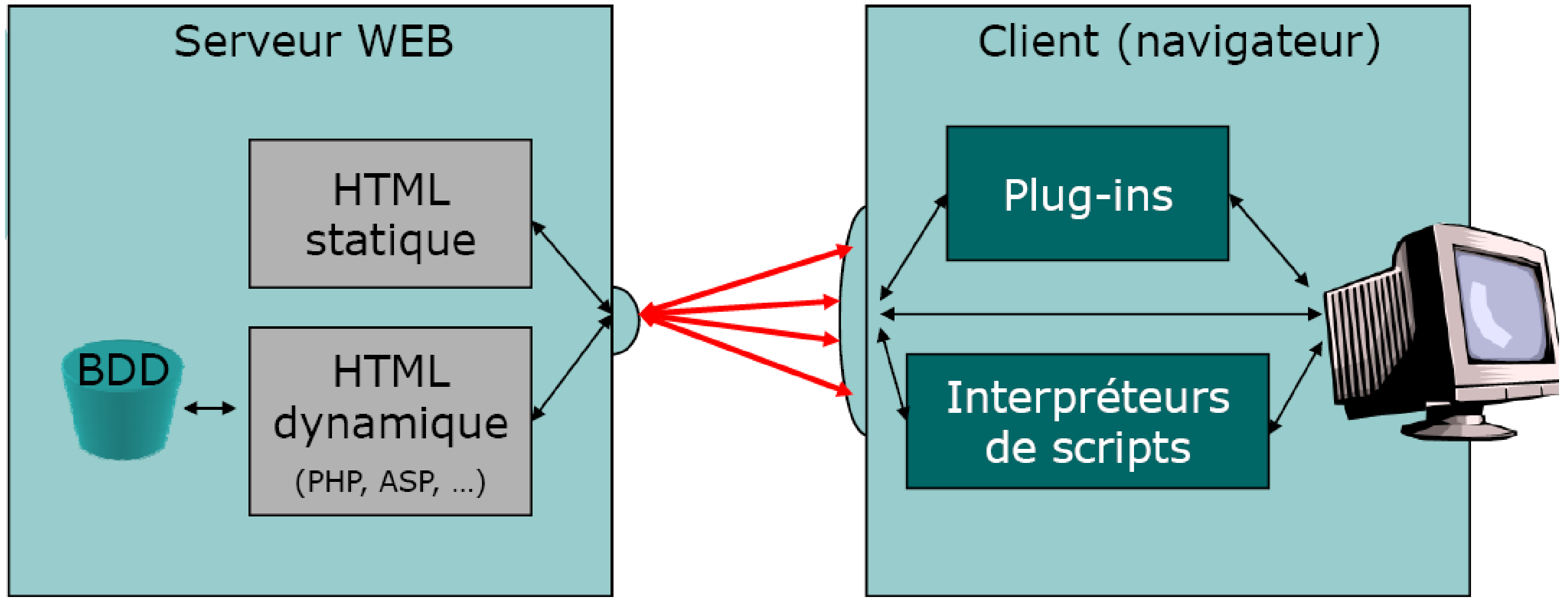
Web and HTTP

- **Web operates on a client-server model**
  - **The client sends requests to the server :**
  - **File transfer request**
  - **Program execution on the server**
  - **File upate**
  - **…**
- **Use of the HTTP protocol**
  - **Defines the syntax used for the exchanges between the client and web server**
- **The objects are recognised by their URL**
- **No permanent session between client and server**

HTTP

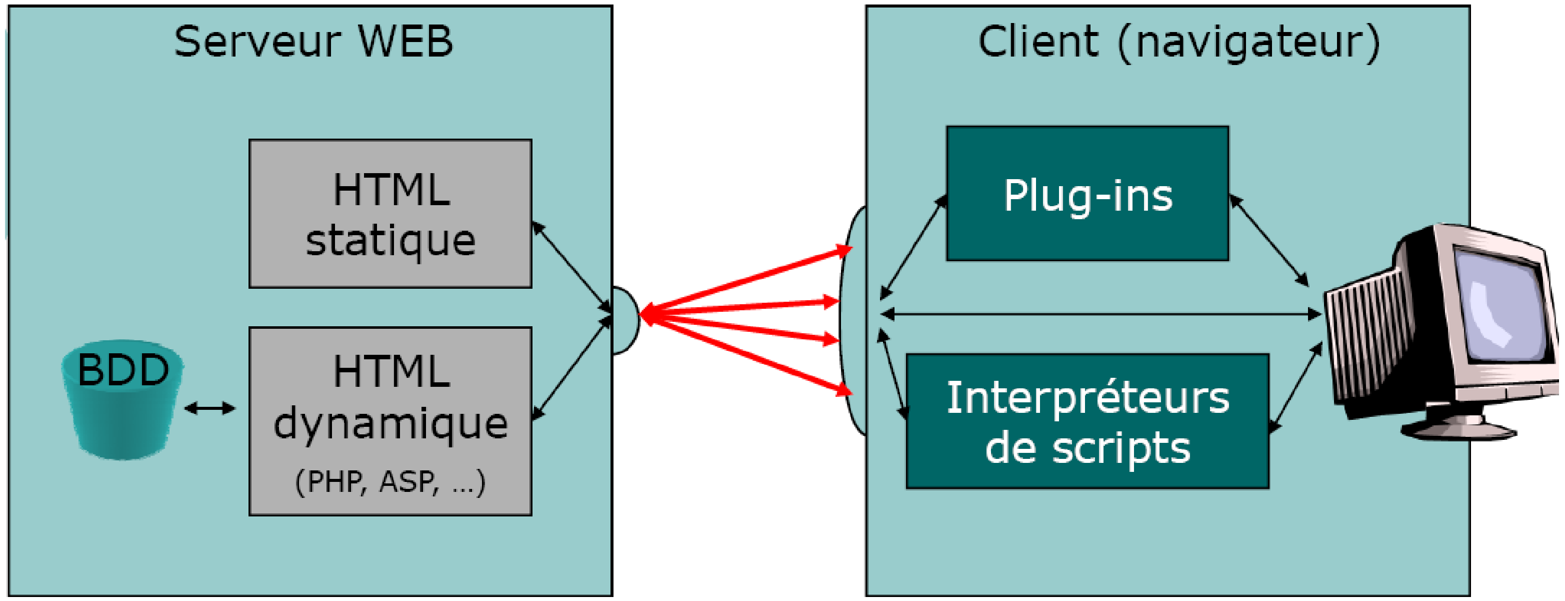
- HTTP is a « memoryless » protocol
    - No notion of a session with HTTP
    - Each communication is independant.
- Cookies
    - Information is stored with the client, concerning browsing.
    - example :
        - Password identification
        - Authentification : creation of a cookie
        - Come back to the page : read a cookie
        - Deconnection or end of the lifecycle of the cookie : destruction

HTTP



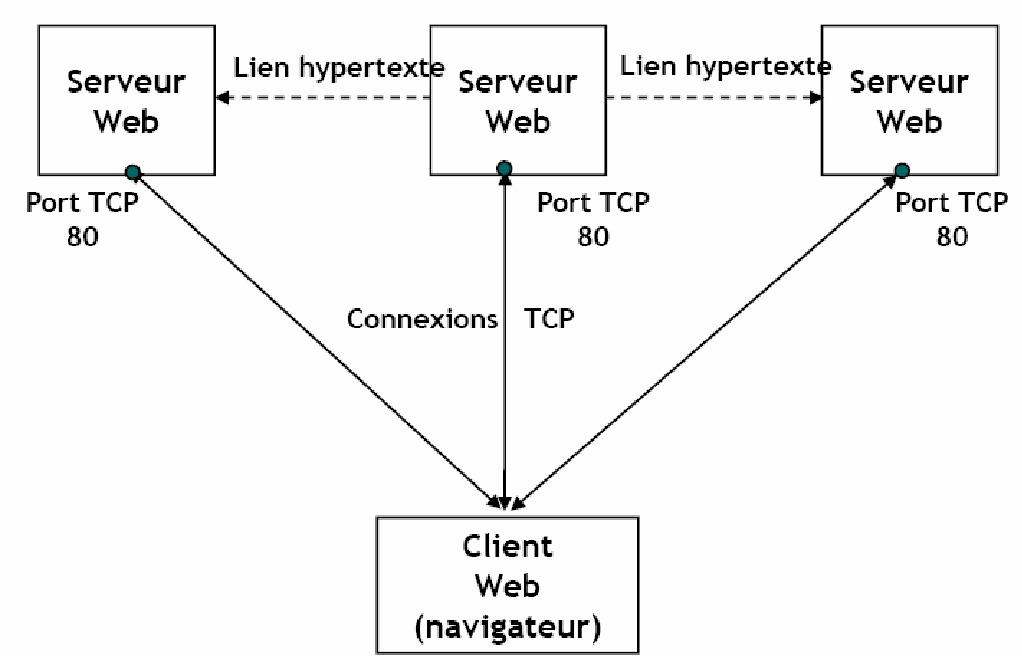
- Server side
  - Static HTML
  - Dynamic HTML
  - Calculation from elements which the client transmits to the server in its request.
  - Information from a database updated by any means.
    - Creation of HTML code to send to the client!

