XHTML and related issues

- Per definire cos'è XHTML possiamo iniziare con una semplice espressione:

  HTML + XML = XHTML

- **HTML**: HTML è un linguaggio di marcatura per presentare i contenuti di una pagina web (per la descrizione di come appaiono documenti all'interno di un browser) . La sua semplicità è la base dell'esplosione di Internet. Fornisce strumenti per presentare titoli, paragrafi, font, link,immagini. non è estendibile a meno di non essere Microsoft o IBM. L'ultima raccomandazione rilasciata dal W3C è la 4.01 (dicembre 1999).

- Con il termine marcatura (markup) si intende una sequenza di caratteri o altri simboli che si inseriscono all'interno di un documento per indicare come il contenuto deve apparire o per descrivere la struttura logica del documento.

- **XML**: è un metalinguaggio: *"a common syntax for expressing structure in data."* ovvero un linguaggio per definire nuovi linguaggi di markup. XML consente di crearsi i propri tipi di documenti. E' quindi uno strumento per scambiare dati. E' possibile validare una struttura dati definita dall'utente. E' una specifica ufficiale del World Wide Web Consortium (W3C) ed è alla base di tutte le nuove specifiche tecnologiche rilasciate dal W3C e adottate come standard dall'industria informatica.

- Lo scopo di XML è quello di separare la definizione dei dati dalla rappresentazione, per favorire lo scambio di documenti strutturati. In particolare i principali obiettivi di XML, dichiarati nella prima specifica ufficiale (ottobre 1998), sono:
  - utilizzo del linguaggio su Internet,
  - facilità di creazione dei documenti,
  - supporto di più applicazioni,
  - chiarezza e comprensibilità.

HTML

-

Breve storia del linguaggio HTML

- Le origini dell'HTML possono essere fatte risalire ad un'idea di Vannevar Bush che descrisse un sistema per collegare in modo associativo le informazioni. Bush definì questo sistema memex (*mem*ory *ex*tension), e lo propose come mezzo per aiutare la mente umana a far fronte ad un sempre piu' crescente numero di informazioni.

- Nel 1965, Ted Nelson coniò il termine *ipertesto* per descrivere un testo che seguisse le idee di Bush. L'ipertesto, come descritto da Nelson, avrebbe dovuto collegare dei documenti per creare una rete di relazioni con l'intento di estendere ed accrescere il significato di un testo "piatto" con collegamenti ad altri testi.

- Le idee di Vannevar Bush e di Ted Nelson furono riprese in un progetto del 1989 da Tim Berners-Lee, un ricercatore del CERN di Ginevra, che propose un sistema basato sull'ipertesto per permettere una più efficiente condivisione delle informazioni tra i membri della comunità scientifica del'istituto per cui lavorava.
- La proposta di Berners-Lee (il cui titolo era "HyperText and CERN") aveva come obiettivi:
  - la creazione di un'interfaccia utente che potesse essere consistente su tutte le piattaforme, per permettere all'utente di accedere alle informazioni da diversi computer;
  - uno schema per quest'interfaccia che permettesse di accedere ad una grande varietà di tipi di documenti e di protocolli di informazione;
  - la predisposizione per un "accesso universale", che avrebbe permesso ad un qualunque utente della rete di accedere ad un qualunque tipo di informazione

- Da questo momento in poi ha inizio la storia dell'HTML, che diventa di pubblico dominio grazie all'introduzione nel febbraio del 1993 del browser Mosaic per il sistema *X Windows*. Mosaic fu sviluppato dall'NCSA (National Center for Supercomputing Applications) sotto la guida di Marc Andersen che in seguito sarebbe diventato uno dei fondatori di Netscape.
- Dal 1994 tutte le modifiche ad HTML sono avvenute sotto l'egida del World Wide Web Consortium (W3C) consorzio nato grazie agli sforzi del MIT e del CERN. Il consorzio comprende i più importanti centri di ricerca e le principali aziende del settore informatico che hanno guidando l'HTML nella sua evoluzione verso la definizione di uno STANDARD per lo sviluppo della rete.

Concetti base del linguaggio HTML

- HTML e' l'acronimo di *HyperText Markup Language*, ovvero un linguaggio di formattazione basato su marcatori che consentono di specificare l'aspetto di un documento e le relazioni o i collegamenti con altri documenti.

- La definizione del contenuto di un documento HTML avviene attraverso appositi marcatori definiti TAG.
  All'interno di ogni pagina HTML è quindi presente una serie di TAG, a cui viene affidata la visualizzazione, e che hanno differenti nomi a seconda della loro funzione. I tag vanno inseriti tra parentesi uncinate: <TAG>. La chiusura del tag viene indicata con una "/": </TAG>.

