Linguaggi per il web

panorama completo

- Introduzione alla comunicazione via web
  - Client browser / web server, protocollo di trasferimento
- Presentazione di risorse sul web
  - XHTML →→→ HTML (?)
  - uso di Cascading Style Sheet (CSS)
- Linguaggi di scripting
  - PHP, con MySQL
- Strutturazione e rappresentazione di informazioni
  - XML
  - Analisi e manipolazione di documenti XML (DOM, SAX)
- qualcosa di piu` …

Scopo: essere in condizioni di costruire una applicazione web non banale durante la parte finale del corso

Altro? XSLT,, HTML5, Javascript, Ajax, Jquery, Json, testing, Github … se possibile

pagina web del corso
http://www.diag.uniroma1.it/~marte (area didattica, link al corso)

Linguaggi per il web: organizzazione

LUN 15:45-18:45 AULA 6, VEN 10-13 LAB

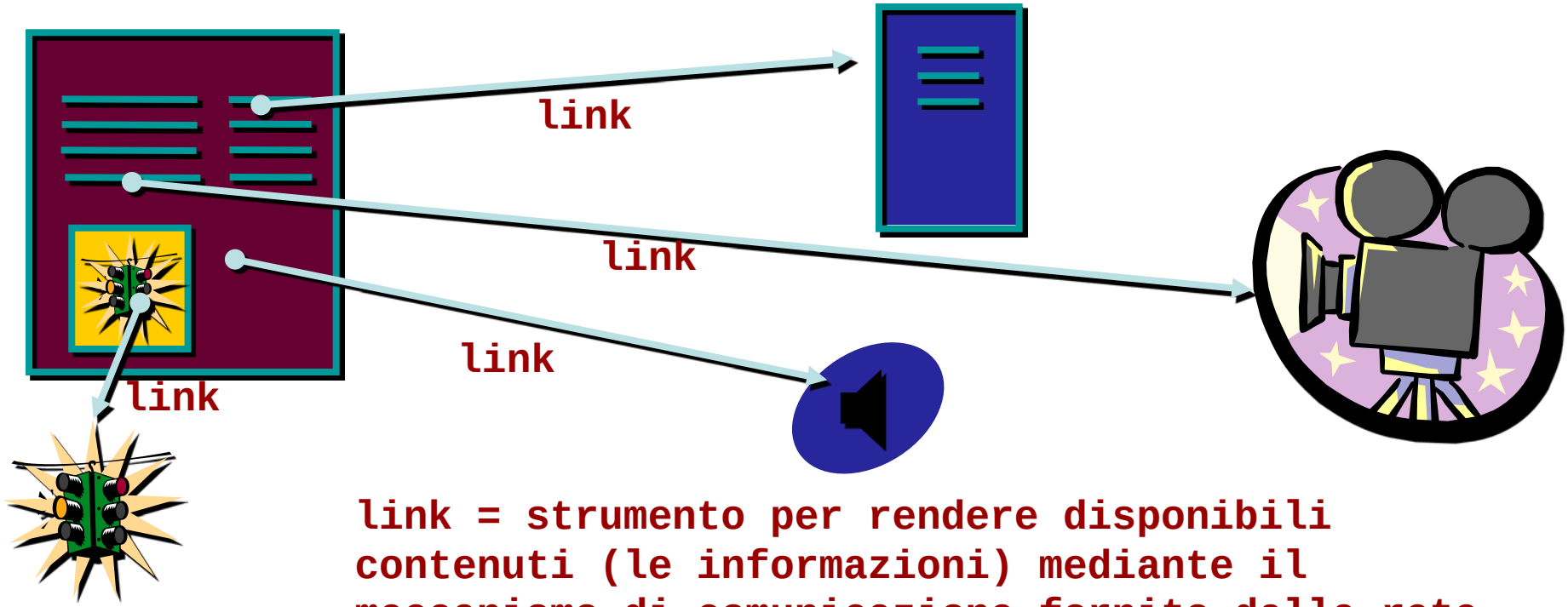
(laboratorio informatico 😭 aula 6 ...)

- Laboratorio → aula 6, casa …
  - Uso (visione e *modifica*) degli esempi visti a lezione
  - esercizi suggeriti in fondo alle lezioni (anche per I “compiti”)
  - Uso di un account personale sul server lweb per pubblicare i propri esercizi e la tesina

- Esame (vedi sito web del corso)
  - Compiti (sottomessi attraverso classroom)
    - collaborativi … 1,2, non necessariamente sempre le stesse persone
  - Tesina (sito web dinamico)
    - 1 o 2 persone
  - Scritto (domande) e poi orale … e poi basta

Tutto comincio` ...

- "testo a + dimensioni" Ipertesto/Ipermedia (testo + link a risorse) …
- = che contiene informazioni testuali e legami/indirizzamenti con risorse di diverso "genere mediale" e con altri ipertesti
- (what's *risorse*?: testi, immagini, fisse o scorrenti, suoni, programmi)



link = strumento per rendere disponibili contenuti (le informazioni) mediante il meccanismo di comunicazione fornito dalla rete.

Disponibili da parte di un "somministratore"

Disponibili ad un "ricevitore"

Un passo avanti: *The Big Picture (1/3)*

Il web ha a che fare con un'area dell'ICT …

ed evolve

- Internet (non e` il web)

  ↓

- Web (collegamenti tra **risorse**)

  ↓

- Semantic Web (esprimere, capire, gestire il significato delle risorse per permettere la ricerca e l'inferenza di informazioni)
  - Relazioni tra risorse
  - Aggiunta di informazioni
  - Inferenza di informazioni

  } significato (Semantic)

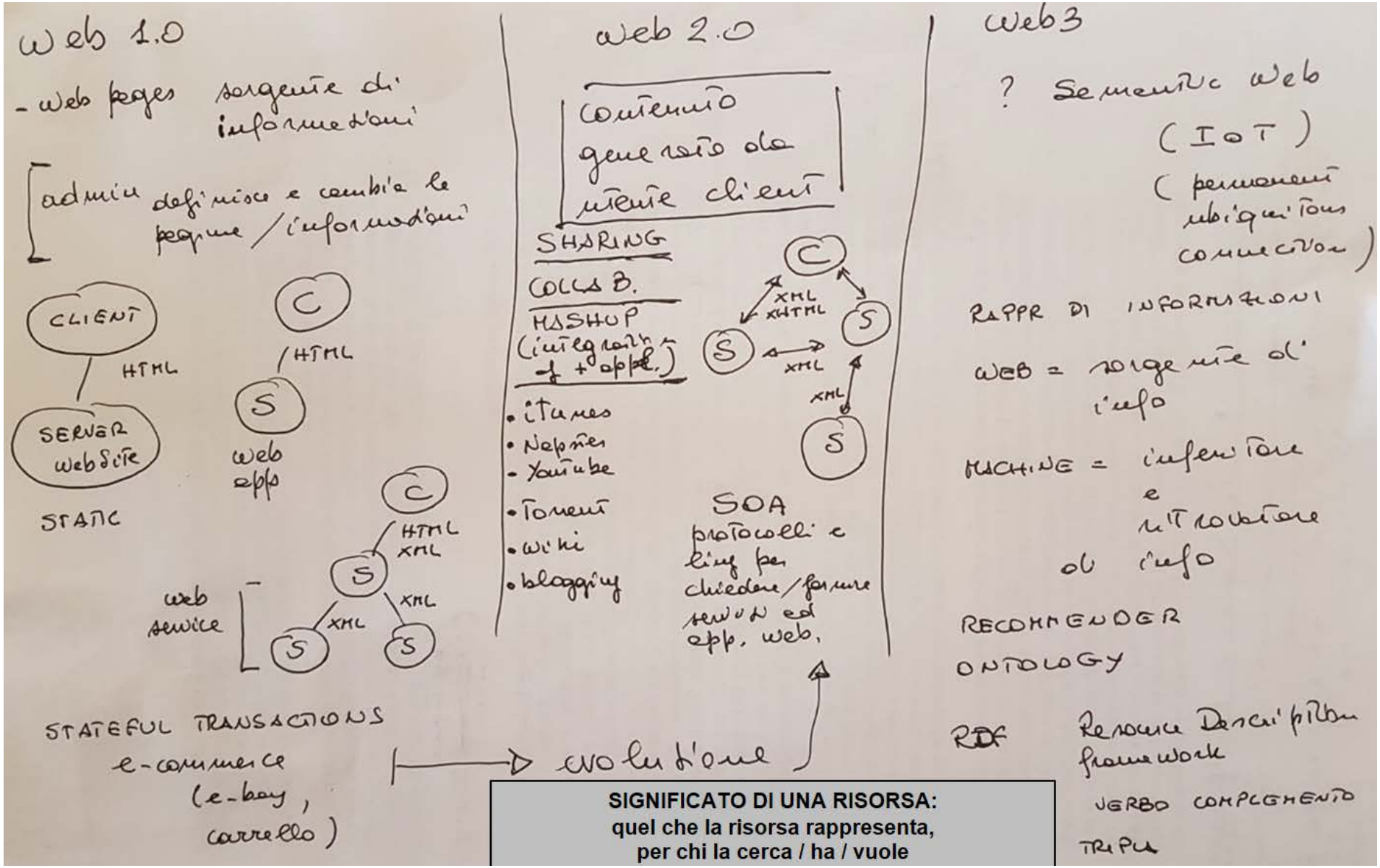
"qualunque cosa che abbia una identificazione (*identita`* … `indirizzo`)

UNAMBIGUOUS

… web come collection of ~~data~~ knowledge, resa in una forma strutturata, leggibile per hu-man and machine

per permettere di cercare la risposta a domande piu` complesse che "dov'e` questa collezione di parole?", trovare relazioni; confidare nelle risposte (TRUST); … per trovare, condividere e combinare informazioni ***piu` semplicemente***

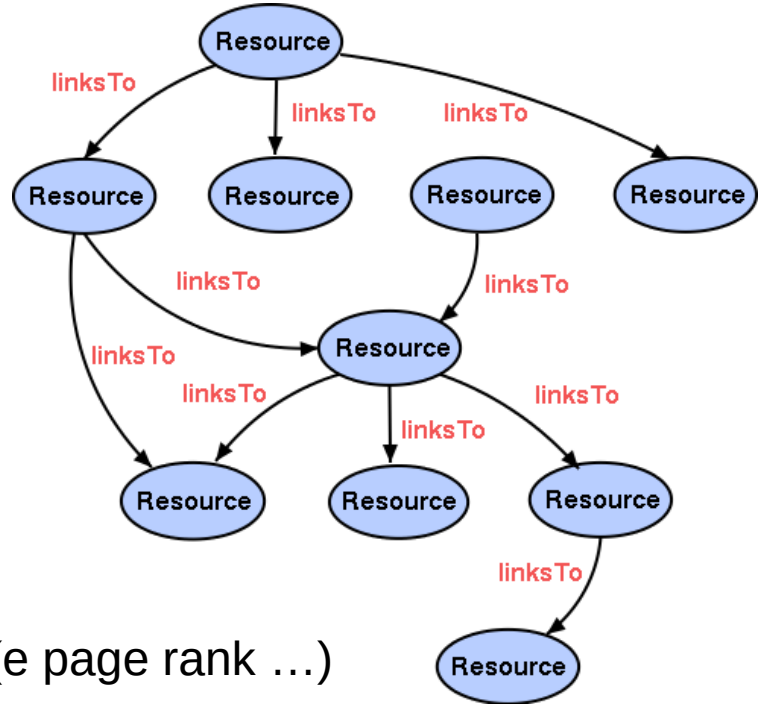
Un passo avanti: *The Big Picture (1/3 bis)*



Web Server, protocollo http per le transazioni, Service oriented Architecture, Resource Description Framework

Un passo avanti: *The Big Picture (2/3)*

What's Piu` semplicemente?



Rappresentazione dell'informazione sul web?

- page layout, natural language, graphics, multimedia
  - Sono convogliatori di significato
  - Humans OK - Machine ?!

Ricerca e combinazione informazioni?

- apparentemente solo in base a word matching (e page rank …)

"che influsso ha la temperatura atmosferica sulla borsa valori?"

- Comparare tabelle di temperature e di stock exchange; usare anche altre tabelle reperibili sul web
- Humans ?? (provaci) - Machine Si`, se l'informazione sul web fosse strutturata in modo piu` formale e gestibile da un automa …

"che influsso ha la classifica della serie A sulla salute di alcune classi di persone? "E sulle elezioni?"

Un passo avanti: *The Big Picture (3/3)*

c'e` differenza tra il puro linking tra risorse e la lore connessione semantica ...

IF (

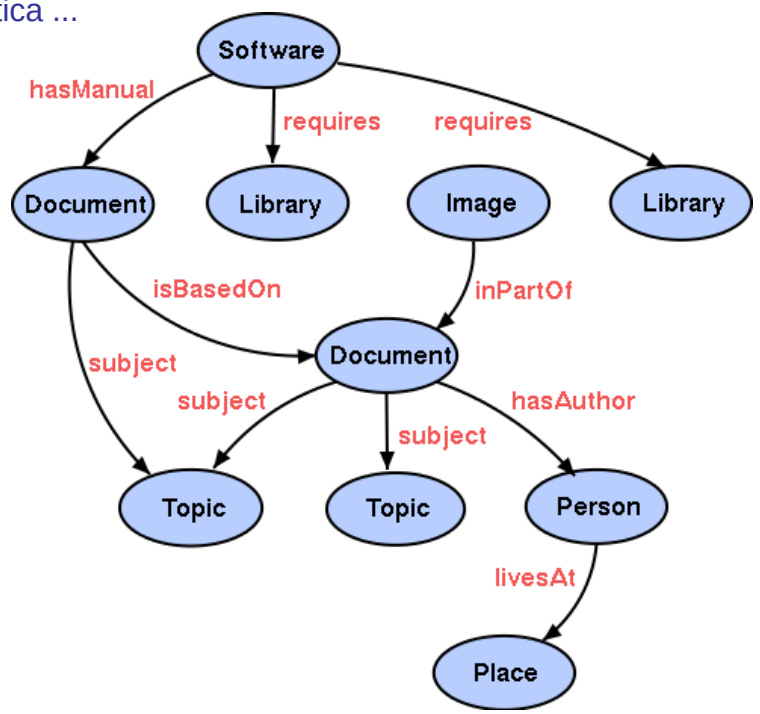
si puo` gestire una

rappresentazione formale

delle risorse e

delle loro relazioni

)