HTTP



- Client side
  - Check the validity of the information entered into a form, before sending them to the server
  - Local processing of certain information to display a result.
    - animations…

HTTP : mechanisms



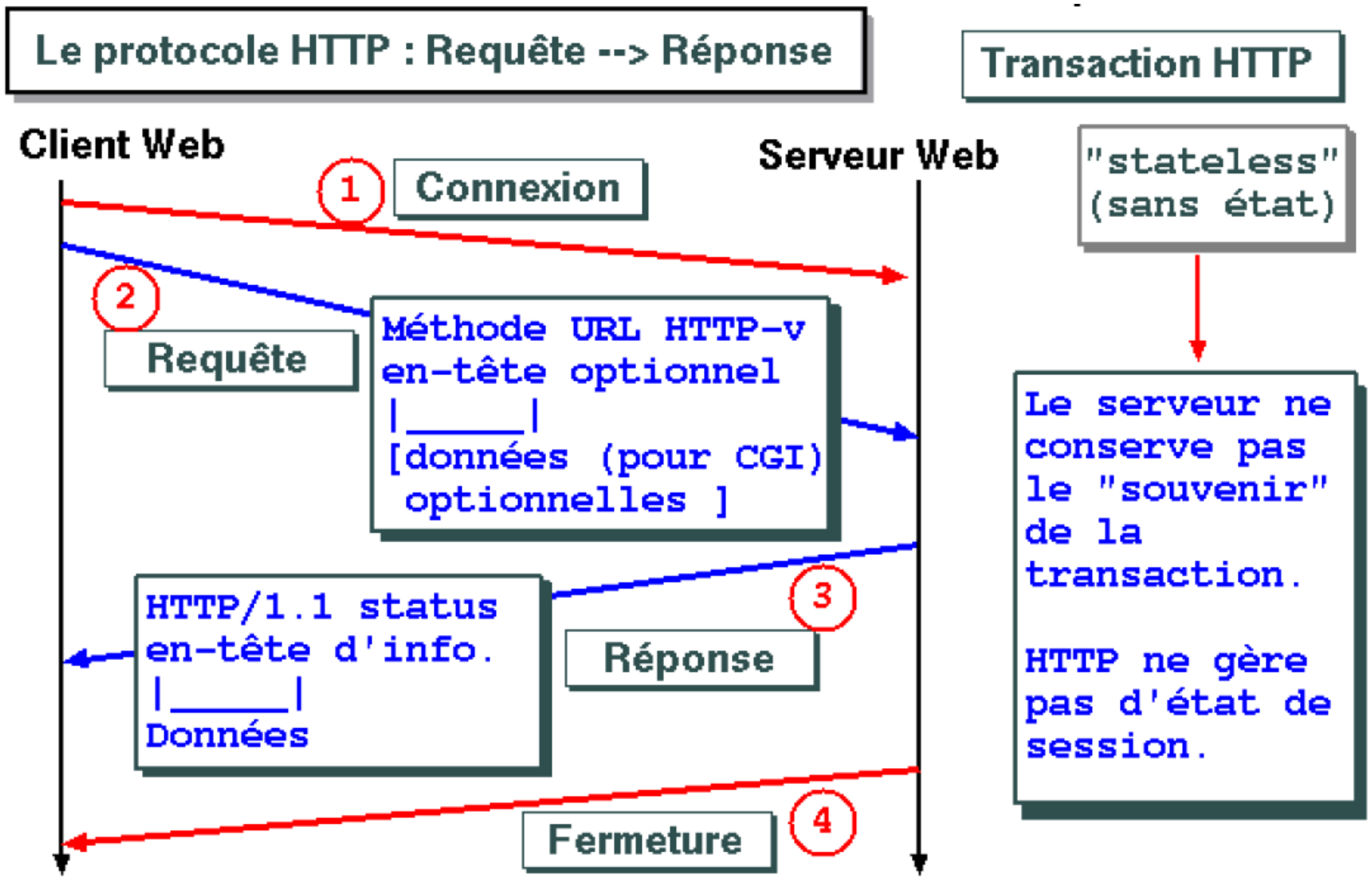
HTTP : history

- The protocol is normalised (RFC 1945 and RFC 2616) which defines three versions :
  - 0.9 which is the same protocol origianally defined by Tim Berners-Lee,
  - 1.0 which adds numerous features (document typing, content coding, identification, ...)
  - 1.1, which is the most advanced, allows different negotiations, can obtain persistent channels etc…

HTTP

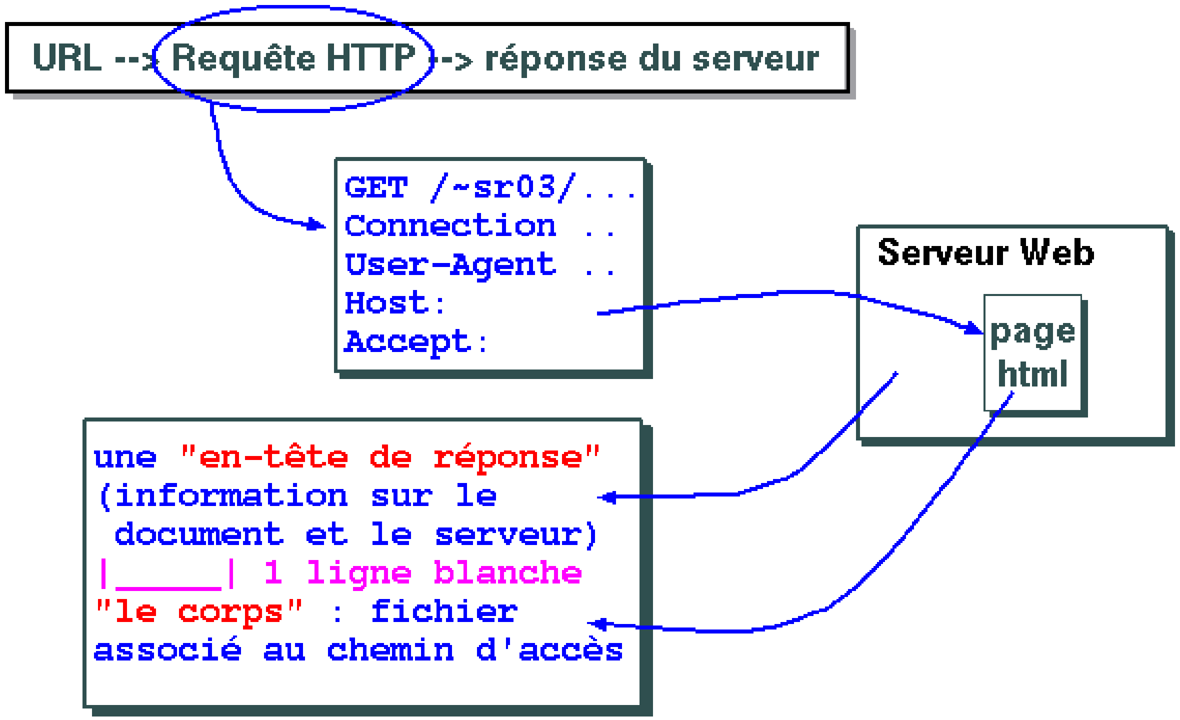
- HTTP only contains text
  - Coding of binary data → text, images, documents, etc…
- The browser (the client) communicates with the Web server via one or several TCP connections.
- The port of public notoriety of a Web server is 80
  - HTTP is a very simple protocol :
    - The client establishes a TCP connection with the server
    - It sends a request and gets the response of the server
    - The server ends the response by closing the TCP connection

HTTP : how does it work ?



Lifecycle of an HTTP connection

HTTP



URL --> Requête HTTP --> réponse du serveur
GET /~sr03/...
Connection ..
User-Agent ..
Host:
Accept:
Serveur Web
page html
une "en-tête de réponse" (information sur le document et le serveur)
|_____| 1 ligne blanche
"le corps" : fichier associé au chemin d'accès

There are two types of HTTP messages : requests and responses

HTTP

- The URL is provided to the browser,
- The browser analyses the URL and sends a request to the web server.
- For example : http ://tuxa.utc.fr/sr03/td1.html sends the following to the server :

```
GET /sr03/td1.html HTTP/1.1
Connection : Keep-Alive
User-Agent : Mozilla
Host : tuxa.utc.fr
Accept-Charset : iso-8859-5, unicode-1-1
Accept : image/gif, image/x-xbitmap, image/jpeg, */*
```

HTTP 1.0 requests

- The HTTP request :
  - *request line*
  - *headers (0 or more)*
  - *<empty line>*
  - *body of the request (for POST only)*

- The format of the request line is :
  - *URL request – required HTTP - version*

```
GET /sr03/td1.html HTTP/1.0
```

HTTP 1.0 requests

- There are three distinct requests :
  - The **GET** request, which returns the content of the required URL (if it exists)
  - The **HEAD** request, where the server returns the header only (verifies accessibility in particular)
  - The **POST** request, which enables sending data to the server (forms…)

Simple requests

HTTP : Requêtes simples

GET

```
Le client veut juste
demander un document:
GET /chemin/doc.htm HTTP/1.1
```

HEAD

```
Le client veut juste
de l'information sur
un document, mais pas
le document lui-même :
HEAD /chemin/doc.htm HTTP/1.1
```

```
Le client
veut
seulement
récupérer
de
l'information
```

Requests with data sending

HTTP : Requêtes avec envoi d'informations

POST

On fournit de l'information au serveur
(par ex. issue d'un formulaire) par:

```
POST /chemin/cgi-bin/form.php HTTP/1.1
en-tête
|_____| ligne blanche
var1=data1&var2=data2&....
```

GET + infos

```
GET + Format "URL encodé"
<FORM>...method=GET>
```

l'info. saisie dans la forme est passée au serveur par:

```
  GET /ch.../form.php?var1=data1&var2=data2 HTTP/1.1
```

fichier ? var = data & var = data ...