TAG

- Ai TAG sono associati attributi e agli attributi un valore. Il contenuto va inserito tra l'apertura e la chiusura del TAG medesimo.
  La struttura di un TAG è quindi: <TAG attributi>contenuto</TAG>.
  La struttura di un attributo è: attributo= "valore"
  Ecco un esempio, con una sintassi che serve a disporre un testo giustificato a destra: <P align="right">testo</P>

- In generale quindi la struttura di un TAG è:
  <TAG attributo_1="valore1" attributo_2="valore2"> contenuto</TAG>

- Alcuni particolari TAG non hanno contenuto - perché ad esempio indicano la posizione di alcuni elementi, come il TAG delle immagini - conseguentemente questi TAG non hanno neanche chiusura. La loro forma sarà dunque: <TAG attributi>. Questo tipo di TAG viene detto "empty", cioè "vuoto".

Un esempio di TAG empty per una immagine è:

<IMG width="20" height="20" SRC="miaImmagine.gif" ALT="alt">

- I TAG possono essere annidati l'uno dentro l'altro:

```
<TAG1 attributi>
        contenuto 1
        <TAG2>
              contenuto 2
        </TAG2>
 </TAG1>
```

- Ad esempio, per attribuire formattazioni successive al testo che stiamo inserendo:

```
 <P align="right">
          testo 1
        <P align="left">
          testo 2
        </P>
  </P>
```

- L’HTML è "case unsensitive", è del tutto indifferente se scrivere i TAG in maiuscolo o in minuscolo: <P ALIGN="RIGHT"> e <p align="right"> vengono letti allo stesso modo dal browser.

- XHTML è invece "case sensitive“

Commenti

- Per rendere il codice più leggibile è opportuno inserire "commenti" nei punti più significativi: in modo da mantenere l'orientamento anche in file molto complessi e lunghi. La sintassi è la seguente:

  <!-- questo è un commento -->

Struttura di un documento

- Un documento HTML è normalmente diviso in due sezioni:
  - Testa (<head>)

    Contiene informazioni che riguardano il modo in cui il documento deve essere letto e interpretato. Questo è il luogo dove scrivere - ad esempio - i meta-tag (p.e. per i motori di ricerca), script JavaScript o VbScript, fogli di stile, …
  - Corpo (<body>)

    Contiene il contenuto del documento

Esempio:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT">
<html>
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=iso 8859-1">
   <title>HTML.it</title>
</head>
<body>
     <!– qui il contenuto del documento -->
     Qui il nostro contenuto
</body>
</html>
```

- Il TAG: <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">

  indica al browser che deve caricare il set di caratteri occidentale

- Il TAG: <title>Nome del sito</title> indica il titolo della pagina che comparirà in alto sulla barra del browser

- Per impostare un colore di sfondo è necessario impostare il relativo attributo del tag body. Così:
  <body bgcolor="blue">

- Per inserire un'immagine come sfondo:
  <body background="imgSfondo.gif">
  L'immagine di sfondo verrà ripetuta in orizzontale e in verticale.

- È anche possibile combinare sullo sfondo il colore con l'immagine:
  <body bgcolor="#0000ff" background="imgSfondo.gif">

- Per scegliere il colore è comunque più opportuno utilizzare la corrispondente codifica esadecimale:

  <body bgcolor="#0000FF">

- Tabella colori e codici:

Colore	parola chiave	notazione esadecimale
Arancione	orange	#FFA500
Blu	blue	#0000FF
Bianco	white	#FFFFFF
Giallo	yellow	#FFFF00
Grigio	gray	#808080
Marrone	brown	#A52A2A
Nero	black	#000000
Rosso	red	#FF0000
Verde	green	#008000
Viola	violet	#EE82EE

- I browser lasciano un po’ di margine tra la pagina e il bordo della finestra. Per eliminare il bordo è sufficiente inserire i seguenti attributi del body:

  <body leftmargin="0" topmargin="0">

- Tramite l’attributo "lang" è possibile specificare ai motori di ricerca e al browser dell’utente quale lingua è utilizzata, es:

  <body lang="it">

Esempio:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//IT">
<html>
<head>
        <meta http-equiv="Content-Type" content="text/html;           charset=iso-
8859-1">
        <title>HTML.it</title>
</head>
<body leftmargin="0" topmargin="0" background="imgs/sfondo00006.gif"
bgcolor="#66CCFF" lang="it">
        Testo di prova
</body>
</html>
```

Intestazioni e paragrafi

- In HTML esistono sei livelli di intestazioni.