THEN ("Piu` semplicemente")

e` possibile la processazione automatica

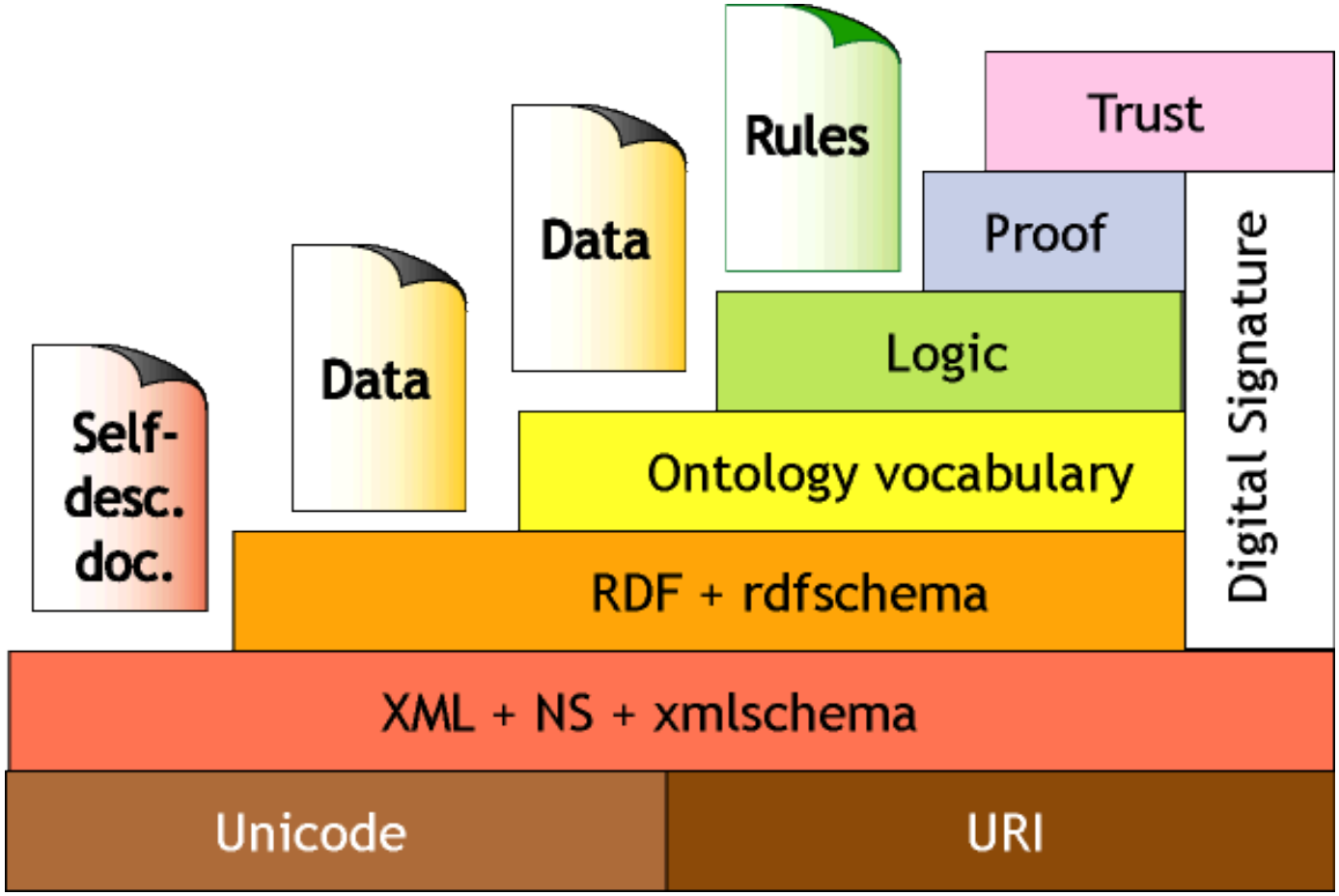
cioe` trovare informazioni su risorse e relazioni tra risorse, e inferire informazioni e relazioni in modo intelligente (secondo le regole della rappresentazione *e quelle dell'inferenza* …) e AUTOMATICO

Esempio di tripla RDF: Jane sells books Altro esempio: Mary is a mother Queste triple rappresentano "fatti".

Esempio di regola di inferenza: a mother is a parent

Esempio di inferenza: Mary is a mother (fact) + a mother is a parent (rule) ... Mary is a parent (fact)

Un passo avanti: *The Big Picture (Semantic Web)*



una introduzione veloce e` qui
https://www.youtube.com/watch?v=V6BR9DrmUQA
una un po' piu` lunga ma bella
https://www.youtube.com/watch?v=rhgUDGtT2EM

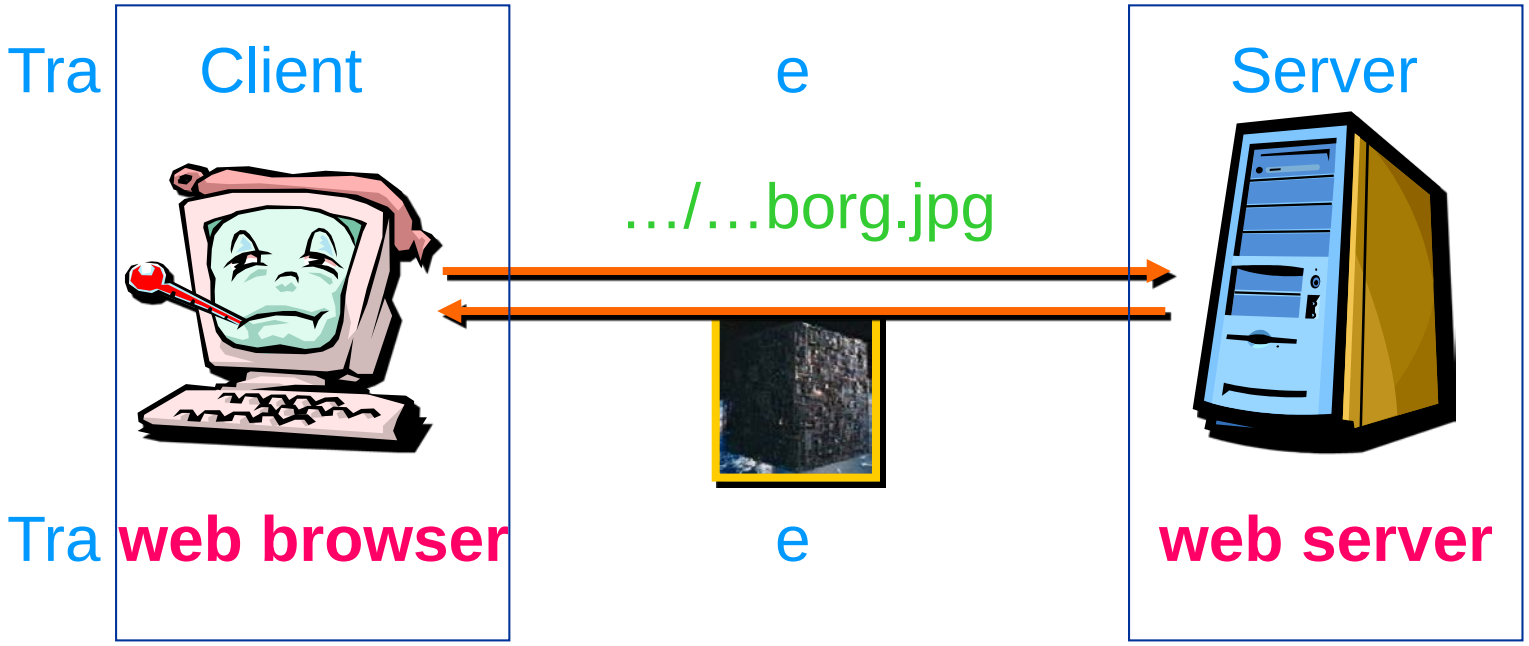
Non faremo tutto il viaggio … solo I primi due gradini

- Internet, parte gia` fatto e parte da fare
- Web, lo facciamo qui
- Semantic web … un giorno ... in futuro ...

e` chatGPT?

Comunicazione via

web = interconnesioni tra le macchine che assumono di volta in volta il ruolo di "somministratore" e "ricevente"



Si basa su

- Un sistema uniforme per il **nome risorsa**
- **un protocollo** per i **messaggi** scambiati
- **programmi** ai capi di collegamento

Protocollo

Regole su cui ci si basa per stabilire il formato e gestire il contenuto dei messaggi che vengono scambiati.

telnet, ftp, SMTP (Simple Mail Transfer Protocol), NNTP (Network News T.P.), **file,**
HTTP (HyperText Transfer Protocol,
ver. 1.0. 1.1, 2)

Nome della risorsa (1/2) - URI

Quando viene richiesta una certa risorsa, tramite un certo protocollo, l'identificazione della risorsa viene fornita secondo il formato (generico)

Protocollo://hostname/path-risorsa

```
http://lweb.dis.uniroma1.it/fun/fig/borg.jpg
http://cs.dwp.alab.edu:8080/...     PORTA
http://...it/scr.pl?dati dati inviati ad un programma
http://...it/fun/joke.html#ultima  frammento

file://fun/joke.html                ("la risorsa è sul client")
```

- URL = Uniform Resource Locator
- URI = Uniform Resource Identifier

NB (URL is a locator; URI is an identifier)

Nome della risorsa (2/2) - localizzazione fisica

La risorsa richiesta
http://lweb.dis.uniroma1.it/fun/fig/borg.jpg
si trova

- su un certo web server (calcolatore, con un certo indirizzo IP)

Per indirizzare il calcolatore si usano nomi “logici”
che vengono mappati sugli effettivi indirizzi IP
lweb.dis.uniroma1.it
oppure direttamente gli indirizzi IP
151.100.16.198

e

- in una certa posizione nel file system di quel web server, identificata dal path: es. **/fun/fig/borg.jpg**

(**NB** **/**=“base del web server”)

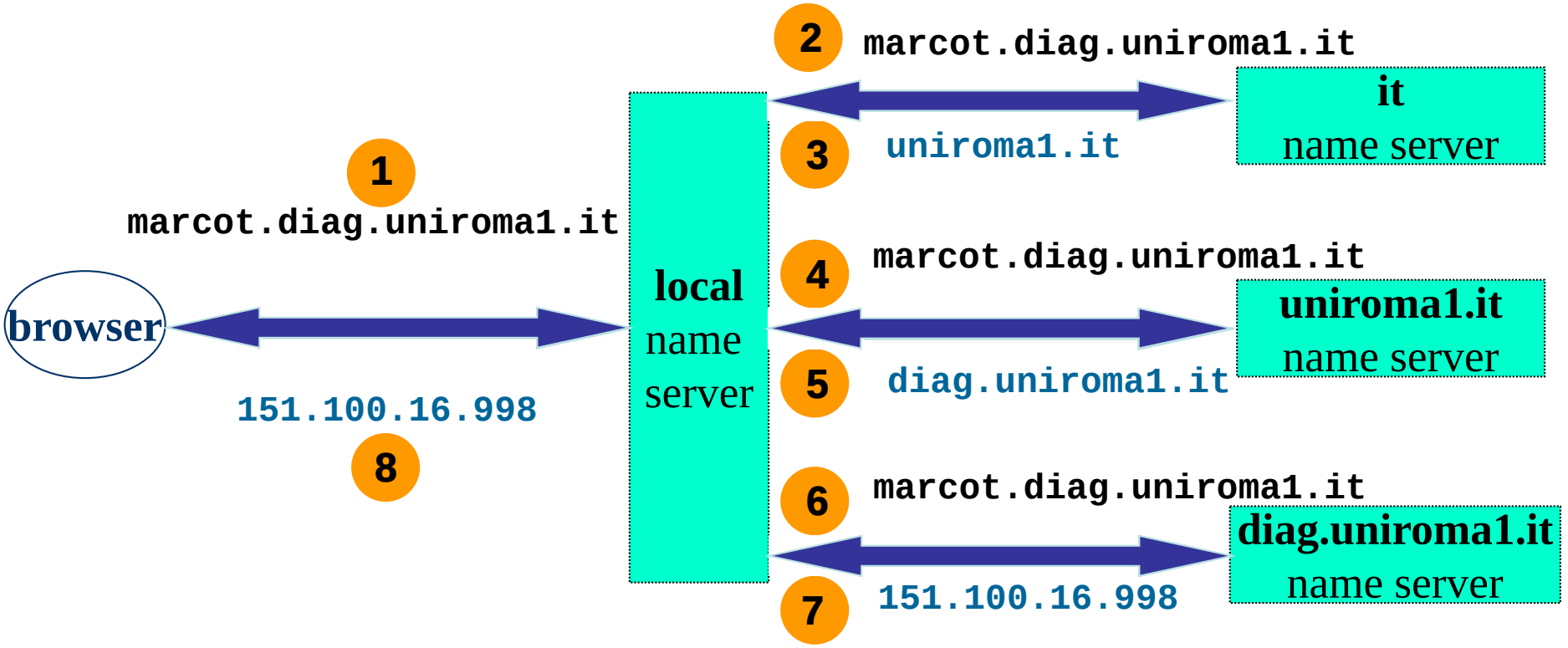
DNS - Domain Name System (1/2)

lweb.diag.uniroma1.it

Top Level Domain	org	edu	com	uk	it	eu
Domini II° livello	mozilla apache		about poste		uniroma1 uniroma2	
Domini III° livello					diag	
zone / servizi	www2		www		lweb	

DNS - Domain Name System (2/2)

Risoluzione di un indirizzo



HTTP1.1/2 – request/response

- Il cliente fa una **richiesta** tramite la porta 80 (default), specificando la URI della risorsa
- Il server, che e` in ascolto su quella porta, esegue
  - Apertura connessione per servire la richiesta
  - **Risposta**
  - Ciclo di processazione altre richieste del medesimo client
  - Chiusura connessione

Specifica il

- **metodo** (cioe` l'azione richiesta), URI della risorsa richiesta e versione del protocollo usato per la rich.
- **Intestazione** (ad es. con i tipi MIME accettati – cioe` gestibili - dal richiedente: coppie **header:value**)
- Una linea vuota seguita dal corpo

```
method URI ver
header: value
...
...
...
header: value

object
```

```
GET .../borg.html HTTP/1.0
Accept: text/*
User-Agent: Mozilla/…
If-Modified-Since: Sun, 2…
Accept-language: en

...ev.dati
```

Request (metodi e header ~)

Metodi

GET: Used to request data from a specified resource. *Richiesta lettura dato (oggetto, payload*)
POST: Used to submit data to be processed to a specified resource.
PUT: Used to upload a representation of a resource to the server.
DELETE: Used to request the removal of a specified resource.
HEAD: Similar to GET but returns only the response headers without the response body.
OPTIONS: Used to describe the communication options for the target resource.
TRACE: Used to perform a message loop-back test along the path to the target resource.
CONNECT: to establish a network connection to a specified resource, usually for proxying purposes.