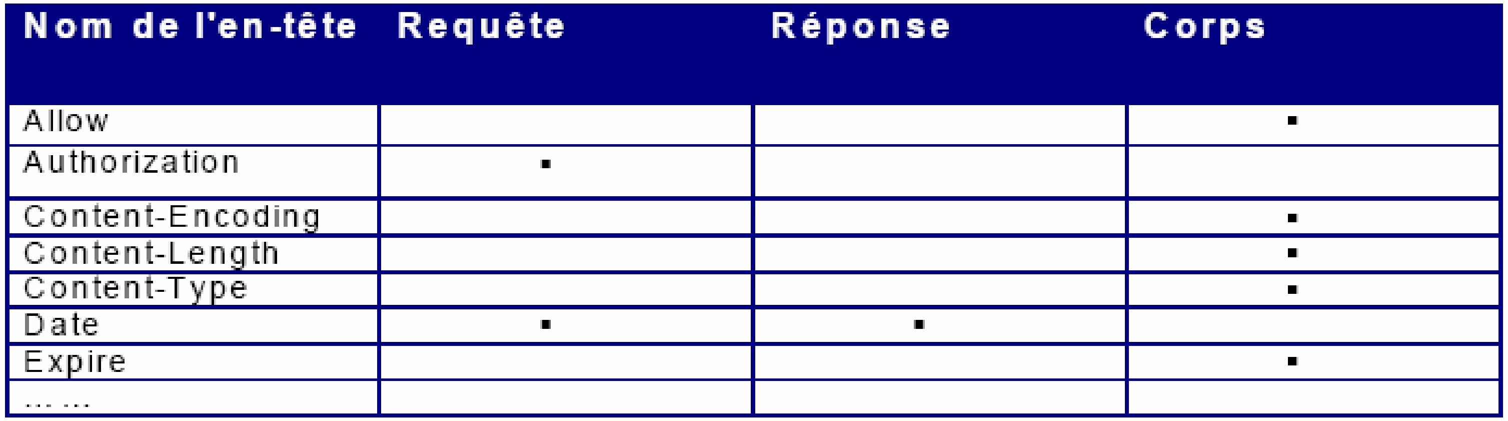
Responses

- An HTTP response is has the following form :
  - status line
  - header (0 or more)
  - <line break>
  - reply body (contains the required document)
- The status line is has the following form :
  - HTTP-version code-response sentence-response

Headers

- Requests and responses can contain a certain number of header fields
- A line break allows the separation of the latter from the document
- A header is separated into a field name, a ‘:’ caracter, a space and a field value
  - The headers are classed into four categories
    - General HTTP headers;
    - Those which apply to requests;
    - Those which apply to responses;
    - Those which correspond to the body of the message.

Headers



Nom de l'en-tête	Requête	Réponse	Corps
Allow			▪
Authorization	▪		
Content-Encoding			▪
Content-Length			▪
Content-Type			▪
Date	▪	▪	
Expire			▪
… …			

General HTTP headers
Cache-Control.
Connection = lists of options (*close to end a connection*).
Date = current date.
MIME-Version = MIME version used.
Pragma = instruction for the proxy.
Transfer-Encoding = type of the transformation applied to the body of the message.
Via = used by proxys to indicate the machines and intermediary protocols.

Headers of client requests

- Headers of client HTTP requests
  - Accept = MIME type visible by the agent
  - Accept-Encoding = accepted coding methods
    - compress, x-gzip, x-zip
  - Accept-Charset = client’s prefered set of characters
  - Accept-Language : list of languages
    - fr, en, …
  - Authorisation : Identification of the browser by the server
  - Cookie = returned cookie
  - Content-Length = length of the body of the request

Headers of client requests (continued)

- From = email address of the user
  - rarely sent in order to keep the user anonymous
- Host = specifies the machine and the port of the server
  - A server may host several other servers
- If-Modified-Since = withdrawal condition
  - The page is transferred only if it has been modified since the given date. Used by caches
- Max-Forwards = max number of proxys
- Proxy-Authorisation = identification
- Range = zone of the document to be returned
  - bytes=x-y (x=0 corresponds to the first byte, can be omitted to specify until the end)
- Referer = original URL
  - Page containing the anchor from which the URL was found.
- User-Agent = give information concerning the client, such as the name and the version of the browser, the operating system.

Example of a request

Example of a client request. Consider the URL: http://tuxa.sme.utc/sr03/td1.html
The message sent to the server is :
GET /sr03/td1.html HTTP/1.1 (1)
*Accept: image/gif, image/x-xbitmap, image/jpeg, */* (2)*
Accept-Language: fr (3)
Accept-Encoding: gzip, deflate (4)
User-Agent: Mozilla/4.0 (compatible; MSIE 5.01; Windows NT) (5)
Host: tuxa.utc.fr (6)
Connection: Keep-Alive (7)

***1-** The client asks the server (GET) for the document whose location is /sr03/td1.html . The client indicates its protocol version(HTTP1.1)*
***2-** The client indicates to the server which are the tyoppe of dcuments it accepts.*
***3-** The client indicates his or her favourite language*
***4-** The client indicates that it can process a response compressed with gzip and/or deflate*
***5-** The client identifies itself as Microsoft Internet Explorer 5.01, Window NT compatible Mozilla4.0*
***6-** The client provides the name of the server with respect to itself.*
***7-** The client asks the server to keep the connection open. This is a TCP connection, and not a application connection.*

Response headers

- Accept-Range = authorisation or refusal of a request by interval
- Age = age of the document in seconds
- Proxy-Authenticate = authentification system of the proxy
- Public = list of non-standard methods managed by the server
- Retry-After = date or number of seconds for a new try, in the case of a 503 code (service unavailable)
- Set-Cookie = create or modify a cookie with the client
- WWW-Authenticate = authentification system for the URI
- Allow = methods allowed for the URI
- Content-Base = base URI
- Last-Modified = date of the last modification of the document.
  - Used by caches

Response headers (continued)

- Content-Length = size of the document in bytes
  - Used by the client to follow the loading progression
- Content-Location : URI of the entity
  - When the URI is located at several places
- Content-Range : position of the partial body in the entity
  - bytes x-y/size
- Content-Transfert-Encoding : transformation applied to the body of the entity
  - 7bit, binary, base64, quoted-printable
- Content-Type = MIME type of the returned document
  - used by the client to select the plugin
- ETag : transformation applied to the body of the entity
  - 7bit, binary, base64, quoted-printable

Response codes

- The first line of response from a Web server is called the status line
- It begins with the HTTP protocol version and is followed by a for three-digit numerical code response
  - 100-199 Information
    - 100 : Continue (the client can send after the request), ...
  - 200-299 Success of the client request
    - 200: OK, 201: Created, 204 : No Content, ...
  - 300-399 Redirection of the client Request
    - 301: Redirection, 302: Found, 304: Not Modified, 305 : Use Proxy, ...
  - 400-499 Client request incomplete
    - 400: Bad Request, 401: Unauthorized, 403: Forbidden, 404: Not Found
  - 500-599 Server error
    - 500: Server Error, 501: Not Implemented, 502: Bad Gateway, 503: Out Of Resources (Service Unavailable)

An example of a response

```
HTTP/1.1 200 OK
Date: Mon, 24 Mar 2003 15:04:59 GMT
Server: Apache/1.3.26 (Unix) PHP/4.1.1
Last-Modified: Thu, 27 Sep 2001 14:48:40 GMT
ETag: "93d24-114-3bb33c48"
Accept-Ranges: bytes
Content-Length: 276
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html
Data (276 bytes)
```

HTTP : cookies

- Principle: Cookies are part of the HTTP specifications, which uses them to deal with the « connection loss » during session following.
  - A **cookie** is in fact a file which is stocked on the hard drive of the user;
  - They enable the web server to recognise the clients from one web page to another.
- The server sends a header :