  H1 è il più importante, H2 è leggermente meno importante, e così via fino ad H6, il meno importante di tutti.

- Ecco come aggiungere un'intestazione importante:

```
<h1>Un'intestazione importante</h1>
```

- ed ecco un'intestazione leggermente meno importante:

```
<h2>Un'intestazione leggermente meno importante</h2>
```

Un'intestazione importante

Un'intestazione leggermente meno importante

- Ogni paragrafo dovrebbe cominciare con il marcatore e chiudere con .

```
<p>Questo è il primo paragrafo.</p>
<p>Questo è il secondo paragrafo.</p>
```

- E' possibile anche creare interruzioni di riga con il tag

```
<p>Questo è il terzo paragrafo.<br>Adesso siamo a capo di una riga </p>
```

Un'intestazione importante

Un'intestazione leggermente meno importante

Questo è il primo paragrafo.

Questo è il secondo paragrafo.

Questo è il terzo paragrafo.
Adesso siamo a capo di una riga

Enfasi al testo

- Si enfatizzano una o più parole grazie ad appositi marcatori.

   o <b> rende il testo grassetto

   o <i> rende il testo corsivo

  Esempio:

```
<p>Con il testo in grassetto, <strong>voglio
attirare l'attenzione</strong> mentre di seguito
faccio una citazione <em>”Thinking HTML”</em></p>
```

Un'intestazione importante

Un'intestazione leggermente meno importante

Questo è il primo paragrafo.

Questo è il secondo paragrafo.

Questo è il terzo paragrafo.
Adesso siamo a capo di una riga

Con il testo in grassetto, **voglio attirare l'attenzione** mentre di seguito faccio una citazione *”Thinking HTML”*

Collegamenti ad altre pagine

- Ciò che rende il Web efficace è la capacità di definire collegamenti da una pagina ad un'altra, sia all'interno dello stesso sito che verso altri siti. I collegamenti (o link) sono definiti per mezzo del marcatore <a>.

  Esempio di un collegamento alla pagina "libro.html":

```
<p>Questo è un collegamento <a
href="libro.html">alla pagina del libro</a></p>
```

- Per fare un collegamento ad una pagina su un altro sito Web è necessario fornire l'indirizzo Internet completo (comunemente chiamato URL), per esempio per creare un collegamento www.w3c.org:

```
<p>Questo è un link al
<a href="http://www.w3.org/">W3C</a></p>
```

Vedi la pagina HTML

Liste

- L'HTML supporta tre tipi di liste:

  - Il primo tipo è l'elenco puntato, definito spesso *lista non ordinata*.

    Esso usa i marcatori <ul> e <li> Per esempio:

```
<ul>
  <li>il primo elemento della lista</li>

  <li>il secondo elemento della lista</li>

  <li>il terzo elemento della lista</li>
</ul>
```

    Notate che è necessario chiudere sempre la lista con il marcatore finale .

- Il secondo tipo di lista è l'elenco numerato, detto anche *lista ordinata*. Usa i marcatori <ol> e <li>. Per esempio:

```
<ol>
  <li>il primo elemento della lista</li>

  <li>il secondo elemento della lista</li>

  <li>il terzo elemento della lista</li>
</ol>
```

Così come per gli elenchi puntati, occorre sempre chiudere la lista con il marcatore finale ,

- Il terzo ed ultimo tipo di lista è l'elenco di definizioni, che vi consente di elencare dei termini e le relative definizioni.

  Questo tipo di lista è aperta con il marcatore <dl> ed è chiusa con </dl>.

  Ogni termine comincia con un marcatore <dt> ed ogni definizione con un <dd>. Per esempio:

```
<dl> <dt>il primo termine</dt> <dd>la sua definizione</dd>
<dt>il secondo termine</dt> <dd>la sua definizione</dd>
<dt>il terzo termine</dt> <dd>la sua definizione</dd> </dl>
```

Tabelle

- Le tabelle sono utilizzate sia per fornire informazioni sia a fini di presentazione. Si possono allargare le tabelle fino a farle coincidere con i margini della pagina, specificare una larghezza fissa o lasciare che sia il browser a dimensionare la tabella per adattarla ai contenuti.