Headers

Host	Nome host da cui proviene la richiesta
User-Agent	Identificazione del browser
Authorization	Può mantenere dati per l'autenticazione
Referer	URL da cui è stato ottenuto quello richiesto
Accept	Specifica dei media-types accettati nella risposta
If-Modified-Since	Richiesta condizionale
Cookie	Contains cookies associated with the request.
scheme	http, https ...
path	Specifies the path of the request URI.

risposta

- **Versione protocollo + codice di stato + testo espl.**
- **Intestazione** (coppie header-value)
- Una **linea vuota**, seguita dalla risorsa fornita in risposta alla richiesta

ver codice_stato testo
header: value
…
…
…
…
header: value

object

```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 10 Oct … GMT
Server: Apache/1.3.213 (Unix)
Last-Modified: Mon, 2 … GMT
Content-Length: 6931
Content-Type: text/html

<?xml version = "1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 ...
<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="en" lang="en"> <head> <title>entities
(da virtual library)</title> </head>
<body> ...
```

HTTP Response: Status codes / Headers

Status Code
200 OK
403 Forbidden
404 Not Found
500 Internal Server Error

Status Code (categorie)	
1	Informativo
2	Successo
3	Reindirizzo
4	Errore dovuto al client
5	Errore dovuto al server

Header	**Descrizione**
Server	Web server al lavoro
Last-Modified	Data dell'ultima modifica della risorsa
Content-length	Dimensione in bytes della risorsa
Content-type	Formato MIME della risorsa
Expires	Data dalla quale la risorsa va aggiornata
Set-Cookie	Sets cookies on the client.

MIME Types

Multipurpose Internet Mail Extension AKA Media Type

una definizione standard che serve a classificare il formato delle risorse che vengono inviate via web. Specifica della *Internet Engineering Task Force* (IETF RFC 6838).

Riferimento: pagina IANA (*Internet Assigned Numbers Authority* – responsabile per la definizione e documentazione dei MIME Types)
https://www.iana.org/assignments/media-types/media-types.xhtml

schema: Tipo/Sottotipo;parameter=value

application/sql, text/plain, text/*, font/otf, image/png, image/svg+xml, audio/x-wav, application/rtf, text/plain;charset=UTF-8

esempi

Type	Subtype	Description
Text	Plain	Unformatted text
	HTML	Text with HTML tags
Image	Gif	Still picture in GIF
	Jpeg	Still picture in JPEG
Audio	Basic	Audible sound
Video	Mpeg	MPEG movie
Application	Octet-stream	Uninterpreted byte sequence
	Postscript	Postscript document

(Sessioni e) cookie

- HTTP e` un protocollo ***stateless***
- E allora come mantenere le informazioni sullo "stato" delle transazioni tra client e server??
- Un metodo usato:
  - Server invia **header Set-cookie**
  - cookie = coppia **nome/attributo** ... anche **path**
  - Client invia **header Cookie**

(Sessioni e) cookie

- Il protocollo e` ***stateless***, cioe` nessuna informazione viene conservata tra un'interazione (req/res) e l'altra

- Pero`, informazioni sullo "stato" dei rapporti in corso tra client e server potrebbero far comodo in certi casi:
  - dati per "adattare" le risposte a certe preferenze del client;
  - dati di autenticazione, per ricordare al server che il client si e` gia` autenticato poco fa …

- Un metodo usato:
  - Il server invia al client uno o piu` cookie, inserendo in una risposta l'header **Set-cookie**
  - Un cookie e` una coppia **nome/attributo** (eventualmente associata ad un "path di competenza" ('/' o '/foo` negli esempi dopo)
  - Nelle successive richieste, il client include uno header "Cookie**"**, rimandando la coppia (o le coppie) al server (se la richiesta e` per un documento nel path di competenza del cookie)

Esempio di transazioni con cookie (1/2)

- **Il client richiede un documento;**
- **nella risposta il server invia (e il client riceve)**

Set-Cookie: CUSTOMER=WILE_E_COYOTE; path=/;
expires=Wednesday, 20-Dec-39 20:12:00 GMT

Da adesso, quando il client fa una richiesta (URL nel path "/" del server), manda i cookie che gli sono stati settati ... tra cui

Cookie: CUSTOMER=WILE_E_COYOTE **(se prima dell'expire date)**

`piu` tardi …`

- **Il client richiede un altro documento** (e invia o no il cookie precedente, ma questo non e` importante ora)
- **e riceve, dal server:**

Set-Cookie: PART_NUMBER=ROCKET_LAUNCHER_0001; path=/

Da ora, se il client richiede una URL nel path "/" del server, manda (tra l'altro)**:**

Cookie: CUSTOMER=WILE_E_COYOTE;
PART_NUMBER=ROCKET_LAUNCHER_0001

sempre se prima dell'expire date)

Esempio di transazioni con cookie (2/2)

piu` tardi …

- Il Client riceve dal server:

  Set-Cookie: SHIPPING=FEDEX; path=/foo

Dopo, se il client richiede una URL nel path "/" del server, manda (tra l'altro) :

Cookie: CUSTOMER=WILE_E_COYOTE;
PART_NUMBER=ROCKET_LAUNCHER_0001

Ma, se il client richiede una URL nel path "/foo" del server, manda (tra l'altro)

Cookie: CUSTOMER=WILE_E_COYOTE;
PART_NUMBER=ROCKET_LAUNCHER_0001;
SHIPPING=FEDEX

(il cookie viene inviato dal client solo se l'invio accade prima dell'expire date)

Esempio di transazioni con cookie (monopagina)

- **Il client richiede un documento e riceve, dal server:**

  Set-Cookie: CUSTOMER=WILE_E_COYOTE; path=/;
  expires=Wednesday, 20-Dec-39 20:12:00 GMT

Dopo, se il client richiede una URL nel path "/" del server, manda (tra l'altro):

Cookie: CUSTOMER=WILE_E_COYOTE **(se prima dell'expire date)**

`piu` tardi …`

- **Il client richiede un altro documento** (e invia o no il cookie precedente, ma questo non e` importante ora)
  e riceve, dal server:

  Set-Cookie: PART_NUMBER=ROCKET_LAUNCHER_0001; path=/

Dopo, se il client richiede una URL nel path "/" del server, manda (tra altro eventuale)**:**

Cookie: CUSTOMER=WILE_E_COYOTE;
PART_NUMBER=ROCKET_LAUNCHER_0001

`piu` tardi …`

- **Il Client riceve dal server: Set-Cookie: SHIPPING=FEDEX; path=/foo**

Dopo, se il client richiede una URL nel path "/" del server, manda (tra l'altro) **:**

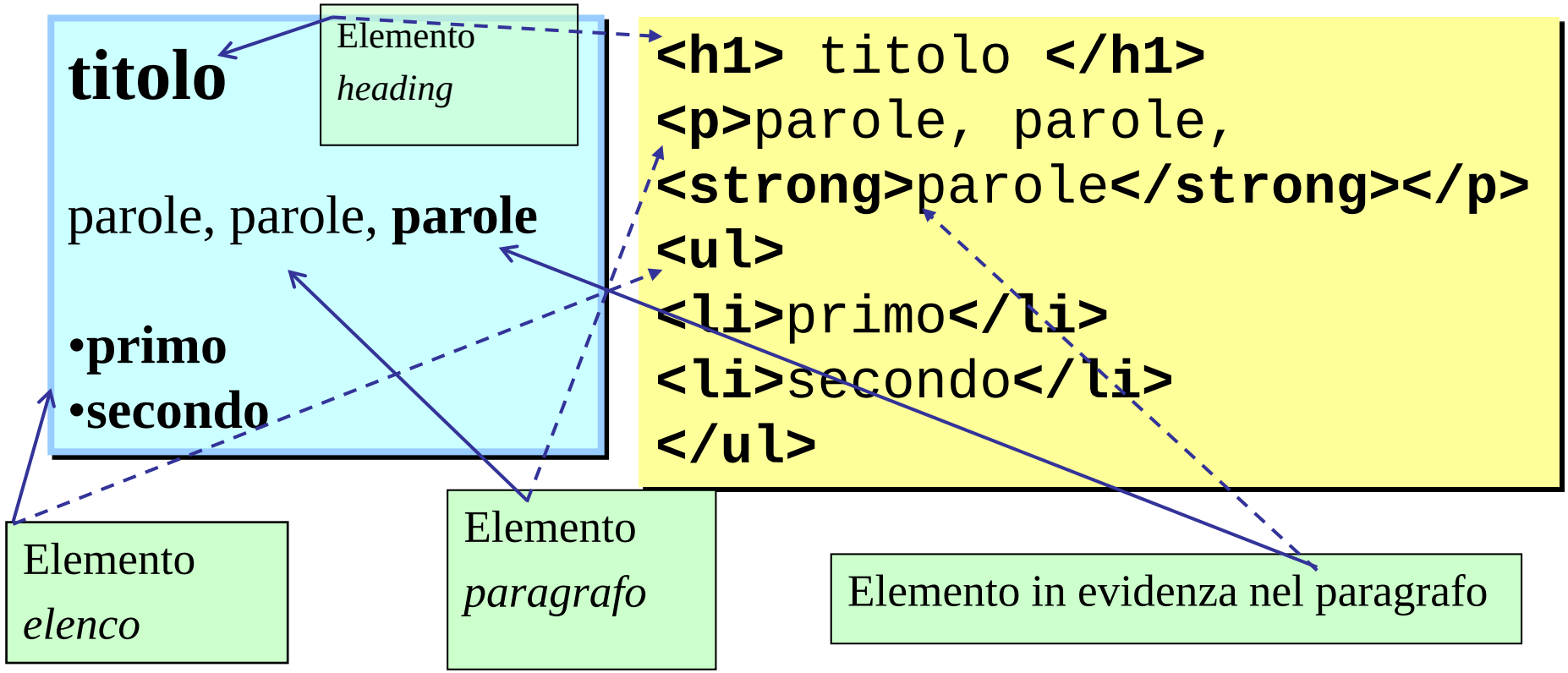
Cookie: CUSTOMER=WILE_E_COYOTE;
PART_NUMBER=ROCKET_LAUNCHER_0001

Ma, se il client richiede una URL nel path "/foo" del server, manda (tra l'altro)

Cookie: CUSTOMER=WILE_E_COYOTE;
PART_NUMBER=ROCKET_LAUNCHER_0001;
SHIPPING=FEDEX

Linguaggio di markup in generale + caso XHTML

Linguaggio che permette di scrivere documenti testuali, marcando (qualificando) ciascuna data parte come "elemento di un certo tipo", in modo che acquisisca un certo *significato*



"*significato*" = **ruolo** dell'elemento nel documento, in base al quale il software di analisi del documento (**es. browser**) opera (**es. visualizza**)

elementi e tag, in generale e per XHTML

linguaggio di markup

elementi che contengono ...

delimitati da **tag**

```
<h1> titolo </h1>
<p>parole, parole, <strong>parole</strong></p>
<ul>
<li>primo</li>
<li>secondo</li>
</ul>
```

contenuto = *parti di testo e riferimenti alle risorse web coinvolte*.

ruolo che quel contenuto ha? Descritto dalla semantica dell'elemento (paragrafo, titolo, altro ...).

Il web browser riceve tag e contenuto,
li processa e, in base al ruolo del contenuto,
ne esegue la "presentazione"
(es. visualizzazione).

elementi e tag, in generale e poi per XHTML

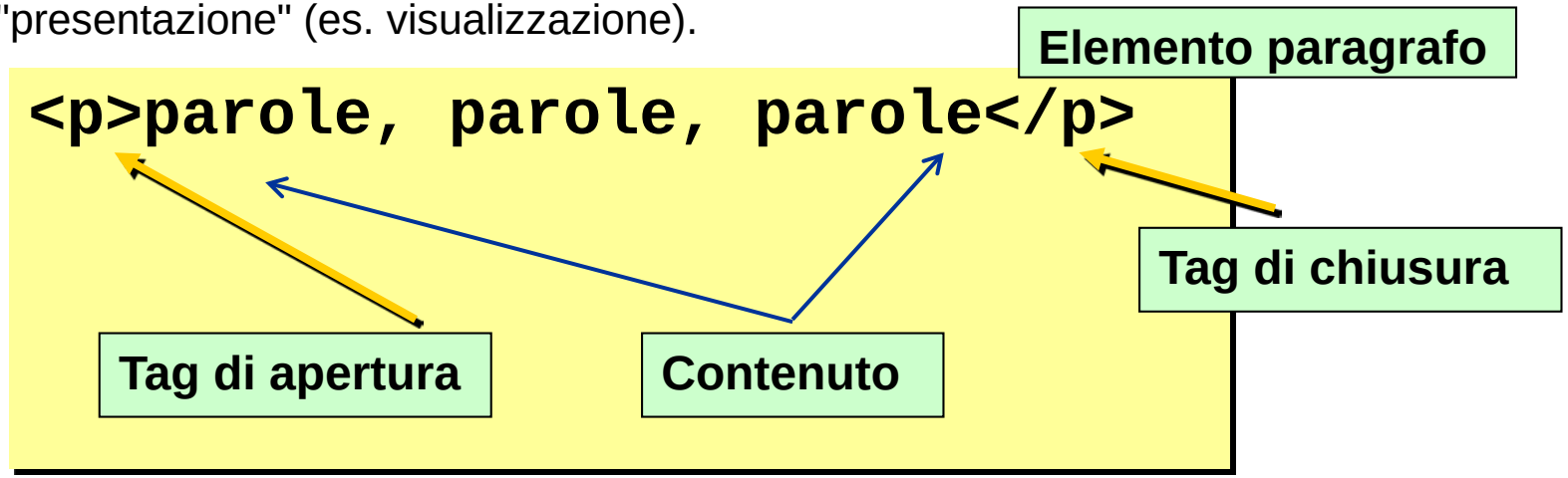
Il documento scritto in un linguaggio di markup e` composto da **elementi**, che ne convogliano il contenuto informativo (sostanzialmente testo).

```
<h1> titolo </h1>
<p>parole, parole,
<strong>parole</strong></p>
<ul>
<li>primo</li>
<li>secondo</li>
</ul>
```

Possono esistere infiniti linguaggi di markup. Uno e` XHTML.

Nel caso di (x)html, il contenuto e` costituito dalle varie *parti di testo e riferimenti alle risorse web coinvolte*. L'appartenenza di un certo contenuto ad un elemento (ovvero la delimitazione del contenuto tramite tag di xhtml) fornisce l'indicazione del ruolo che quel contenuto ha nel documento (paragrafo, titolo, altro ...).

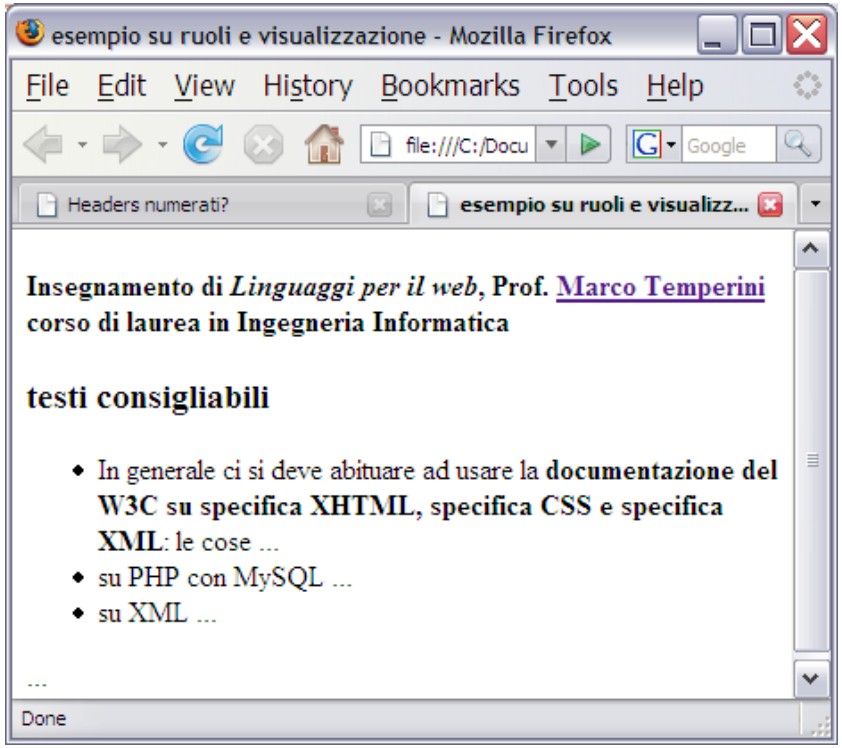
Il web browser riceve tag e contenuto, li processa e, in base al ruolo del contenuto, ne esegue la "presentazione" (es. visualizzazione).