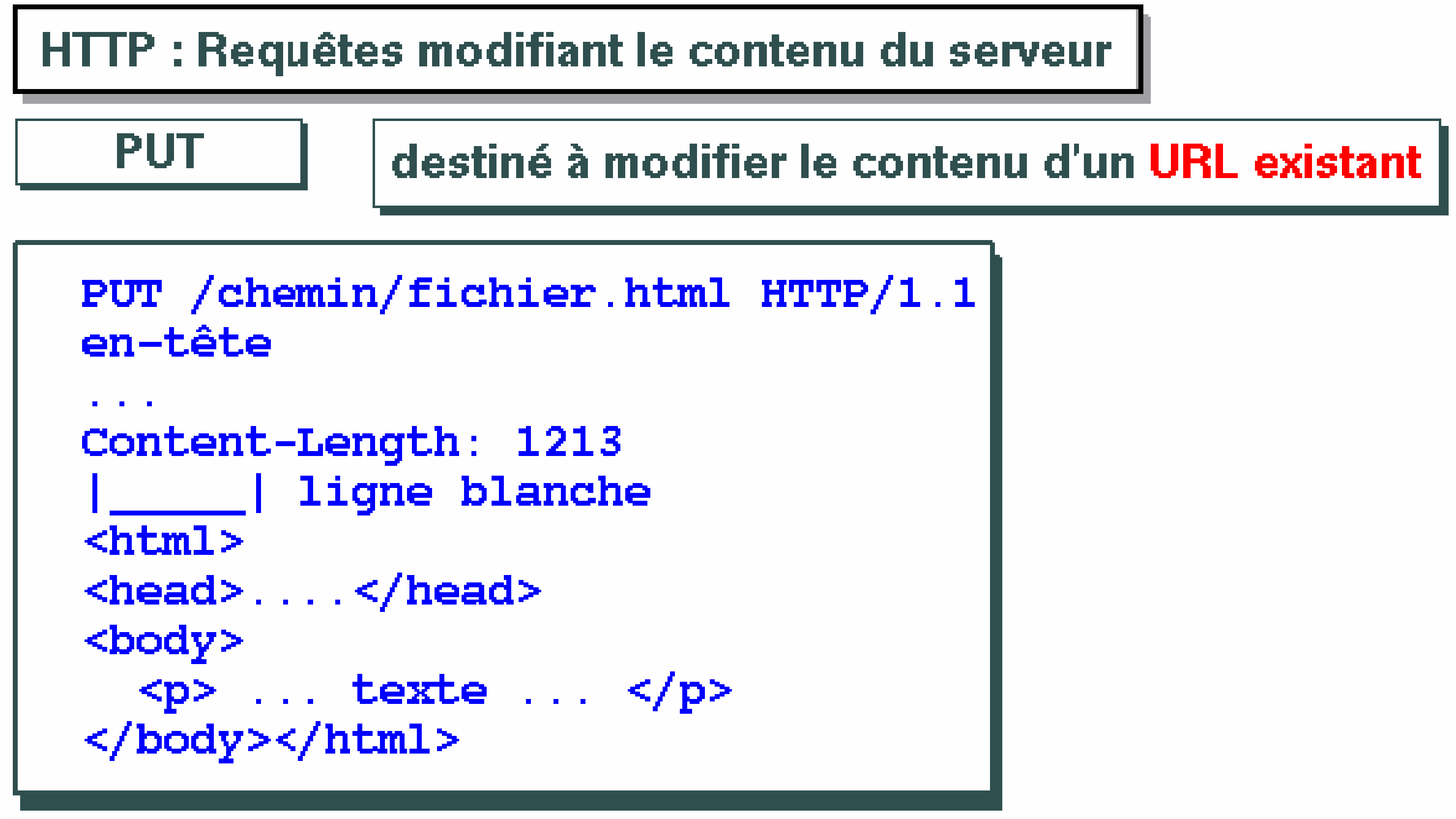
  Set-Cookie : NAME=VALUE; domain=NAME_OF_DOMAIN; expires=DATE;
- The navigator sends to the server :

  Cookie : NAME1=VALUE1; NAME2=VALUE2;

HTTP 1.1

- Persistant TCP connections
- New headers
- New requests
  - **PUT** : sends a document to the server, this document is to be saved at the specified URI.
    - Headers allow us to verify that all has gone well
  - **DELETE** : erases the specified resource.
  - **OPTIONS** : allows the client to see the usable communication options for obtaining the resource.
    - Know what can be done, without having to ask for a resource.
  - **TRACE** : control method. Asks the server to send the request back exactly as it was received.
  - …

HTTP 1.1 (continued)



HTTP : Requêtes modifiant le contenu du serveur

PUT

destiné à modifier le contenu d'un **URL existant**

```
PUT /chemin/fichier.html HTTP/1.1
en-tête
...
Content-Length: 1213
|_____| ligne blanche
<html>
<head>....</head>
<body>
  <p> ... texte ... </p>
</body></html>
```

HTTP 1.1 (continued)

HTTP : autres Requêtes du client

DELETE	**destiné à effacer un URL**
TRACE	**voir ce que devient une requête à travers un proxy**
OPTIONS	**Demander les options disponibles pour un URL donné. Serveur répond (par exemple) : Public: GET, HEAD, POST**

```
  DELETE /chemin/fichier.html HTTP/1.1
  en-tête
  ...
Serveur répond (si succès):
  HTTP/1.1 200 OK
  Date: ...
  .....
  <h1>URL deleted.</h1>
```

The essentials of HTTP

1. HTTP is located in the Application layer.
2. HTTP, light protocol, only carries text;
3. HTTP relies on the request/response system ;
4. HTTP is a protocol with no state;
5. HTTP enables bidirectionnel transfers;
6. Feature management ;
7. Cache and proxy server;
8. Session management using cookies.

HTML: basic principles

- The main Web documents are text files. Structure is added to this text using tags ("tags") such as <title>, <h2>,<head>, <p>, <ol>, <li>, etc ...

- HTML is a markup language from an earlier and more complex language ,SGML, defined by the community of librarians and library managers to fully describe all types of documents.

- After the definition and successful use of XML, HTML has evolved into XHTML, which incorporates and extends HTML to XML

HTML: basic principles

- Html is simple and « light »
  - Helloworld.doc = 23.5 KB
  - Helloworld.pdf = 7.02 KB
  - Helloworld.html = 88 bytes

HTML: basic principles

- HTML is based on the concept of tags.

- The different types of tags :
  - Tags containing meta-information
  - Web page formatting tags
  - Link tags
  - Multimedia insertion tags (images, sound, videos..)
  - ..

- Tags are not case sensitive (*preferably in lowercase*).

- How to write tags :
  - <name-tag> … </name-tag>

Tags containing meta-information

- Meta tags are used for :
  - Finding information about the language used by the web document, the type of document, the coding used, the author,...
  - Automatically redirect to another page,
  - Stop caching or indexation by research motors…

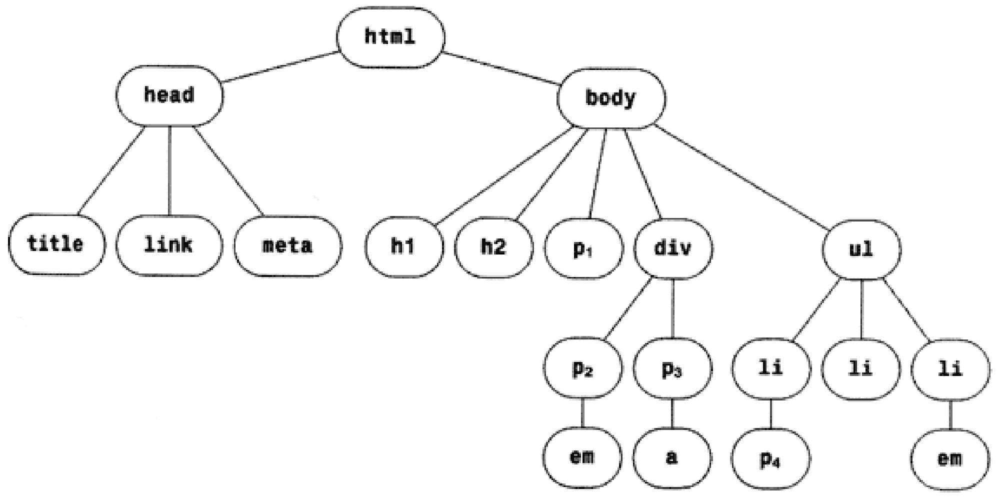
- Examples :

  The **NAME** metas, describe the HTML page
  - <META NAME="Author" CONTENT= "Michel Vayssade, Dritan Nace">
  - <META NAME="Copyright" CONTENT= "UTC">
  - <META NAME="Keywords" CONTENT= "SR03, Internet Architecture, Distributed applications">

  - **The <META HTTP-EQUIV tags, send additional information to the browser via the HTTP protocol**
    - <META http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
    - <META http-equiv="Content-Language"content= "en, fr">
    - <META http-equiv="Refresh" content="10; URL=http://www.hds.utc.fr">
    - <META http-equiv="Pragma" CONTENT="no-cache">
    - <META http-equiv="Content-Style-Type" CONTENT="text/css">

Web page formatting tags

– Examples of tags :
  - <Title>
  - <body>
  - <h1> h2, h3…h6.
  - <p>
  - <em>, <i> ..
  - <font>
  - <frame>
  - <hr>
  - <table> <col>…
  - <ol> <li>…
  - ….