  Esempio:

```
<table border="1">
<tr><th>Anno</th><th>Vendite</th></tr>
<tr><td>2000</td><td>$18M</td></tr>
<tr><td>2001</td><td>$25M</td></tr>
<tr><td>2002</td><td>$36M</td></tr>
</table>
```

- Per tutte le celle della tabella è possibile incrementare la spaziatura interna [*distanza tra il margine esterno della cella e l'inizio del contenuto*] usando l'attributo *cellpadding* dell'elemento table. Per esempio, per impostare la spaziatura interna a 10 pixel:

```
<table border="1" cellpadding="10">
```

- Per contro, l'attributo *cellspacing* determina lo spazio tra le celle. Ecco come impostare la spaziatura tra le celle a 10:

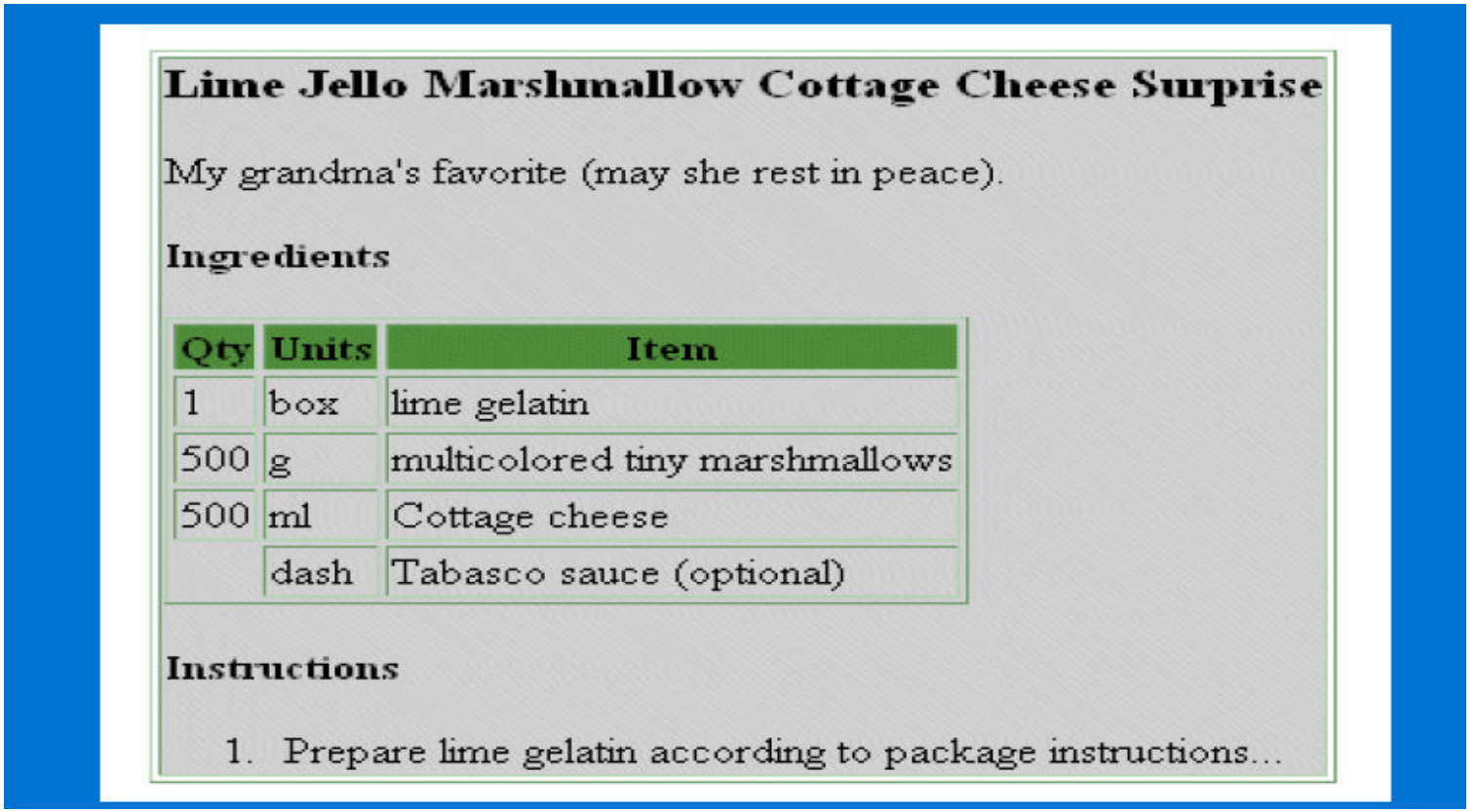
```
<table border="1" cellpadding="10" cellspacing="10">
```

Vedi la pagina HTML

- Per impostazione predefinita i browser centrano le intestazioni delle celle (th) ed allineano a sinistra i dati (td). E' possibile modificare l'allineamento usando l'attributo *align*, che può essere aggiunto ad ogni cella o alla riga (elemento tr). E' usato con i valori "left", "center" o "right":

Vedi la pagina HTML

- Esempio:



Lime Jello Marshmallow Cottage Cheese Suprise

My grandma's favorite (may she rest in peace).