(No, i tag non vengono "presentati"…).

Esempi (1/2)

```
<h4>
Insegnamento di <em>Linguaggi per il web</em>,
Prof. <a href="http://www.dis.uniroma1.it/~marte/...">Marco ...</a>
<br />
corso di laurea in Ingegneria Informatica
</h4>

<h3>testi consigliabili</h3>
```



<h4> elemento *block-level (?)* **TITOLO**

<a> elemento **con contenuto** "Marco ..."
e attributo
href="http://www.dis.uniroma1.it/~marte/..."

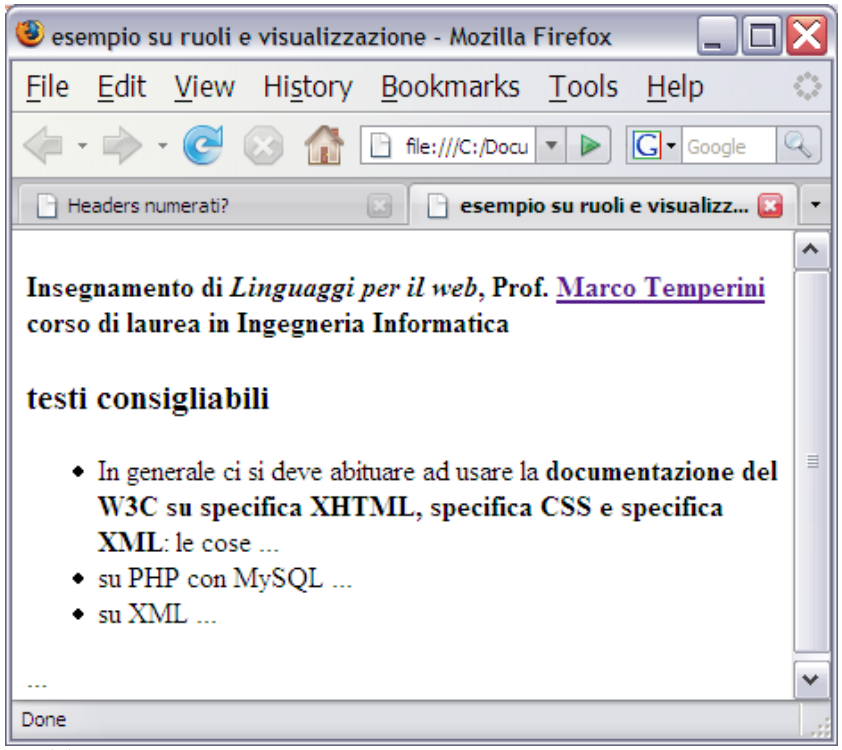
**** e` un elemento *inline (?)* per testo *in evidenza*

**
** e` un elemento non "strutturale"!

<h3> e` un altro elemento **TITOLO**

...(2/2)

```
...</h4>
<h3>testi consigliabili</h3>
<ul>
 <li>In gen ... <strong>documentazione del W3C ...</strong>: le...</li>
 <li>su PHP con MySQL ... </li>
 <li>su XML ...  </li>
</ul>
...
```



<h3> e` un elemento **TITOLO**

<ul> e` un elemento **LISTA** (una lista **contiene** elementi "punto di lista")

<li> e` un elemento **che costituisce un punto di lista**

**** e` un elemento *inline* per testo in **forte** evidenza

Attributi?

Ogni elemento puo` avere degli **attributi** , cioe` delle parole chiave associate a valori, che funzionano come delle specie di parametri, destinati a dare delle informazioni che completino o specifichino in modi alternativi il ruolo dell'elemento nel documento.

Gli attributi di un elemento - cioe` gli attributi che e` **ammissibile o obbligatorio** usare in un elemento - sono previsti nella definizione formale del linguaggio di markup.

Dato un elemento del documento, quindi, suoi attributi possono (*o devono*) essere assegnati esplicitamente a valori.

L'applicazione che riceve il documento decide come trattare l'elemento, oltre che in base alla sua definizione, anche in base al valore specificato per I suoi attributi.

Cosi`, nel caso di un documento XHTML, l'applicazione "web-browser" che lo riceve con lo scopo di presentarlo, visualizzera` (presentera`) l'elemento anche in base ai valori assegnati ai suoi attributi

Elemento tabella (bordi grossi)

```
<table border="12" cellspacing="5">...</table>
```

Elemento tabella (senza bordi)

```
<table border="0" cellspacing="5">...</table>
```

Elemento img

```
<img src = "img/2.1.jpg" width = "100" alt = "leo" />
```

Elementi vuoti

Si tratta di elementi per i quali non e` previsto un contenuto da visualizzare.

Tutte le informazioni che, eventualmente, servono sono espresse tramite gli attributi (vedi img, che e` un elemento vuoto).

Il tag di chiusura (visto che non c'e` contenuto testuale, e` particolare.

Elemento br

```
<br />
<br> </br> (possibile ma no …)
<br/>      (no …)
```

Elemento hr

```
<hr />
```

Elemento img

```
<img src = "img/2.1.jpg" width = "100" alt = "leo" />
```

Storia (yawn ...)

Molto a grandi linee ...

- 1989, Tim Berrners-Lee fonda il web (bum!) per organizzare documenti in internet. Mediante:
  - Strumento per leggere “web pages”
  - Linguaggio per definire web pages
  - Protocollo per permettere che lo strumento possa chiedere ed ottenere web pages
- Prima? ftp, email, news? ... per (nemmeno tanto) pochi ...
- Sviluppo di HTML 1.1 (92), 2.1 (‘95), 3.2 (‘97), 4.01 (‘99) e delle relative versioni dei browser (Mosaic... Netscape, Opera, IE). Browser wars
  - Elementi proprietari
  - Javascript/jscript
  - Code forking
- Web page = documento HTML contenente STRUTTURA E PRESENTAZIONE, regolata da elementi del linguaggio HTML, e comportamento (regolato da linguaggi di scripting)

“POSSIBLY “BEST VIEWED BY ...”

- Problemi del code forking (un modello di business sepolto parecchi anni fa = COSTI MAGGIORI - per chi produce il sito e per chi lo paga)
  - Codice proprietario (esclusione di larghe fette di utenti e potenziali clienti)
  - Codice ridondante (larghezza di banda per trasmetterlo ...)
  - Manutenzione complessa (differenza tra aggiornare una unica versione del sito, scritta in un codice rigoroso e asciutto, e aggiornare o aggiungere le versioni browser-dependent)
- Sviluppo e affermazione degli standard per il web (W3C - da recommendation a standard) ... continua ...

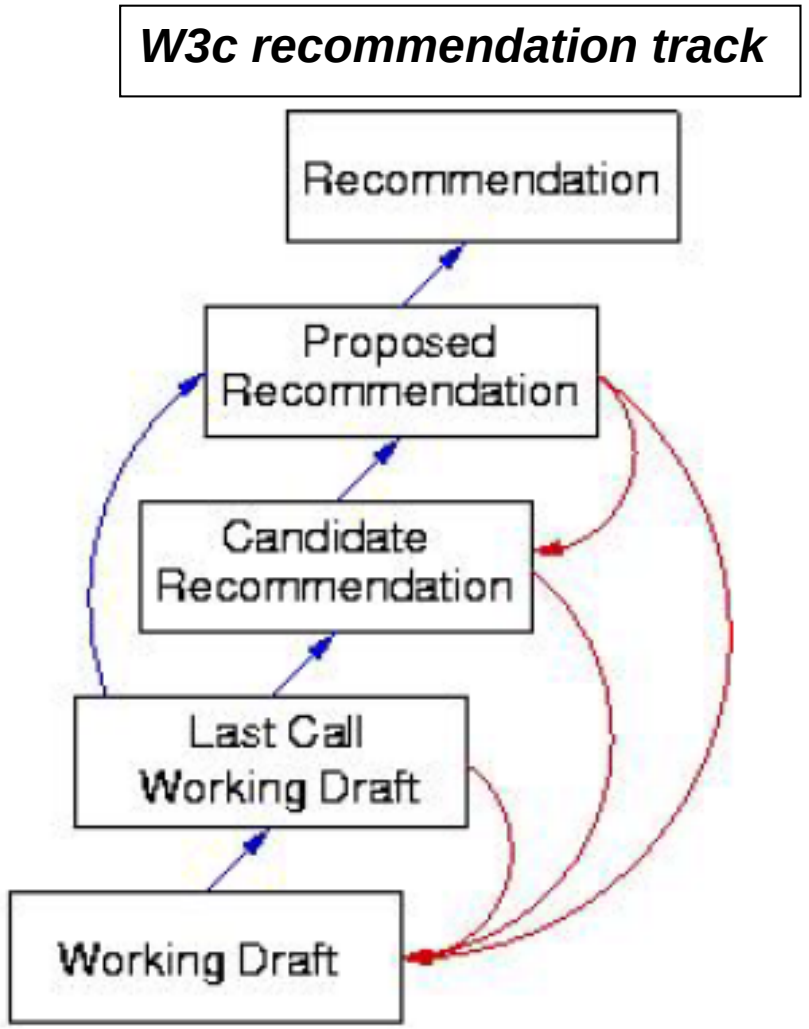
HTML

Linguaggio di markup usato per costruire documenti specificando come ciascun tipo di parte (elemento) dovra` (piu` o meno …) essere visualizzata da un web browser

Definito mediante SGML

(Standard Generalized Mark.Lang.)

HTML 2 … HTML 3.2 … HTML 4.01 stop



XML

Sottoinsieme di SGML (meno vasto e piu` facile da usare).

XHTML

Linguaggio di markup, replica di HTML4.01, definito (cioe` “i cui elementi sono stati definiti”) usando XML. E` quello che evolve.

http://www.w3.org/TR/tr-groups-all#tr_Web_Applications_Working_Group

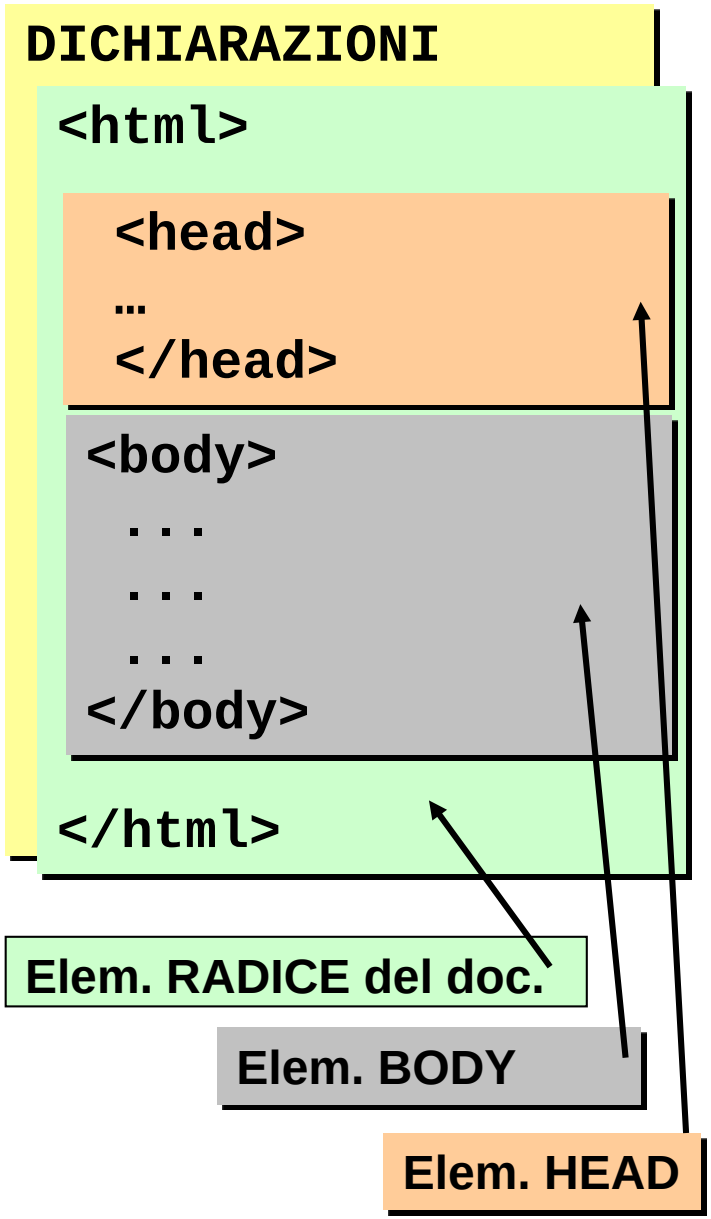
Esempio di definizione di un elemento

specifica di come dovra` essere scritto un elemento, cioe` di quali sono le parti (elementi) mediante le quali il documento potra` o dovra` essere composto

(si usa la sintassi SGML)

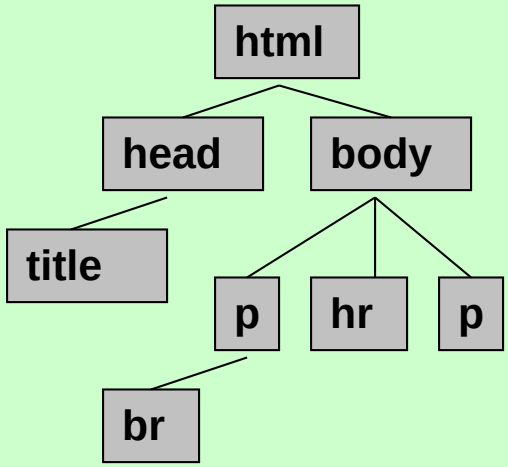
```
<!ELEMENT img EMPTY>
<!ATTLIST img
   %attrs;
   src %URI; #REQUIRED
   alt %Text; #REQUIRED
   longdesc %URI; #IMPLIED
   height %Length; #IMPLIED
   width %Length; #IMPLIED
   usemap %URI; #IMPLIED
   ismap (ismap) #IMPLIED
>
```