Other tags

- An HTML document should begin with a document type declaration(<!DOCTYPE HTML PUBLIC "*type_of_HTML*" "*address_of_DTD*"> :
  - **Strict** : <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Strict//EN" "http://www.w3.org/TR/html4/strict.dtd">
  - Transitional : <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
  - <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

- Mutlimedia insertion tags (images, son, vidéos..)
  - <img>, <object>
- <! … --> Comments

- Tags used for implementing a stylesheet by whole blocs
  - <div>, <span>
- Including scripts, styles, PHP…
  - <script>, <style>, <?php … ?>

Link tags

<HTML>

<HEAD> <TITLE>title</TITLE> </HEAD>

<BODY>

<H3>Level 3 title </H3>

<p><b>First paragraph :</b> Example of HTTP links </p>

<br>

An "absolute" link :

    <A HREF="http://www.utc.fr/interne">Go to internal page of UTC</A>

A "relative" link :

    <A HREF="./test3.html">voir test3.html</A>

…

Internal link : It is possible to mark a specific location of a page and access it with a hyperlink with a NAME attribute or ID :

    <tag id "internalLink"> ... </tag> The call is as follows :
    <a href="#internalLink"> ... </a>

It is also possible to access a specific section of another page :

    <a href="url/name_of_file.html#internalLink"> ... </a>

HTML : first examples

- The tags give a **structure** to the HTML documents :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<HTML>
<HEAD>
<TITLE>Document title</TITLE>
</HEAD>
<BODY>
<P>Insert text or images here</P>
<P>A paragraph</p>
</BODY>
</HTML>..
```

HTML : first examples (continued)

The **tidy** program enables the verification of the correctness of an HTML document.

Another example :

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<HTML>
<HEAD>
<!-- comment, (in reality SGML ignored by the HTML) -->
<TITLE>Title displayed in the browser title bar</TITLE>
</HEAD>
<BODY>
<H1>This is a level 1 title (the biggest)</H1>
<p><b>First paragraph :</b> HTTP is based on text files.</p>
<p><i>Second paragraph :</i> The web : bla bla ...</p> …
```

Hyperlinks : labels in the document

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<HTML><HEAD><TITLE>
Document example with internal returns</TITLE></HEAD><BODY>
<A NAME="debut"></A><!– definition of an étiquette -->
<H3> Document example with internal returns : start.</H3>
<UL><LI><A HREF="#label1">Texte 1</A></LI>
<LI><A HREF="#label2">Text 2</A></LI></UL>
<HR>
<A NAME="label1"></A><H3>Text 1</H3>
<P>Texte 1 ......</p>
<A HREF="#start">start</A></P>
<HR>
<A NAME="label2"></A><H3>Text 3</H3>
<P>Text 3 ...... A label in another file can also be referenced :
<A HREF="test2.html#start">go to "start" in test2.html.</A></p>
<A HREF="#start">start</A></P>
</BODY></HTML>
```

Frames

- The *frames* technology enables display of several HTML pages in different zones (or *frames*).
  - Do not forget to insert <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">

- Example:

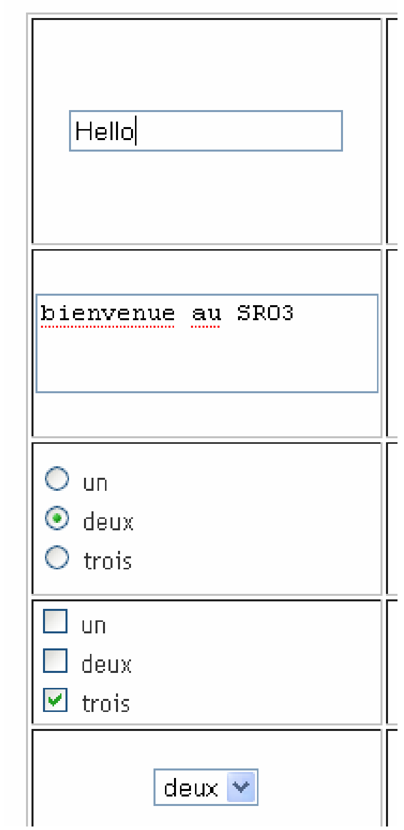
```
<FRAMESET COLS=« 50%, 50%">
<FRAME SRC="frame1.htm" NAME="left">
<FRAMESET ROWS="50%, 50%">
<FRAME SRC="frame2.htm" NAME="upper_right">
<FRAME SRC="frame3.htm" NAME="lower_right">
</FRAMESET>
```

Forms

Form tags :

opening <FORM> and closing </FORM> tags:

The specification of the method used for sending information to the server + the information destination address (<FORM METHOD=GET ACTION=cgi-bin/test8.cgi>.. </FORM>)



The tags:

- The tag <input>
- The tag <select>
- The tag <option>
- The tag <textarea>

The fields:

- Text fields
- Text zone fields
- Radio buttons
- Boxes to tick
- Scrolling list
- Open list

Forms

```
<HTML><HEAD><TITLE>
Page example with a shape</TITLE></HEAD><BODY>
<H3>Sample page with a text entering field.</H3>
<FORM METHOD=GET ACTION=cgi-bin/test8.cgi> .
<P>Enter your Name and address :<BR>
<P>Name : <INPUT NAME=name SIZE=40>
<P>Address :<INPUT NAME=address SIZE=60>
<P><INPUT TYPE=submit VALUE=Send> .
<INPUT TYPE=reset VALUE=Erase>
</FORM></BODY></HTML>
```

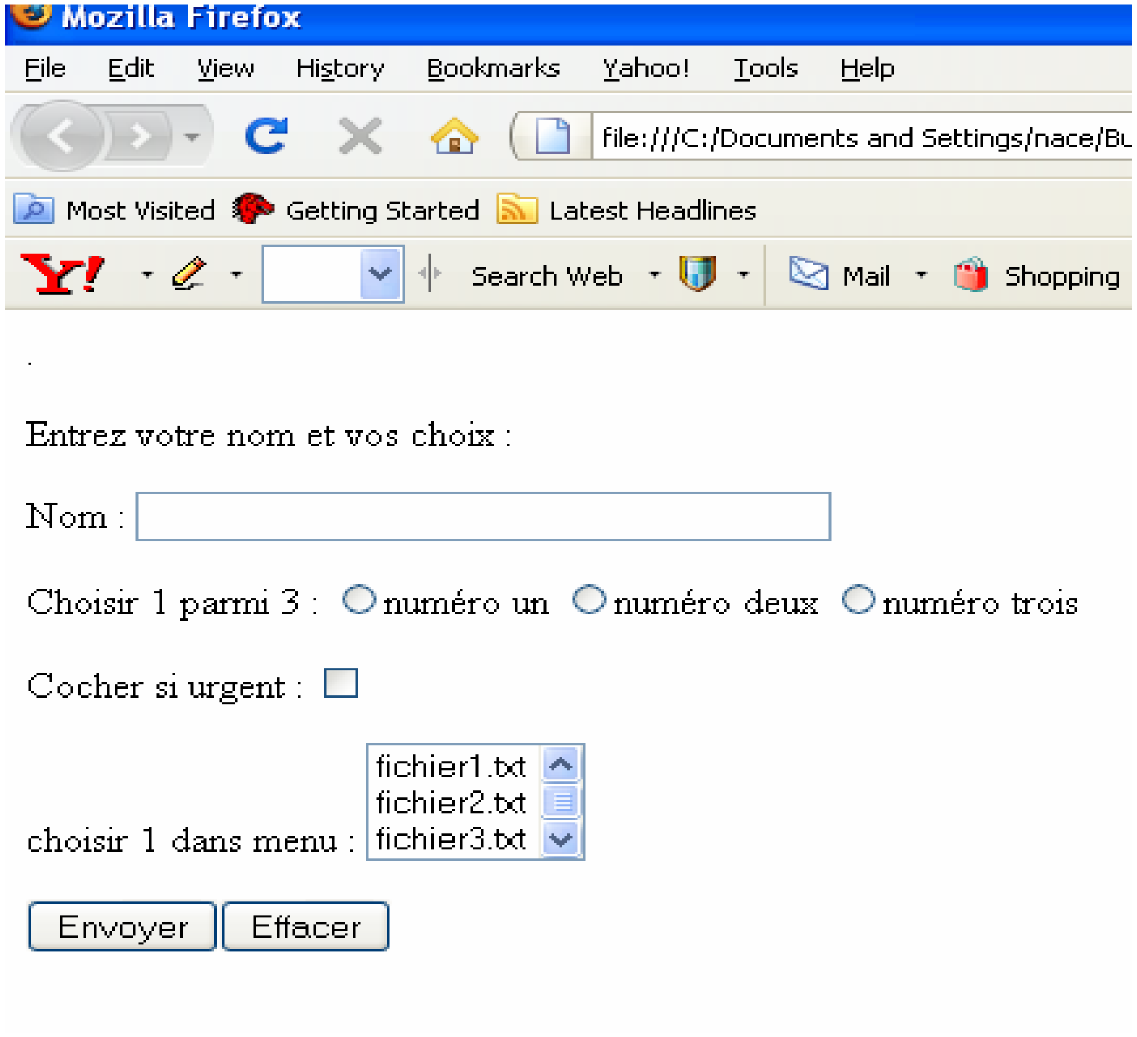
ACTION=cgi-bin/test8.cgi : Web servers are often configured to accept the "cgi" only in specific directories that can be protected (drwx-x-x). The ". Cgi" files can be written in any language (PHP, Perl, Python, C, C + +, bash, ...).

Forms

```
<FORM METHOD=GET ACTION=cgi-bin/test9.cgi> .
<P>Enter your name and your choices :<BR>
<P>Name : <INPUT TYPE=TEXT NAME=name SIZE=40>
<P>Choose 1 out of 3 :
<INPUT TYPE=radio NAME=choice1 value="one">number one
<INPUT TYPE=radio NAME=choice1 value="two">number two
<INPUT TYPE=radio NAME=choice1 value="three">number three
<P>Tick if urgent : <INPUT TYPE=checkbox Name=urg>
<p>choose 1 in menu :
<SELECT NAME=file SIZE=3>
<OPTION>file1.txt <OPTION>file2.txt
<OPTION>file3.txt <OPTION>file4.txt <OPTION>file5.txt
</SELECT>
<P><INPUT TYPE=submit VALUE=Send><INPUT TYPE=reset VALUE=Erase>
</FORM>
```

Rem : <SELECT .. SIZE=3> => menu with « elevator" ;
Si SIZE=1 => "scrolling" menu.

Forms


Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
file:///C:/Documents and Settings/nace/Bu
Most Visited Getting Started Latest Headlines
Y! Search Web Mail Shopping

Entrez votre nom et vos choix :

Nom :

Choisir 1 parmi 3 : ○ numéro un ○ numéro deux ○ numéro trois

Cocher si urgent : ☐

choisir 1 dans menu : fichier1.txt fichier2.txt fichier3.txt

Envoyer Effacer

Events associated with a form

- Events associated with the tag <form> :
  - onSubmit
  - onReset

- Events associated with fields :
  - onChange
  - onBlur
  - onFocus
  - onSelect
  - ..

Colours

Colour code specified by: color="#RRGGBB".

RR, GG or BB = 00 at FF (from 0 to 256 values of red, green, blue).

Example : "#FFFF00" = 100% of red + 100% of green + 0 = yellow.

See **http ://www.w3.org/MarkUp/Guide/Style**.

HTML

- HTML describes the structure of the document, and (unfortunately) as information on how to print certain items. But the means of printing depends on the configuration of the browser upon which the author of the document has no control.

- **This mixing of types** between structure et "appearance" is bad and is corrected by using **CSS and XHTML**.

CSS stylesheets
(Cascading Style Sheets)

Objective : simplify the presentation/design of web pages

Advantages of CSS :

Structure and présentation are managed separately

Regular presentation,

Rigorous positioning of elements,

The HTML code is **streamlined** and the lisibility is improved.

CSS

It is useful to tell the browser which language is used for the stylesheet definition.

```
<META HTTP-EQUIV="Content-Style-Type" CONTENT="text/css">
```

- Tag <style> </ style>
  - Attributes : type, media, title;
- Watch out for browsers which do not support stylesheets :

```
<STYLE type="text/css">
<!--
…..
-->
</STYLE>
```

CSS : properties

CSS - properties:

- Police formatting properties
- Text formatting properties
- Text colour and page background properties
- Paragraph formatting properties
- Border properties in the BORDER box
- External margin properties MARGIN
- Internal margin properties PADDING
- LIST-STYLE properties

The details can be found at :

http://www.w3.org/TR/CSS2/propidx.html

Stylesheets

Here is an example of a CSS rule which diplays the principle headers (H1) in blue :

```
H1 { color : blue }
```

A CSS rule is composed of two parts :
a selector (here H1) and a declaration (color:blue).

A declaration has itself two parts :
a property (color) and a value (blue).

```
H1, P { color : red }
P { margin-left : 1cm ; text-style : italic }
```

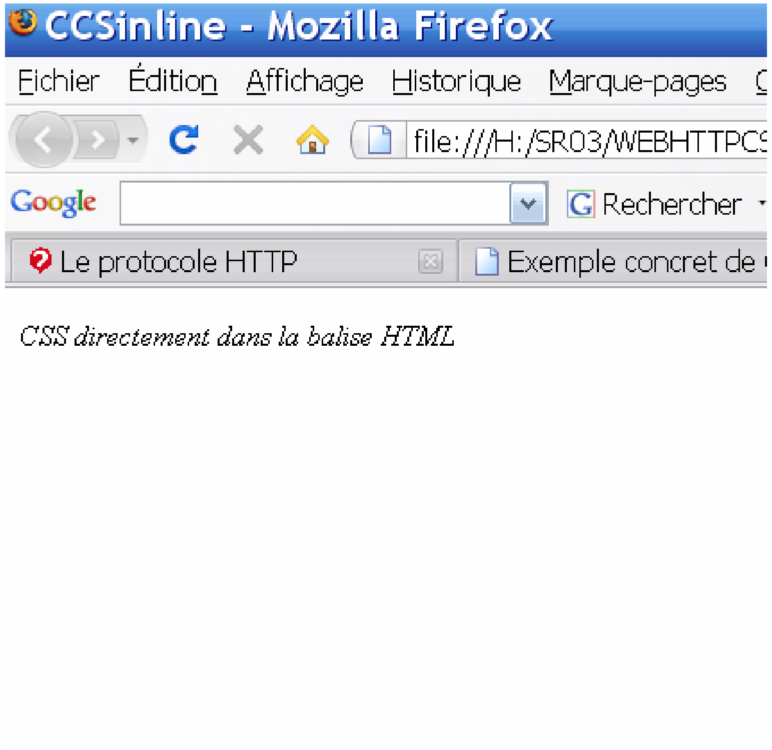
Including a stylesheet

- The CSS code can be included in three different places in the page.
  - In the HTML element itself
  - In the HTML page
  - In an independant file

Putting a stylesheet directly in the HTML tag

```
<HTML>
  <HEAD>
    <TITLE>
      CCSinline
    </TITLE>
  </HEAD>
  <BODY>
<p style="font-style: italic; size: 1.5em;">
CSS directly in the HTML tag</p>
  </BODY>
</HTML>
```

Rem: This method is best avoided

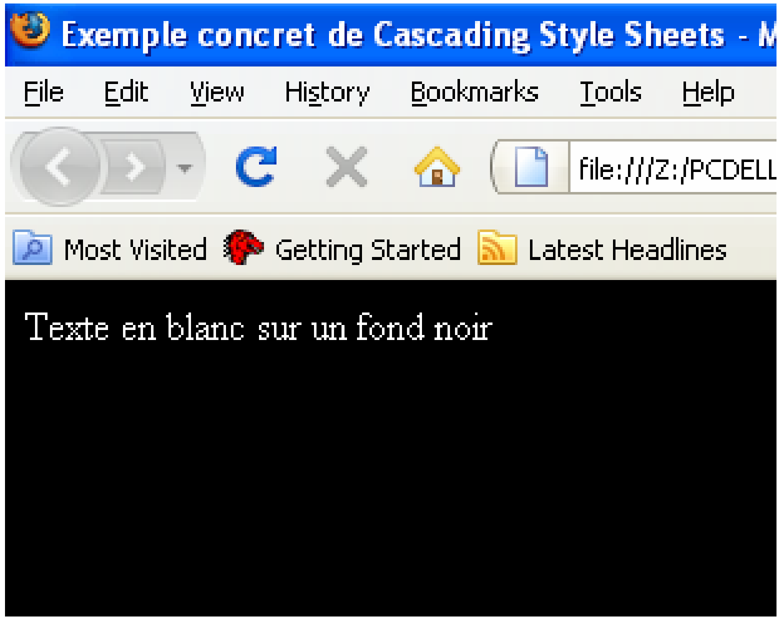


CSS directement dans la balise HTML

Putting a stylesheet in the header of a HTML page

- In the code of the HTML page, between the two tags <head> and </head>.

```
<HTML>
  <HEAD>
    <TITLE>Concrete example of Cascading Style Sheets</TITLE>
    <STYLE TYPE="text/css">
      <!--
        BODY {
          color: white;
          background: black;
        }
      //-->
    </STYLE>
  </HEAD>
  <BODY>
    White text on a black background
  </BODY>
</HTML>
```



Texte en blanc sur un fond noir

Putting a stylesheet in a separate file

- Syntax: <link rel="stylesheet" type="text/css" href="style.css" />
- The method "<link href =..." can also set up style sheets for different media (printer, PDA browsers, etc..).
  - screen, projection, handheld, print, all.

- How it works: place the style sheet in a separate file, and refer to them in the document header.