Struttura base del documento XHTML




```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="en" lang="en">

<head>
<title>Struttura base documento XHTML</title>
</head>

<body>

<p>
Tanto rumor per poco.
<br/>
poco rumor per poco.
</p>
<hr/>
<p>
altro rumor per tanto.
</p>

</body></html>
```

albero del documento



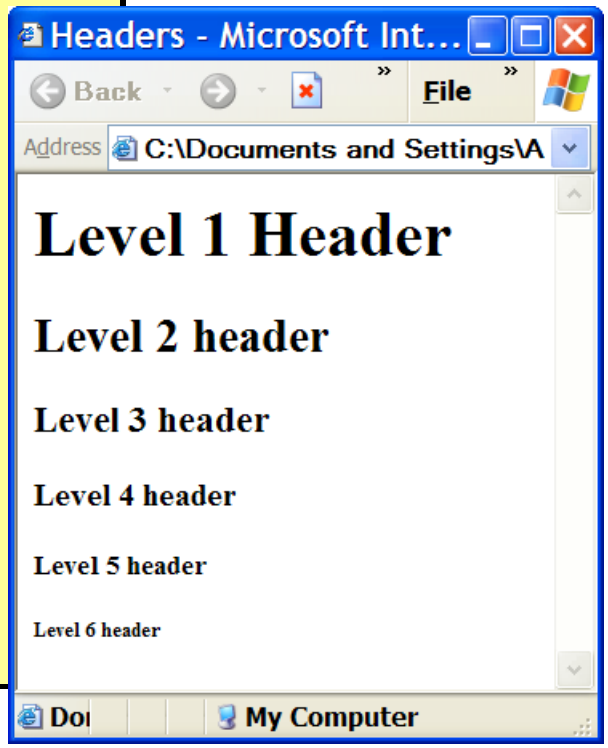
~~Enfatizzano testo~~, sempre di meno al crescere del numero ...

un classico esempio di *elemento strutturale*:

indica che

"il contenuto e` un titolo"

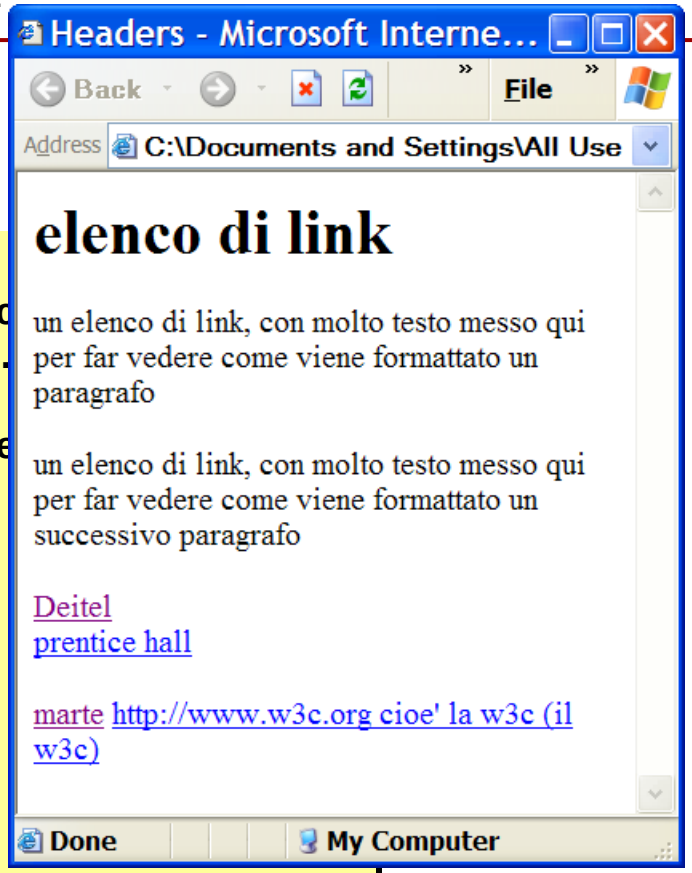
```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
lang="en">
       <head>
         <title>Headers</title>
      </head>

      <body>

         <h1>Level 1 Header</h1>
         <h2>Level 2 header</h2>
         <h3>Level 3 header</h3>
         <h4>Level 4 header</h4>
         <h5>Level 5 header</h5>
         <h6>Level 6 header</h6>

      </body>
   </html>
```



link.html

L'elemento anchor <a> determina il collegamento tra il testo incluso e una url, fornita come attributo

```
<?xml version = "1.0"?>
    <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Stric
       "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="e
<head><title>Headers</title></head>
      <body>
         <h1>elenco di link</h1>

 <p>un elenco di link … </p>
 <p>un elenco di link … </p>
 <p>
    <a href = "http://www.deitel.com">Deitel</a>
   <br />
    <a href = "http://www.prenhall.com">prentice hall</a>
 </p>
 <p>
  <a href = "http://www.dis.uniroma1.it/~marte">marte</a>
  <a href = "http://www.w3c.org">http://www.w3c.org cioe` la w3c (il
w3c)</a>
 </p>
      </body>
   </html>
```



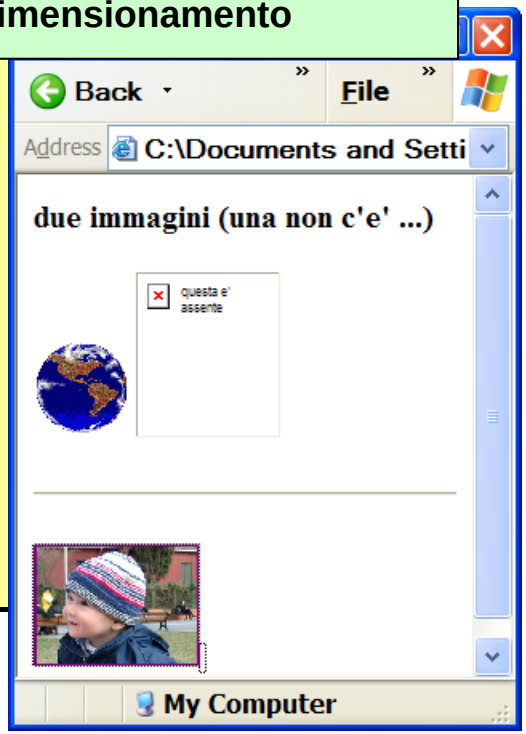
Il line break manda a capo (nella presentazione), rimanendo nel medesimo paragrafo

qui si va a capo nel file html, non nella presentazione …

Immagini: elemento <img>

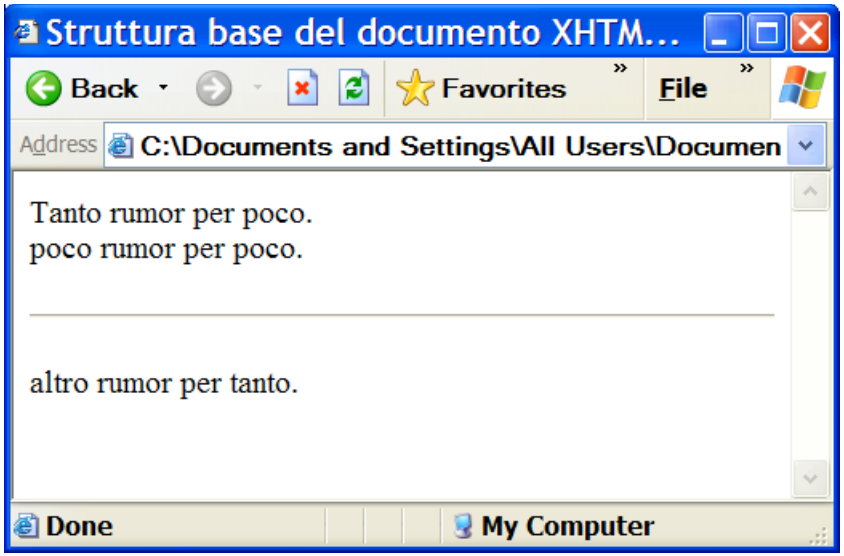
img.html

```
...
<head>
   <title>immagini ed immagini link</title></head>

<body>
    <h3>due immagini (una non c'e` ...)</h3>

    <p><img src = "img/anim_earth.gif"
           height = "60" width = "60"
           alt = "gif animata" />

       <img src = "anim_earth.gif" width="90"
            alt = "questa e` assente" />
    </p>
    <hr />
    <p>
     <a href="http://www.dis.uniroma1.it/~marte">
        <img src = "img/2.1.jpg"
             width = "100"
             alt = "leo" />
     </a>
    </p>
</body></html>
```



perche' usare gli elementi? (1/2)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="en" lang="en">

<head>
<title>Struttura base documento
XHTML</title>
</head>

<body>

<p>
Tanto rumor per poco.
<br />
poco rumor per poco.
</p>
<hr />
<p>
altro rumor per tanto.
</p>

</body>
</html>
```



NB

l'interpretazione

(*la visualizzazione*)

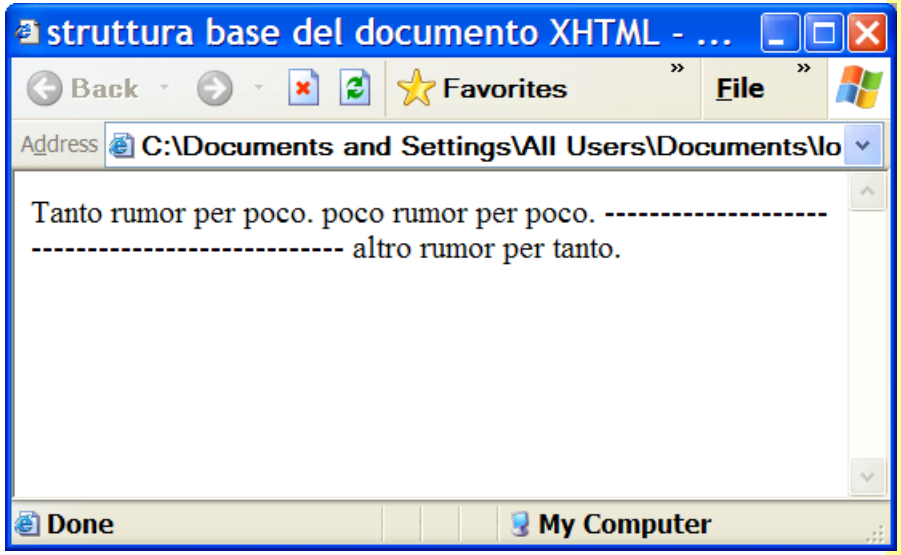
del documento e` sequenziale...

un elemento, poi il successivo, poi ...

perche' usare gli elementi? (2/2)

NB

il testo non contenuto in elementi (non qualificato con un ruolo nel documento), normalmente, viene spedito sullo schermo cosi` come viene trovato




```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="en" lang="en">

<head>
<title>struttura base del documento
XHTML</title>
</head>

<body>

Tanto rumor per poco.
poco rumor per poco.

----------------------------------------
-----

altro rumor per tanto.

</body>
</html>
```

Struttura base del doc. XHTML (1/3) - dichiarazioni

Documento html aderente alla *document type definition* indicata dal file .dtd).
PUBLIC corrisponde al fatto che la DTD e` definita universalmente e non localmente.
(SYSTEM e` invece usato per definizioni locali al server).

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.1 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
...
```

Si fa riferimento ad un documento di tipo DTD, curato dal W3C, relativo a XHTML1.1 Strict (EN=english), identificato dalla URI …

Piu` possibilita` di scelta per il tipo di documento cui si vuole essere conformi

in conseguenza della scelta, il browser interpretera` secondo regimi diversi il codice nel documento

(DOCTYPE DECLARATION)

```
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

Struttura base del doc. XHTML (2/3) - prologo XML

"prologo XML" non indispensabile se lo scopo e` solo la visualizzazione; indica che il nostro documento e` un doc. XML che rispetta lo standard XML 1.0 ed e` scritto usando i caratteri della codifica UTF 8 bit.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.1 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
...
```

codifica unicode = associazione codici/caratteri, indipendente da piattaforma, software, SO. (Es. `iso-8859-1`=Latin-1).

Per aggiungere la codifica dei caratteri del doc si puo` usare anche il seguente nell'elem. **<head>**

```
<meta http-equiv="Content-Type" content="text/html" charset="ISO-8859-1" />
```

Struttura base del doc. XHTML (3/3) - dichiarazione el. radice

```
<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="en" lang="en">
```

xmlns=NAMESPACE

serve perche' il nostro e` ormai un doc. XML: XML ammette l'occorrenza di tag uguali in scope (aree del medesimo doc) diverse: ciascuna occorrenza potrebbe avere significato diverso, determinato dal fatto che nello scope agisce un linguaggio di markup diverso; cio` puo` generare conflitti (a quale linguaggio di markup appartiene il tag qui?).

Il namespace e` una URI che permette l'associazione di un prefisso univoco al tag, per risolvere l'ambiguita`.

xml:lang="codice linguaggio"
lang="codice linguaggio"
(servono tutti e due)

```
<table …>…
<furniture:table>…
<paint:table>…
```

Il documento deve essere well-formed

- elementi e attributi in minuscolo

SI --- `<em>seconda</em>` NO --- `<EM>seconda</EM>`

- elementi ben nidificati:

`<p>metti la <em>seconda</em></p>`

`<p>metti la <em>seconda</p></em>`

- elementi sempre terminati dal corretto end tag:

`<p>metti la <em>seconda</em></p>`

`<p>metti la <em>seconda</em>`

- elementi vuoti sempre terminati:

`<hr />, <br />` `<hr>, <br>`

- attributi sempre e tra virgolette

`<table cellspacing="14">` `<table cellspacing=14>`

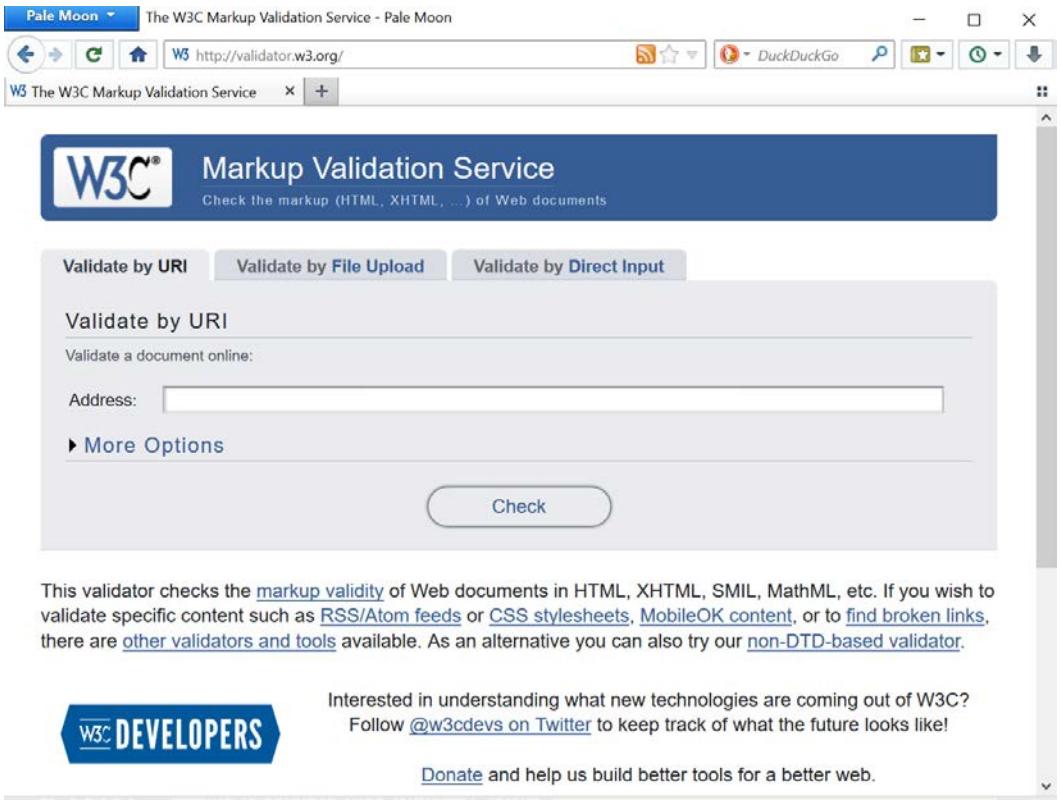
raccomandazioni per salvaguardare la "compatibilita`":

- usare `<hr />` invece di `<hr/>` non usare `<hr></hr>` che pure sarebbe permesso ("element minimization")
- non fare *element minimization* su elementi non vuoti, anche se occasionalmente lo sono (cioe` non fare ` ` ma ` `

(ma poi, perche' isolare un paragrafo vuoto? ... Per fare spazio ci sono altri modi ...)