```
…<head>
<title><title>
<link rel="stylesheet" type="text/css" href="style.css“media="screen,
    projection" />
</head>
<body> …
```

```
Fichier CSS (style.css) :
H1, H2,{ color : red; font : bold}   /* Main titles are in red   */
H3, H4, H5 { color : green} /* Secondary titles are in green*/
```

Importing a stylesheet

The @import rule : @import is a CSS2 property which has to be followed by the URL of a file which contains the styles to apply additionally to the current styles.

Syntax:

```
<style type="text/css">
@import url(styles.css) media;
</style>
```

Remarks :

1. Certain old versions of Internet Explorer and Netscape (versions 4 or before) do not recognise "@import".
2. Can be useful for importing stylesheets in other stylesheets.

Element classes

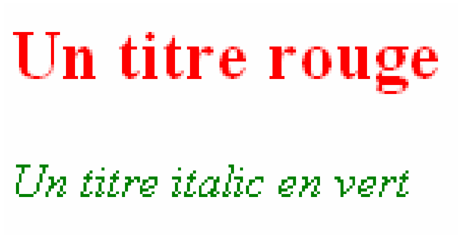
- Allow the choice between several presentation types for the same HTML element.
  - Example : several header paragraphs, comments and normal paragraphs are possible.

- The class attribute, which can be applied to all HTML elements, enables the definition of element classes. The styles which will be applied to each class can therefore be defined.

  - .red { color: red } P.header { font-style: italic }

An example

```
<HTML>
  <HEAD>
    <TITLE> Les classes </TITLE>
    <STYLE type="text/css">
      <!--
.rouge { color: red }
P.entete { color: green; font-style: italic }
      //-->
    </STYLE>
  </HEAD>
  <BODY>
<H2 class=rouge> Un titre rouge </H2>
<P class=entete> Un titre italic en vert </P>
 </BODY>
</HTML>
```


Un titre rouge

Un titre italic en vert

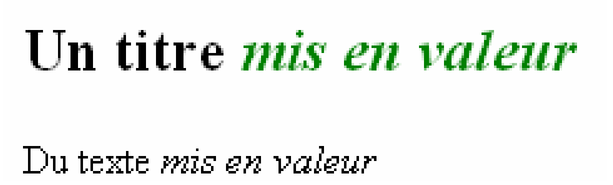
Contexts

- It is possible to specify properties that must be applied to an element when it is used within another element (and not elsewhere).
  - H2 EM {color : green}

- In the HTML code, the EM element can be used in an H2 header or not :
  - <H2>A title <EM>highlighted</EM></H2>
    <P>Text <EM>highlighted</EM>

An example

```
<HTML>
  <HEAD>
    <TITLE> Contexts </TITLE>
    <STYLE type="text/css">
      <!--
        H2 EM {color : green}
      //-->
    </STYLE>
  </HEAD>
  <BODY>
<H2>A title<EM>highlighted</EM></H2>
<P>Text <EM>highlighted</EM>
    </P>
 </BODY>
</HTML>
```



Un titre *mis en valeur*

Du texte *mis en valeur*

Pseudo-classes

- **Anchor pseudo-classes :link et :visited**
  - CSS is used to represent in a different manner links which have not been visited by those who have already been through pseudo-classes ': link' and ': visited'.
    - Ex. A:link { color : fushia; } A:visited { color : olive; } A:active { color : green; }
    - The pseudo-class :link applies to links which have not been visited ;
    - The pseudo-class :visited applies when the link have been visited by a user.

- Pseudo-classes can be combined with the following classes : A.aclass:link { color : #05f7a2 }

- Can be used with different contexts :   A:link IMG { border : solid yellow; }

Pseudo-classes

example

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML
4.01 Transitional//EN">
<html>
  <head>
    <link rel="stylesheet" href="P.css"
type="text/css">
    <title>CSS Pseudo Classes</title>
  </head>
  <body>
        <td>
          <a href="http://www.utc.fr">site UTC</a>
        </td>
</body>
</html>
```

```
a:link
{
background-color: YellowGreen;
  color: white;
  font-family: Arial;
  font-size: .8em;
}
a:visited
{
  background-color: red;
  color: white;
  font-family: Arial;
  font-size: .8em;
}
a:hover
{
  background-color: cyan;
  color: blue;
  font-family: Arial;
  font-size: .8em;
}
```

Pseudo-elements

- Pseudo-elements create abstractions in the tree in addition to items already specified by the language of the document.
  - Therefore, some languages do not provide mechanisms for correspondence with the first letter or first line of the contents of an element. The CSS pseudo-elements allow authors to access them.

Pseudo-elements

- P:first-letter { font-size: 200%;}
- P:first-line { font-variant:small-caps; }

```
<HTML>
  <HEAD>
    <TITLE>Lettrine and first line</TITLE>
    <STYLE type="text/css">
      <!--
        P               { color: red; font-size: 12pt }
        P:first-letter { color: green; font-size: 200%; font-weight: 900 }
        P:first-line   { color: blue }
      //-->
    </STYLE>
  </HEAD>
  <BODY>
    <P>
      Here is the paragraph whose first letter is green and big,
      whose first line text is blue and other lines are red.
    </P>
 </BODY>
</HTML>
```

Identifiers and DIV/SPAN tags

Identifiers are an alternative from classes :

- **Identifiant**
  - #header {-------} : this style will be applied whatever the tag of the page which has the identifier ;
  - div#header {-------} : this style will only be applied to the tag <div id="header">

The DIV/SPAN tags

- The <div> and </div> are used in order to apply a style to content.
- The use of <span> </ span> tags can also apply a CSS style to bold or italic type, whereas the div allow page formatting

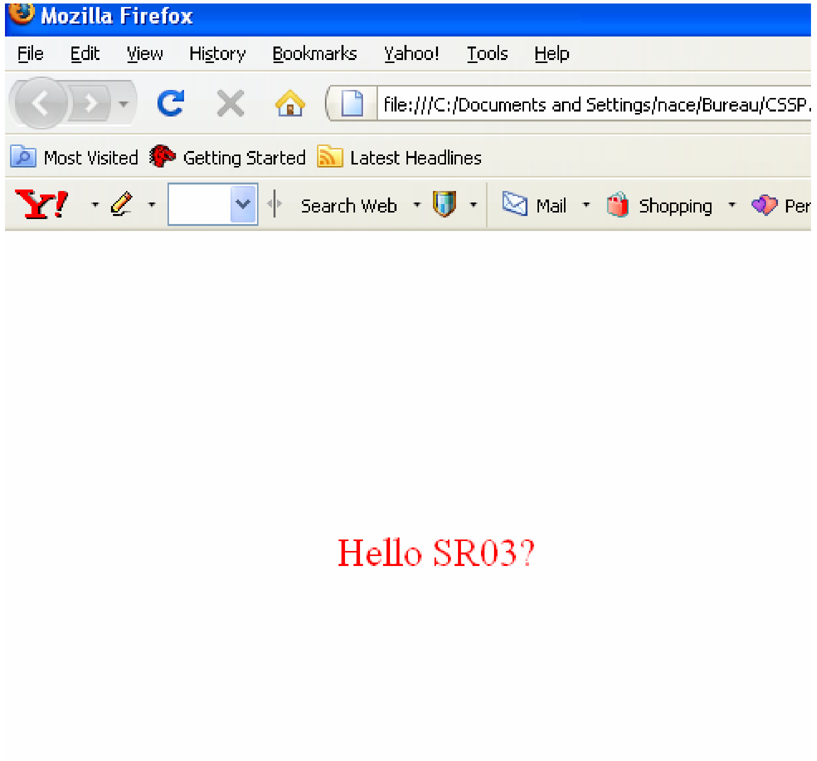
Position objects with CSS

- It is possible to position text or images with a pixel level precision, with stylesheets and using the tags <DIV> and <SPAN>.

- **Relative and absolute positioning**
  - Absolute positioning (position: absolute) is determined from the upper left corner of the browser window. The coordinates of a point are then expressed from top to bottom (top) and left to right (left).
  - The relative positioning is relative to other elements, that is to say that the elements contained in the DIV or SPAN tags will be set following HTML elements after which they follow.

Position objects with CSS

```
<HTML>
<HEAD>
<STYLE>
<!--
.pos{position: absolute; top: 180px; left: 200px; color: red; font-size: x-large}
-->
</STYLE>
</HEAD>
<BODY>
<DIV class=pos>
Hello SR03?
</DIV>
</BODY>
</HTML>
```


Mozilla Firefox
File Edit View History Bookmarks Yahoo! Tools Help
file:///C:/Documents and Settings/nace/Bureau/CSSP.
Most Visited Getting Started Latest Headlines
Search Web Mail Shopping Per

Hello SR03?

Cascade

- The CSS properties can be defined several times. It is always the last definition that counts. This way, multiple stylesheets can be imported, and certain styles can be redefined in the document.

```
<body>
        <div>
                <p>
                        bla bla bla
                </p>
        </div>
</body>
```

- Let us suppose that we have a first stylesheet, which we shall call style1.css and contains the following properties :
  – H1 { color : red; font-size : 48pt }
    H2 { color : blue; font-size : 12pt }

- Another stylesheet is also available, called style2.css, and contains the following properties :
  – H2 {color : green; }
    H3 {color : pink; font-size : 12pt }

Cascade

- In a given page, the call to these two pages is included, as well as the definition of other properties.

```
<HEAD>
<TITLE>...</TITLE>
<LINK rel=STYLESHEET href="style1.css" type="text/css">
<LINK rel=STYLESHEET href="style2.css" type="text/css">
<STYLE type="text/css">
<!–
H1 { color : fushia; } H2 { font-size : 16pt; } H3 { font-size : 14pt; } -->
</STYLE>
</HEAD>
```

- Determine the values used for the document :
  – H1 : fushia, 48 points
  – H2 : vert, 16 points
  – H3 : rose, 14 points

Cascading/Inheritancee

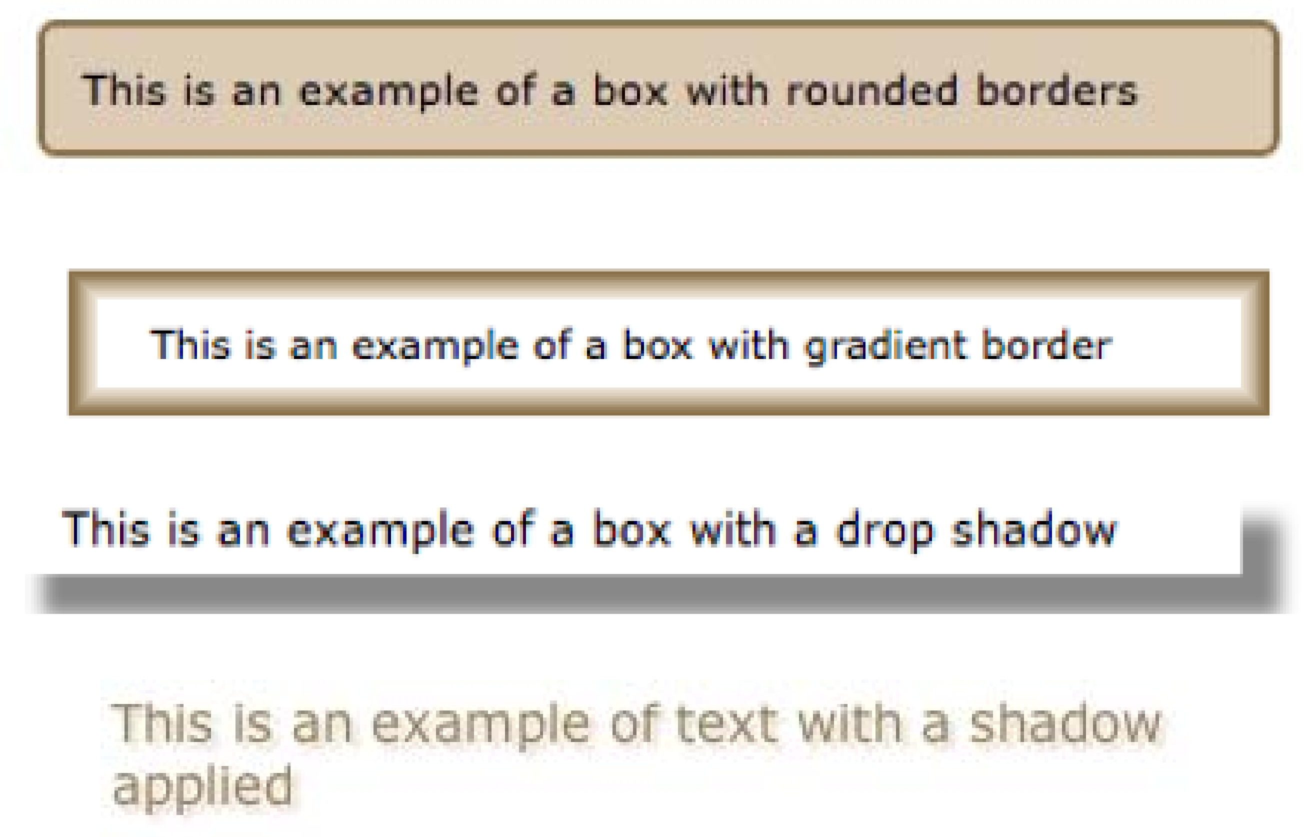
- In order to determine the value of a property, there is the concept of cascading. In case the property has not been defined, two possibilities are available :
  - In the first case, the property is "inherited". In this case, the value of the "parent" element is used, in other words the element in which the current element is used.
  - In the other case, the default value is used.

REMEMBER: the last rule which was read has the priority!!

CSS3

- CSS3 is being developed.
- It offers new features to improve web site design, in particular :
  - Backgrounds and borders
  - Text effects
  - Multicolumn page formatting…

CSS3



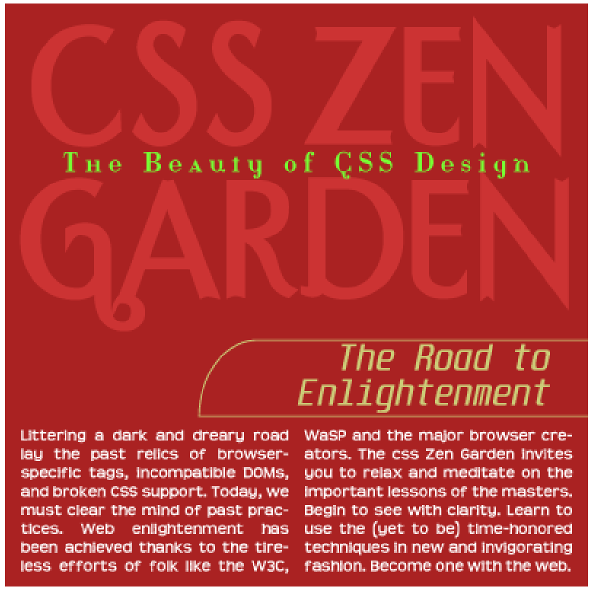
This is an example of a box with rounded borders

This is an example of a box with gradient border

This is an example of a box with a drop shadow

This is an example of text with a shadow applied

Web design via CSS

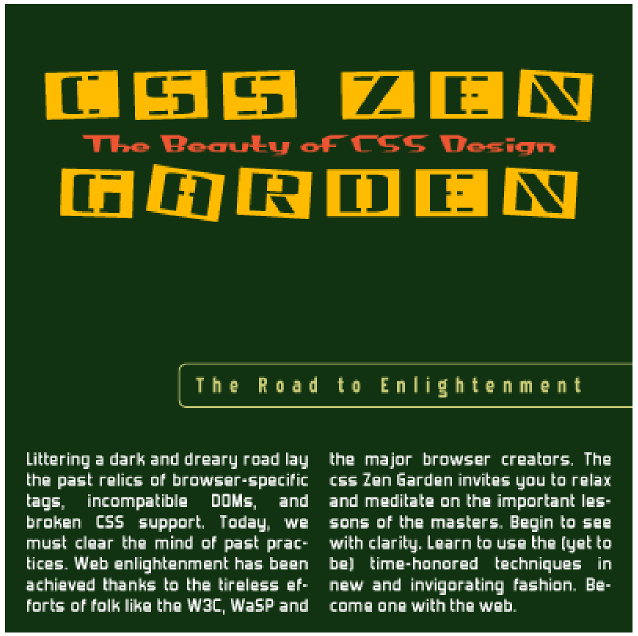


CSS ZEN GARDEN

The Beauty of CSS Design

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support. Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators. The css Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.



CSS ZEN

The Beauty of CSS Design

GARDEN

The Road to Enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support. Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators. The css Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

http://www.alistapart.com/articles/cssatten

DHTML

- ***Dynamic HTML***, or **DHTML**, is a generic name given to all the techniques used by the author of a web page so that the page will be able to modify itself while being viewed in a web browser.

- How does it work :
  - The internal representation is initially determined by the HTML document and the CSS style information which make up the web page.
  - The modifications are carried out using JavaScript, which accesses the internal representation through the Document Object Model (DOM) programming interface.
    - Careful :the CSS proerties of the DOM do not necessarily correspond to the « CSS » names.

Bibliography

– M. Vayssade "SR03 : ARCHITECTURES INTERNET", 2007;
– S. Cateloin, "Cours *HTTP*", l'Université Louis Pasteur de Strasbourg;
– "HTML et javascript", S. Maccari et S. Martin, ed. MicroApplication, 2004.
– Douglas Comer, " TCP/IP Architectures, protocoles et applications", Pearson Education, 5eme édition, 2009.
– A. Ploix, Cours "*Réseaux IP : Internet Réseaux d’entreprise*", Université de Technologie de Troyes.
– Internet souces :

http://www.developpez.com

http://fr.wikipedia.org

http://www.commentcamarche.net/
http://www.alistapart.com/articles/cssatten