… e valido

Un documento e` valido se rispetta la grammaica di riferimento, cioe`, nel caso dei documenti xhtml, la dtd specificata in testa al documento.



Rispettare la grammatica vuol dire

- usare gli elementi consentiti dalla grammatica di riferimento
- e rispettare la struttura degli elementi: un elemento puo` avere assegnati certi attributi e non altri, gli attributi possono essere di certe tipologie e formati, l'elemento può contenere altri elementi, oppure no …

I browser (sebbene ci siano ancora eccezioni qui e la`) fanno affidamento sulla grammatica per interpretare (dal punto di vista delle presentazione) il contenuto del documento

e` conveniente sviluppare in XHTML corretto, per

- garantire (il piu` possibile) che il sito web abbia presentazioni consistentemente simili da parte di browser diversi
- e anche perche' solo cosi` le caratteristiche piu` avanzate di documenti possono essere sfruttate appieno (ad es. quelle riguardanti l'accessibilita` per i disabili e l'adattabilita` a diversi ambienti mediali).

Storia (bzzz ...)

- Sviluppo e affermazione degli standard per il web (W3C - da *recommendation* a *standard*)

Con le difficolta` a far accettare le raccomandazioni da tutti ...

- Mosaic/Netscape immagini nel doc!!!! Aggiungiamo <img> ad HTML ...
  - W3C: no, usate <object>
  - Nobody cares
- Future spLASH plug-in, RealAudio, QuickTime: nuove risorse utilizzabili in un doc
  - Netscape aggiunge <embed>
  - W3C: no, usate <object>

Un'altra storia:

-2004: W3C announces XHTML2.0 – no backward compatibility; forget HTML, and XHTML

-Web Hypertext Application Technology WorkGroup says WHATwg!? *Vogliamo supporto ad HTML, meno rigore, backward compatibility, e non vogliamo chiudere *

(non e` proprio cosi` ☺)

-2007-2009: W3C cancella il progetto XHTML2.0

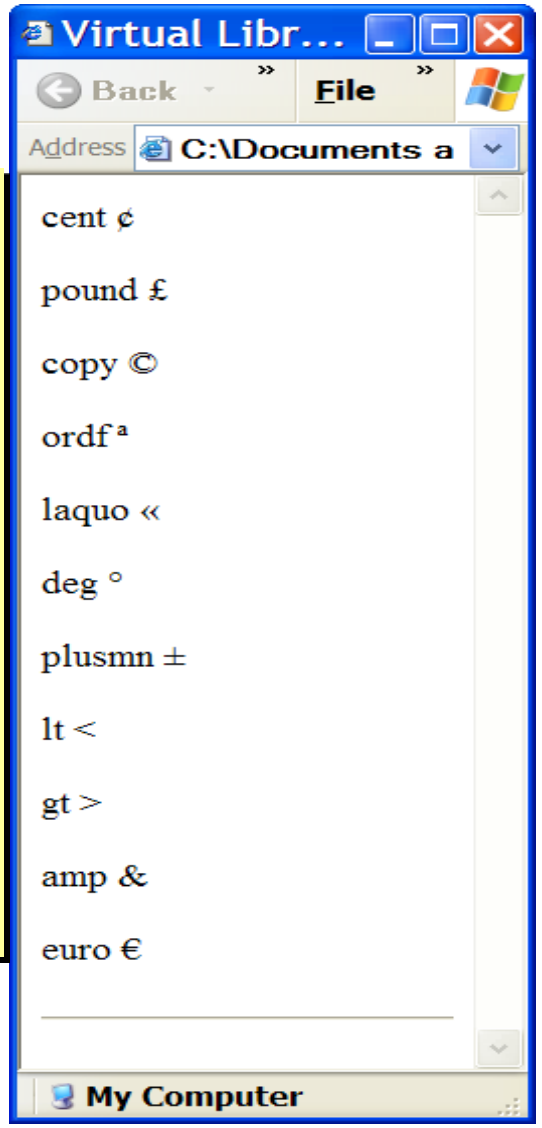
-HTML5 arises

- Compatibile con XHTML
- Meno rigoroso e disciplinato ma per documenti piu` interattivi (device capabilities; video and animations; graphics; editing, style, typography, and other)
- <nav>, <footer>, ... <aside> per layout
- <video>, <audio>, <canvas>, no prologue, doctype simple, no , drag&drop

entities.XHTML.html

Oggetti definiti NELLA GRAMMATICA DEL LINGUAGGIO
per inserire nel testo dei documenti caratteri speciali
USO: **&nomeEntity;**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en"
lang="en">
<head><title>Virtual Library</title></head>
<body>
<p>cent &cent;</p>
<p>pound &pound;</p>
<p>copy &copy;</p>      <!-- parte tagliata (questo
...                         e` un commento)    -->
<p>lt &lt;</p>
<p>gt &gt;</p>
<p>amp &amp;</p>
<p>euro &euro;</p>
<hr />
</body></html>
```



strutturaBase.XHTML.3.html
strutturaBase.XHTML.4.html
fanno uso estensivo di entita` per riportare e descrivere parti di codice

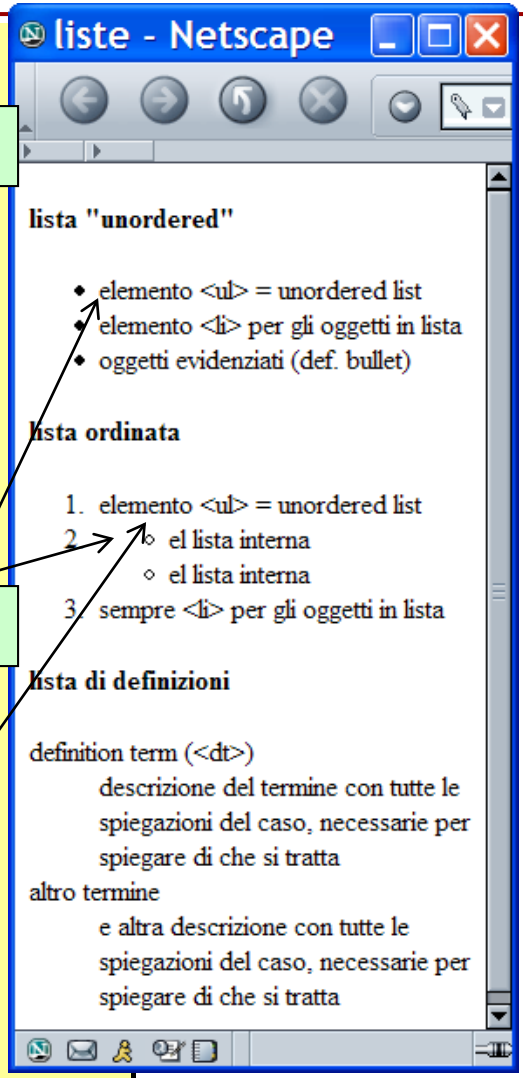
liste.html

```
…
<body>
<h4>lista "unordered"</h4>
<ul>
 <li>elemento &lt;ul&gt; = unordered list</li>
 <li>elemento &lt;li&gt; per gli oggetti in lista</li>
 <li>oggetti evidenziati (def. bullet)</li>
</ul>

<h4>lista ordinata</h4>
<ol>
 <li>elemento &lt;ul&gt; = unordered list</li>
 <li> <ul>
      <li>el lista interna</li>
      <li>el lista interna</li>
      </ul>
 </li>
 <li>sempre &lt;li&gt; per gli oggetti in lista</li>
</ol>

<h4>lista di definizioni</h4>
<dl>
 <dt>definition term (&lt;dt&gt;)</dt>
 <dd>descrizione del termine ... si tratta</dd>
 <dt>altro termine</dt>
 <dd>e altra ... tratta</dd>
</dl> ...
```

, <ol>, <dl>
tipo di lista

 oggetto in lista

lista innestata



Diversi simboli/contatori disponibili
(mediante style – cioe` css)

tabelle1.html

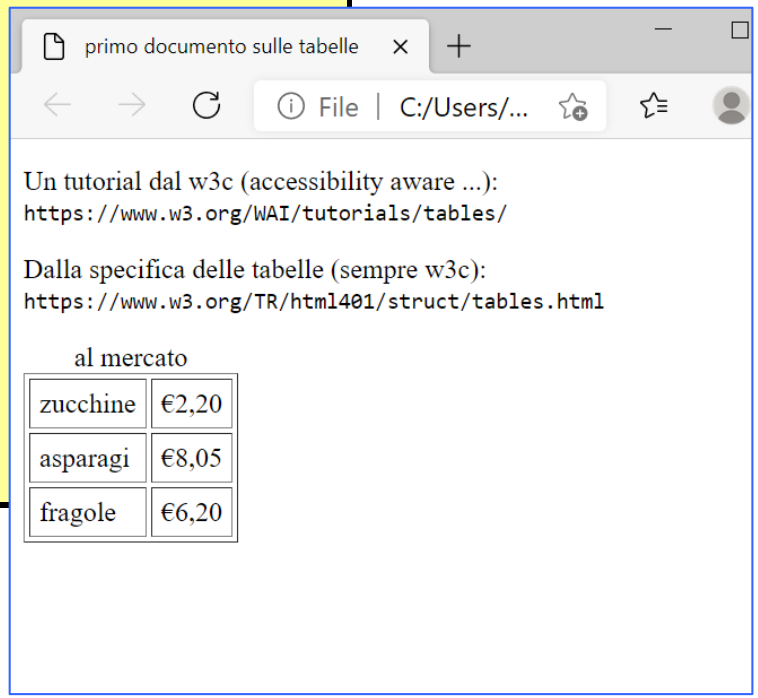
```
…
<table border="1" cellspacing="3" cellpadding="5"
       summary="spiegazioni sul significato della tabella">

<caption>al mercato</caption>

<tbody>
 <tr>
  <td>zucchine</td>
  <td>&euro;2,50</td>
 </tr>
 <tr>
  <td>asparagi</td>
  <td>&euro;3,05</td>
 </tr>
 <tr>
  <td>fragole</td>
  <td>&euro;6,20</td>
 </tr>

</tbody>
</table>
...
```

Titolo tabella

Attributi table: spessore bordo, spazio tra celle, spazio intorna al dato in una cella; spiegazioni

Corpo tabella

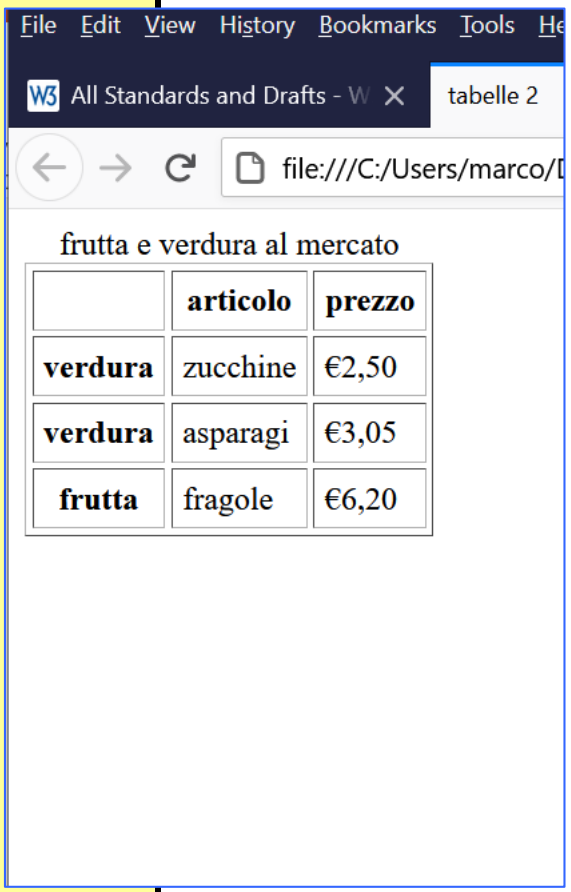
<tr> RIGA </tr>
<td> CELLA </td>



```
…
<table  ...>
<caption>frutta e verdura al mercato</caption>

<thead>
 <tr>
  <th></th>
  <th>articolo</th>
  <th>prezzo</th>
 </tr>
</thead>

<tbody>
 <tr>
  <th>verdura</th>
  <td>zucchine</td>
  <td>&euro;2,50</td>
 </tr>
 <tr>
  <th>verdura</th>
  <td>asparagi</td>
  <td>&euro;3,05</td>
 </tr>
 <tr>
  <th>frutta</th>
  <td>fragole</td>
  <td>&euro;6,20</td>              </tr>     </tbody></table>...
```

<thead> = Intestazione tabella

<th> = cella di intestazione colonna o riga (default: centrato e grassetto)



padding?

Elem. footer tabella

```
…
<table ... cellpadding="10” ...>
<caption>frutta e verdura al mercato</caption>
<thead style="background-color: green">
 <tr>  <th></th>
  <th>articolo</th>
  <th>prezzo</th> </tr>
</thead>

<tfoot style="background-color: pink">
 <tr>
  <th>tipo</th>
  <th>articolo</th>
  <th>prezzo</th>
 </tr>
 <tr>
  <td colspan="2">mala tempora currunt</td>
 </tr>
</tfoot>

<tbody>
...</tbody></table> ...
```

2 righe nel footer (volendo anche nell’header)

questa cella copre 2 colonne

frutta e verdura al mercato

	articolo	prezzo
verdura	zucchine	€2,50
verdura	asparagi	€3,05
frutta	fragole	€6,20
tipo	articolo	prezzo
mala tempora currunt		

```
…
<table ...>
<caption>frutta e verdura al mercato</caption>
<thead style="background-color: green">
 ...
</thead>

<tfoot style="background-color: pink">
...
</tfoot>

<tbody>
 <tr>
  <th rowspan="2">verdura</td>
  <td>zucchine</td>
  <td>&euro;2,50</td>
 </tr>
 <tr>
  <td>asparagi</td>
  <td>&euro;3,05</td>
 </tr>
 <tr>
  <th>frutta</td>
  <td>fragole</td>
  <td>&euro;6,20</td>
 </tr>
</tbody>
</table> ...
```

questa cella copre 2 righe

ELEMENTI BLOCK-LEVEL / IN-LINE

Un elemento block-level viene per default visualizzato nettamente separato dagli altri elementi: **<p>, <table>, le liste**

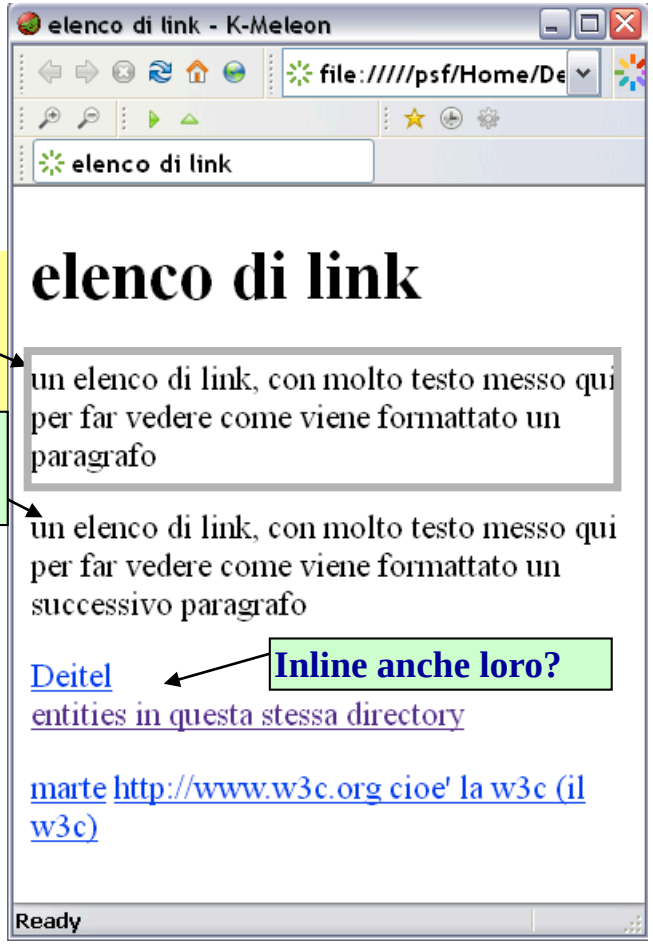
Un elemento in-line viene visualizzato con continuita` rispetto al testo che lo circonda **<img>, , ** … **<a>**

```
<p>un elenco di link … </p>

<p>un elenco di link … </p>
```

Due elementi block level (due blocchi separati)

```
<p>
<a href = "http://www.dis.uniroma1.it/~marte">marte</a>
<a href = "http://www.w3c.org">http://www.w3c.org cioe`
la w3c (il w3c)</a>
 </p>
```

Due elementi inline (inseriti, uno dopo l'altro senza interruzioni, in un elemento block-level)



Inline anche loro?

** dummy block level element**

** dummy inline element**

usati per racchiudere e isolare parti di documento

in un medesimo "ambiente" con regole di presentazione (*style*) proprie;

Se nel dummy element e` definito uno stile, questo si applichera` a tutti gli elementi contenuti (li vedremo poi …).

link a parti interne del documento 1/3 (courtesy Deitel&Deitel)

```
1  <?xml version="1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. 5.6: links.html -->
6  <!-- Internal Linking     -->
7
8  <html xmlns="http://www.w3.org/1999/xhtml">
9    <head>
10      <title>Internet and WWW How to Program - List</title>
11   </head>
12
13   <body>
14
15      <!-- <a name=".."></a> creates an internal hyperlink -->
16      <p><a name="features"></a></p>
17      <h1>The Best Features of the Internet</h1>
18
19      <!-- An internal link's address is "#linkname"        -->
20      <p><a href="#ceos">Go to <em>Favorite CEOs</em></a></p>
21
22      <ul>
23        <li>You can meet people from countries
24          around the world.</li>
25
26        <li>You have access to new media as it becomes public:
27          <ul>
28            <li>New games</li>
29            <li>New applications
30              <ul>
31                <li>For Business</li>
32                <li>For Pleasure</li>
33              </ul>
34            </li>
35
```

To internally link, place a # sign in front of the name of the desired anchor element within the page.

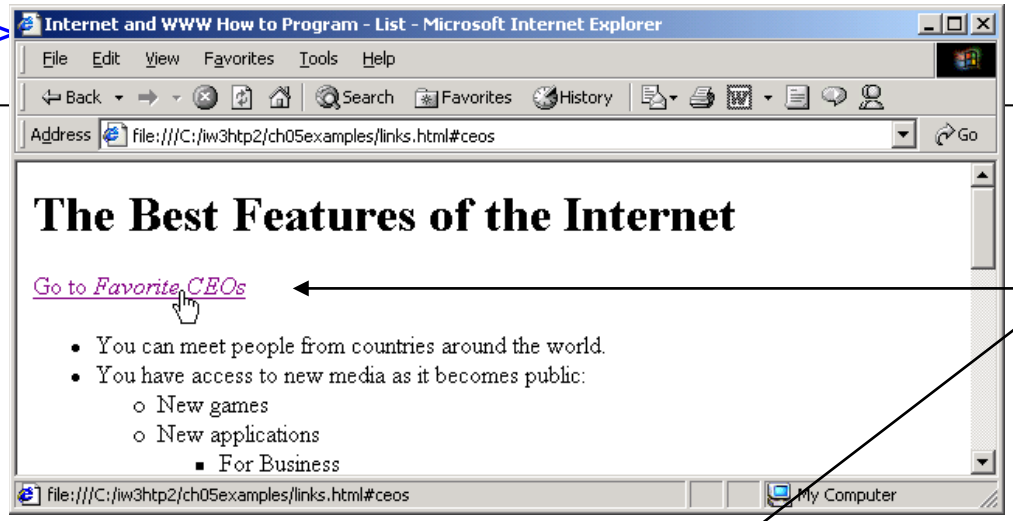
link a parti interne del documento 2/3 (courtesy Deitel&Deitel)

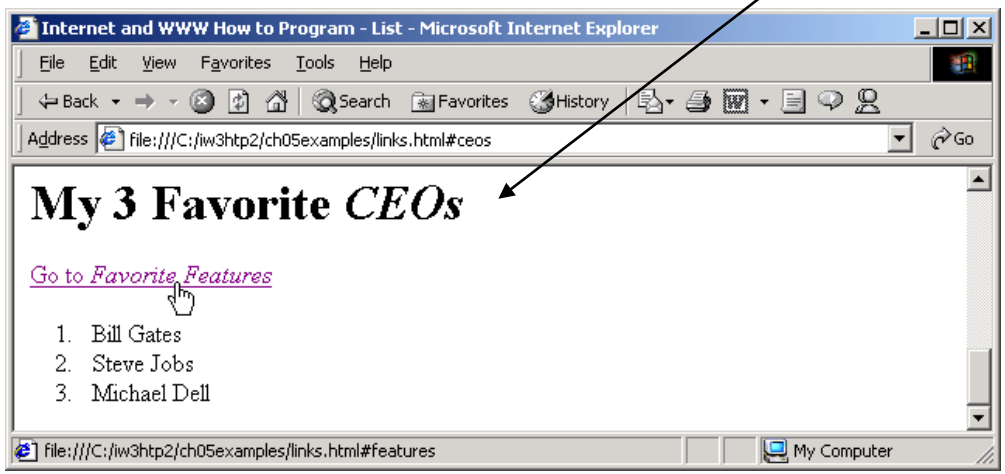
```
36          <li>Around the clock news</li>
37          <li>Search Engines</li>
38          <li>Shopping</li>
39          <li>Programming
40            <ul>
41              <li>XHTML</li>
42              <li>Java</li>
43              <li>Dynamic HTML</li>
44              <li>Scripts</li>
45              <li>New languages</li>
46            </ul>
47          </li>
48        </ul>
49      </li>
50
51      <li>Links</li>
52      <li>Keeping in touch with old friends</li>
53      <li>It is the technology of the future!</li>
54    </ul>
55
56    <!-- Named anchor -->
57    <p><a name="ceos"></a></p>
58    <h1>My 3 Favorite <em>CEOs</em></h1>
59
60    <p>
61
62      <!-- Internal hyperlink to features -->
63      <a href="#features">Go to <em>Favorite Features</em>
64      </a></p>
65
```

An anchor named **ceos** will be created at this point on the page. This anchor does not link and will not be seen on the page. However, other anchors can refer to this anchor and link to it.

link a parti interne del documento 3/3 (courtesy Deitel&Deitel)

```
66      <ol>
67        <li>Bill Gates</li>
68        <li>Steve Jobs</li>
69        <li>Michael Dell</li>
70      </ol>
71
72    </body>
73  </html>
```





Clicking on this internal link will bring the user to the bottom of the page where **My 3 Favorite CEOs** is located.

Seguono suggerimenti per esercizi da fare, e poi risorse di approfondimento

Il material per gli esercizi e` nella directory XHTML.CSS.1, che giace, compressa, nell'area dove sono pubblicate le slide delle lezioni

Attivita` in laboratorio / prodotti individuali (1/6)

XHTML-1

ripercorrere gli esempi presentati a lezione: aprire, consultare, modificare i file di esempio presentati a lezione; verificare quali cambiamenti determinano le modifiche (chi si aspetta un cambiamento e non lo vede deve approfondire ...). Per modificare un file di esempio, prima duplicarlo con un nome diverso e poi agire modificando solo la copia ...). Vedi gli esercizi seguenti: forse vuoi interrompere questo esercizi quando ti senti in grado di affrontare uno dei prossimi. Poi riprendi sempre questo esercizio fino a terminarlo.

XHTML-2

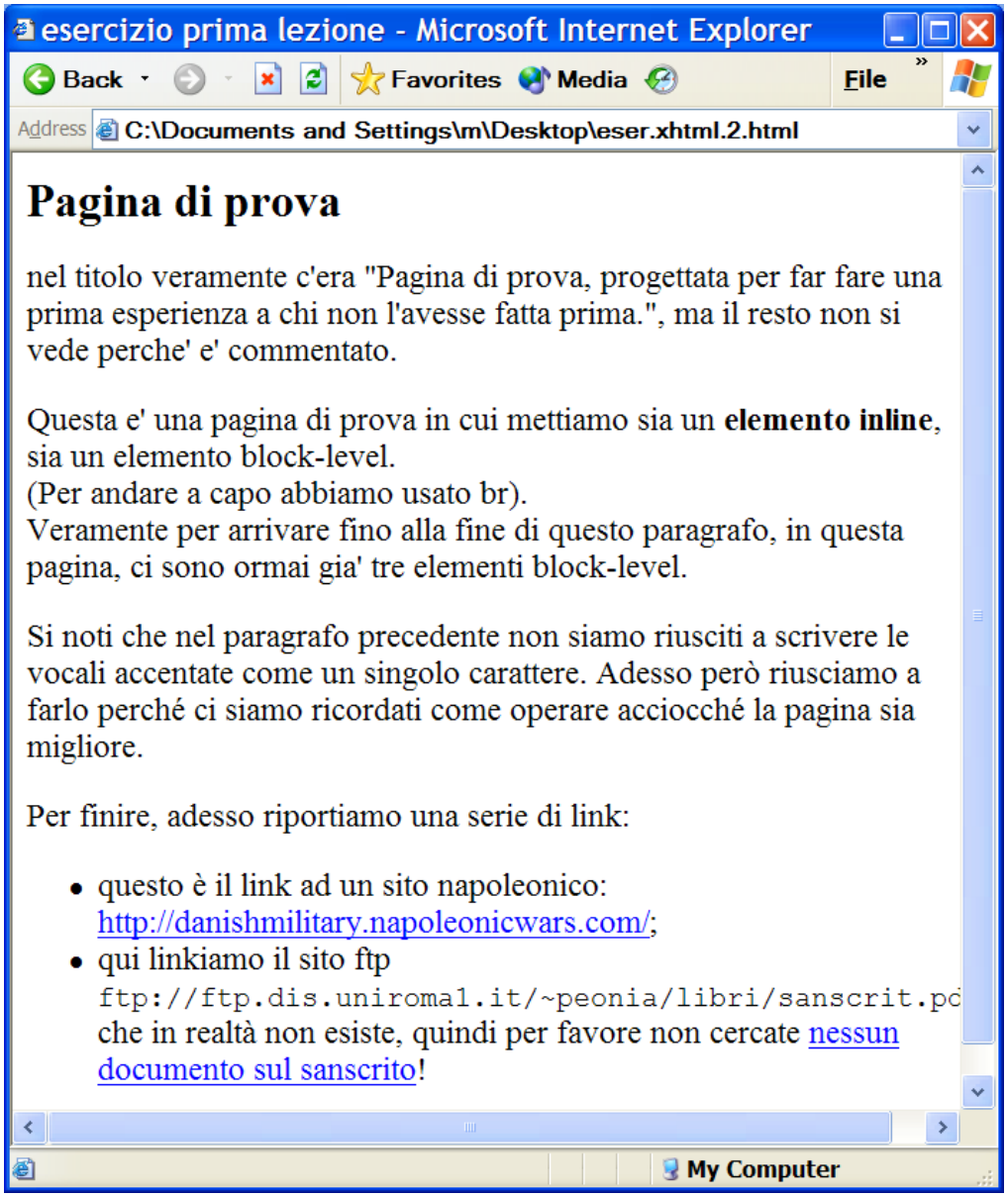
produrre un file `eser.xhtml.2.html` che replichi quanto in figura, per quanto possibile. Spiegare brevemente il significato degli elementi utilizzati, in un commento all'interno del codice xhtml.

Forse userai **é è à** e `cos&igrave;` (o era `cos&iacute;`?) via ...

Il file `eser.xhtml.2.html` va costruito ex-novo, anche se con cut&paste da file usati precedentemente. Insomma si vorrebbe che l'autore del file mettesse gli occhi e le dita su ciascuna linea del file ...

XHTML-3

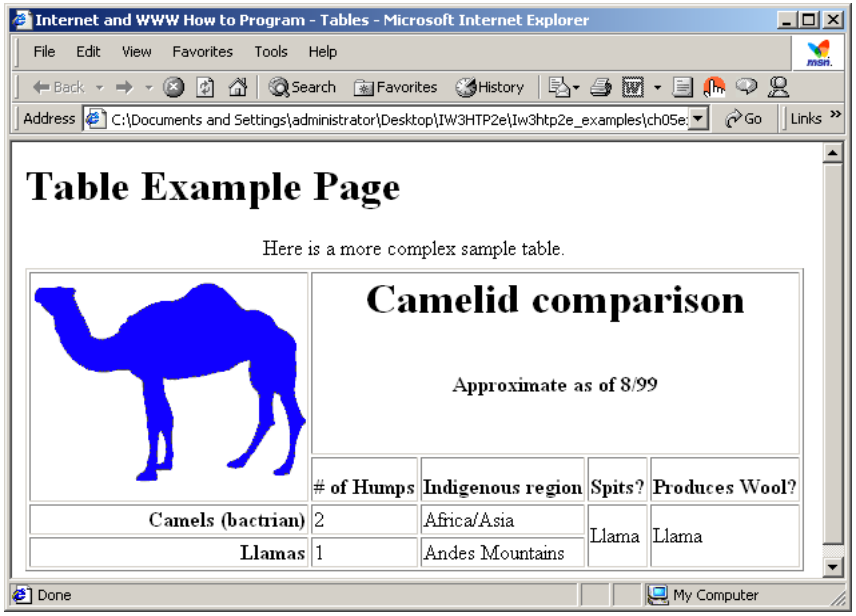
usare il validatore del W3C per verificare se si e` scritto codice valido nel file prodotto precedentemente `eser.xhtml.2.html`.



Attivita` in laboratorio / prodotti individuali (2/6)

XHTML-4

Nel libro di Deitel&Deitel appare la tabella a fianco riportata (thanks). Replicarla in un proprio documento html. Si puo` usare una figura alternativa a quella del cammello ...



XHTML-5

produrre un file eser.xhtml.5.html in cui siano presentati (separatamente) almeno **due elementi XHTML**,(almeno uno tra tra p, h..., strong, em, hr, br, e almeno uno tra link e img).

Per ciascuno spiegare la definizione e far vedere esempi di uso. Per i concetti fare riferimentro alla definizione XHTML 1.0 o 1.1 del W3C.

NB In questa prima versione, questo documento fa poco o nessun uso di comandi di stile e non usa tabelle ne` liste.

XHTML-6

Produrre una seconda versione, `eser.xhtml.6.html`, piu` ricca, dell'esercizio precedente:
Organizzare i contenuti in almeno tre pagine distinte. Aggiungere a quanto gia` fatto la spiegazione di list. Il documento va strutturato usando tutte le armi XHTML a disposizione (anche liste e tabelle). Limitare al minimo (o anche a zero) l'uso di comandi di stile.

XHTML-7

usare il validatore del W3C per verificare se si e` scritto codice valido nel file eser.xhtml.6.html precedentemente prodotto.

XHTML-8

Scrivere una o piu` pagine web che discutano la definizione dei link in XHTML, soffermandosi anche su tipi non discussi a lezione (fare riferimento alla documentazione del W3C). Ad esempio, cosa e` mailto? Questo e` un buon posto per descrivere l'uso dell'attributo title in un elemento <a>. Un altro attributo interessante e` tabindex, che serve per alterare l'ordine dei link raggiunti attraverso la pressione del tab. E perche' non parlare anche di accesskey, menzionato nel file di esempi sui link?

Attivita` in laboratorio / prodotti individuali (3/6)

XHTML-9

In questo esercizio bisogna anche familiarizzare con l'uso dell'attributo "title" in un elem. anchor. A che serve? e` utile? Investigare. Comunque, dopo aver completato l'analisi (del resto) della parte xhtml di lezione, scrivere un documento xhtml `eser.xhtml.4.html` ispirato a quello in figura: il layout della pagina e` organizzato mediante una tabella di una sola riga e due celle. Si puo` presumere che la pagina sia quella introduttiva di un sito (da realizzare fittiziamente) e i link in essa presenti permettano la navigazione del sito. Il fiore e` `fiore.jpg` nella directory `XHTML.CSS/img`

XHTML-10

Cosa succede se in file .html vengono usati tag non riconosciuti dal browser? Ad esempio <grande> oppure <grassetto>. Il browser rifiuta di visualizzare il file? Gli elementi corrispondenti a tag non riconosciuti (unrecognized tag) sono ignorati? Il testo degli elementi non riconosciuti viene visualizzato ma con una formattazione di default? Il testo viene visualizzato, ma con la formattazione dell'elemento di livello superiore in cui e` contenuto quello non riconosciuto? 1) Prima si pensi a quale risposta si crede giusta; 2) poi la si verifichi. Costruire una pagina web che descrive alcuni di questi problemi e le relative soluzioni. Estendere facendo dapprima uso di piu` pagine distinte e linkate (ove necessario) una all'altra. I file siano tutti contenuti in una medesima directory.

XHTML-11

Scrivere una pagina web in cui sia spiegato il meccanismo di uso delle entita`; a proposito, cosa sono le entita`? Fare esempi di entita`, mostrandole e definendole brevemente; spiegare per ciascuna da dove, nella documentazione o su testi, sono state tratte

XHTML-12

Facendo riferimento alla documentazione e al testo, scrivere una o piu` pagine web che discutano i tag di formattazione diretta del testo: considerare quelli suggeriti dalla documentazione (ad esempio, <b> e <i> non dovrebbero interessarci, perche'? Che vuol duire "deprecato" nello standard?). Cosa e` em? Cosa e` strong? Cosa e` sup? Cosa e` del? Cosa e` sub?

Attivita` in laboratorio / prodotti individuali (4/6)

XHTML-13

scrivere una pagina web in cui viene spiegato che cosa e` un linguaggio di markup; menzionare, nei limiti delle conoscenze apprese o introdotte a lezione, diversi linguaggi di markup; fare esempi relativi ad almeno un linguaggio di markup conosciuto.

XHTML-14

scrivere un documento xhtml in cui si spiega la funzione di una document type description, mostrando esempi nei quali se ne veda l'occorrenza nel codice di una pagina html. Strict? Transitional? Frameset?? Di cosa si tratta? Utilizzare la documentazione per rispondere.
Costruire un piccolo sito web in cui si illustrino e si mettano in pratica le differenze tra le tre definizioni. Se I file (come auspicabile) sono piu` di uno, fare in modo che siano tutti contenuti una medesima directory e facciano riferimento l'uno all'altro (ove necessario). sub?

XHTML-15

Consultare l'HTML tutorial di w3schools (http://www.w3schools.com/html) non e` affatto una cattiva idea; anche fare i test li` presentati e` cosa buona. Pero` consultate anche le differenze tra HTML attuale ed XHTML:
https://www.w3schools.com/html/html_xhtml.asp

XHTML-16

Definire una pagina web, non necessariamente fatta di piu` documenti, in cui si discute il problema della selezione delle modalita` di resa di una pagina da parte del browser (rendering mode selection). Ci sono modi diversi (standard, quirks, quasi-standard) che vengono selezionati in base a ...

Come sorgente di informazioni, oltre al testo e alle note in fondo a queste slide, ad esempio http://en.wikipedia.org/wiki/Quirks_mode,
http://www.ericmeyeroncss.com/bonus/render-mode.html

Attivita` in laboratorio / prodotti individuali (5/6)

HTML5, qualche risorsa e un esercizio.

Riguardo ad html5, la pagina del w3c (https://www.w3.org/TR/html5/) fornisce molte interessanti informazioni, mentre quella di w3schools (https://www.w3schools.com/html/html5_intro.asp) permette di giocare un po' e rendersi conto delle caratteristiche di html5.

e` bene occuparsi di questo argomento dopo aver acquisito conoscenza e fatto esercizi sulla parte precedente (xhtml).

L'esercizio consiste nel consultare la presentazione «`The Web is 28 years old.pdf`», in directory XHTML-CSS, che da` un'idea delle attese suscitate anni fa da html5 e delle caratteristiche che questa nuova tecnologia motte a disposizione.

Poi, seguire i link li` contenuti e consultare la pagina dedicata ad html5 su w3schools, per

produrre un file eser.html5.html in cui siano presentate (separatamente) almeno due caratteristiche giudicate interessanti in html5, cosi` come fatto in esercizi precedenti per gli elementi di xhtml.

Per ciascuna caratteristica spiegare il concetto e far vedere esempi di uso.

CSS non lo abbiamo ancora visto e non lo usiamo per questi esercizi. Pero` se quando si stanno facendo questi esercizi la lezione su CSS e` gia` stata fatta, allora li si puo` estendere usando le cascading style sheet (cio` vale anche per i precedenti in realta`)

XHTML-CSS-1

scrivere un documento xhtml in cui si spiega la differenza tra elementi block level e inline; se si usa CSS per la formattazione del documento, scegliere di specificare regole inline, oppure con una internal style sheet

XHTML-CSS-2

scrivere un documento xhtml in cui si spiega il significato dell'espressione "documento valido". Come si puo` verificare la validita` di un documento xhtml? Se si usa CSS per la formattazione del documento, scegliere di specificare regole inline, oppure con una internal style sheet

XHTML-CSS-3

scrivere un documento xhtml in cui si spiega il significato dell'espressione "documento ben formato". Come si puo` verificare la well-formedness di un documento xhtml? Se si usa CSS per la formattazione del documento, scegliere di specificare regole inline, oppure con una internal style sheet

Attivita` in laboratorio / prodotti individuali (6/6)

XHTML-CSS-4

scrivere un sito web, composto da almeno due pagine xhtml, in cui sia trattato l'argomento della buona formazione e validita` di un documento xhtml. Definire opportune regole CSS per la formattazione omogenea dei documenti, usando una external style sheet; nella discussione fare esempi di documenti validi e non validi, well formed e non well formed.

XHTML-CSS-5

Realizzare una pagina web il cui contenuto e` arricchito con delle spiegazioni associate a parole/espressioni contenute nel testo. Si tratta di aggiungere dei cosiddetti link di help nella pagina.

Tipicamente si affianca alla parola una icona di help che fa accedere alle infomazioni necessarie (come nuova pagina, come finestra *pop up*, come *tooltip* - nella lezione su xhtml abbiamo visto, ad esempio, come far apparire del testo relativo ad una immagina quando il cursore del mouse va sopra l'immagine. Non e` l'unico modo.)

Come fare queste cose va cercato (non chiesto semplicemente ...), senno` l'esercizio e` inutile.

Un posto dove cercare potrebbe essere inizialmente http://www.ericmeyeroncss.com/bonus/proj04-excerpt.html

XHTML-CSS-6

Realizzare una pagina web in cui, usando piu` documenti html, sia delineata la struttura base del documento XHTML, spiegandone ed esemplificandone le varie parti note. Usare regole CSS per una visualizzazione omogenea dei documenti.

XHTML-CSS-7

scrivere una pagina web in cui sia spiegata, nei limiti delle conoscenze acquisite a lezione, la seguente definizione: si riconoscono elementi? si riconoscono attributi? se si usa CSS per la formattazione del documento, scegliere di specificare regole inline, oppure con una internal style sheet

```
                                                    >
<!ELEMENT img EMPTY>
<!ATTLIST img
        %attrs;
        src %URI; #REQUIRED
        alt %Text; #REQUIRED
        longdesc %URI; #IMPLIED
        height %Length; #IMPLIED
        width %Length; #IMPLIED
        usemap %URI; #IMPLIED
        ismap (ismap) #IMPLIED
```

risorse (1/2)

TBL: The Founder ***(un intervento su TED, con sottotitoli)***
https://www.youtube.com/watch?v=OM6XIICm_qo&ebc=ANyPxKroHI8PsuZqAskyabg2Cunx4f-TehSGgMu3C1sd_kzdJguBJZseFygFmbe_1z-dO9hjJX02wE99vxTFKJuIaBJAnYlRTg

sui linked data (dalla pagina del w3c sul semantic web): www.w3.org/standards/semanticweb/data

(Come detto a lezione, l'idea era gia` prevista quando la tecnologia web era ancora in corso di sviluppo. e` al semantic web che si guardava gia`, sebbene non implementabile ancora. Ecco una vecchia ma ancora perfettamente Chiara e attuale, introduzione ... http://www.xml.com/pub/a/2000/11/01/semanticweb
Una intro un po' piu` recente ...
https://graphdb.ontotext.com/documentation/free/introduction-to-semantic-web.html

una introduzione veloce e` qui https://www.youtube.com/watch?v=V6BR9DrmUQA
una un po' piu` lunga ma bella https://www.youtube.com/watch?v=rhgUDGtT2EM

www.w3c.org ***specifica di XHTML, per consultare la definizione nella DTD e la descrizione degli elementi***
(vedi anche DOCUMENTAZIONE) nella directory XHTML.CSS.1)

validazione di documenti ***www.w3c.org*** (per il validatore di markup, http://validator.w3.org/
e per quello CSS https://jigsaw.w3.org/css-validator/)
ci sono varie altre possibilita` ... https://en.wikipedia.org/wiki/CSS_HTML_Validator,
http://www.html-tidy.org/ (tidying your html ... con anche il support per HTML5),
https://www.freeformatter.com/ (vari validatori)

browser ***http://browsers.evolt.org/*** ***(ci sono tutti i browser ...)***

editor ***(textpad, notepad2, notepad++, sublime, Visual Studio Code ...*** *qualunque cosa e` molto meglio di M$ notepad****)***

w3schools ***http://www.w3schools.com***
ricco di informazioni ed esempi, riguardanti vari aspetti delle tecnologie web (da tenere nei bookmarks ...)

risorse (2/2)

Entita` https://www.w3.org/TR/xml-entity-names/ vedi anche More Advanced Features di D.Ragget nella prossima slide per una quick reference guide.
Inoltre vedi i link contenuti nel file entities.XHTML.html della directory XHTML.CSS.1 …

mime types ... oltre a w3schools e ai link suggeriti nella slide dove ne parliamo, anche http://www.mhonarc.org/~ehood/MIME/2046/rfc2046.html

Approfondimenti su DOCTYPE switching

Lascio questo link, nonostante l'eta`. Parla abbastanza semplicemente dei modi di resa grafica (quirk-old style, compliant-standard, vedi anche prossima lezione)
http://gutfeldt.ch/matthias/articles/doctypeswitch.html e` un buon punto di partenza. (vari link obsoleti)

("... both Microsoft and Mozilla decided to do something about it. The solution was two different rendering modes: The first mode, known as quirk or compatible mode, renders the pages like the old, incompatible browsers used to. The second mode, known as standard or compliant mode, renders them like the pages should be rendered according to W3C recommendations.")

qui c'e` qualcosa d'altro di leggibile a riguardo ...
- **https://developer.mozilla.org/en-US/docs/Web/HTML/Quirks_Mode_and_Standards_Mode**
- **https://en.wikipedia.org/wiki/Quirks_mode**

articoli con introduzione ad alcuni aspetti di base e meno di base (dave ragged)

Getting started with HTML	**http://www.w3.org/MarkUp/Guide/**	**una lettura un po' vintage ma piena di saggezza**
More advanced features	**http://www.w3.org/MarkUp/Guide/Advanced**	**idem**
XSlidy	**http://www.w3.org/Talks/Tools/Slidy/**	**curiosita`**