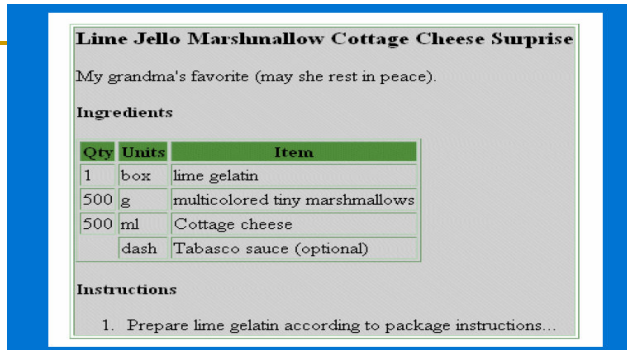
Ingredients

Qty	Units	Item
1	box	lime gelatin
500	g	multicolored tiny marshmallows
500	ml	Cottage cheese
	dash	Tabasco sauce (optional)

Instructions

1. Prepare lime gelatin according to package instructions...

HTML - esempio di pagina




```
<HTML>
<HEAD>
    <TITLE>Lime Jello Marshmallow Cottage Cheese Surprise</TITLE>
</HEAD>
<BODY>
    <H3>Lime Jello Marshmallow Cottage Cheese Surprise</H3>
        My grandma's favorite (may she rest in peace).
    <H4>Ingredients</H4>
            <TABLE BORDER="1">
    <TRBGCOLOR="#308030"><TH>Qty</TH><TH>Units</TH><TH>Item</TH></TR>
                <TR><TD>1</TD><TD>box</TD><TD>lime gelatin</TD></TR>
                <TR><TD>500</TD><TD>g</TD><TD>multicolored tiny marshmallows</TD></TR>
                <TR><TD>500</TD><TD>ml</TD><TD>cottage cheese</TD></TR>
                <TR><TD></TD><TD>dash</TD><TD>Tabasco sauce (optional)</TD></TR>
            </TABLE>
            <P>
                <H4>Instructions</H4>
                <OL>
                <LI>Prepare lime gelatin according to package instructions...</LI>
                <!-- and so on -->
</BODY>
</HTML>
```

- Il significato degli elementi della pagina non è reso dall'HTML: gli elementi della prima colonna NON sono quantità, sono solo testo libero

Immagini

- Le immagini possono essere usate per veicolare il messaggio che si intende comunicare. Il modo più semplice di aggiungere un'immagine è usare il marcatore <img>. Avremo quindi:

Poniamo che vi sia un file immagine chiamato ”libro.jpg" nella stessa directory del vostro file HTML. Esso misura 200 pixel di larghezza per 150 di altezza.

```
<img src=”libro.jpg" width="200" height="150”
alt=“Il Dizionario Sabatini-Colletti”>
```

L'attributo src richiama il file immagine.

L'attributo alt serve per le persone che non possono vedere l'immagine ed hanno bisogno di una descrizione da leggere in sua assenza.

Contenuti multimediali

- Dobbiamo prima di tutto distinguere tra due tipi di contenuto multimediale: il contenuto “interno” ed il contenuto “esterno”.Il contenuto *interno* è quello che è fruibile direttamente attraverso il browser, mentre quello *esterno* richiede del software che non è parte integrante del browser.
  - Un tipico contenuto interno è ad esempio una semplice animazione come una gif animata, una applet Java o un controllo ActiveX.
  - Per contenuto esterno si può far riferimento ad un filmato mpeg o una sorgente RealAudio (che comportano l'apertura dell'apposito lettore)

- Per inserire un contenuto esterno l'operazione da effettuare è la scrittura di un link che punti al file multimediale che si desidera mandare in esecuzione. Ad esempio per inserire il filmato ”libro.mpeg" nella propria pagina è necessario scrivere :
  <A href=”filmato.mov"> Vedi il Filmato </A>

  - E' consigliabile inserire nel testo allegato ai contenuti multimediali anche qualche informazione riguardo alle dimensioni del file e ai tempi di caricamento. Es.: Tempo richiesto per il download. Si può anche collegare il filmato ad una immagine o frame significativo.

- L'inserimento di contenuti interni è invece leggermente più complesso e richiede l'uso di appositi tag. Conviene focalizzare l’attenzione sul tag <OBJECT> che è quello raccomandato dal W3C.

- Per inserire una animazione Macromedia Flash all’interno della pagina HTML, andrà usato il tag <OBJECT> corredato dagli appositi parametri. Ogni plug-in, secondo le specifiche del produttore, usa i propri parametri.

CSS (Fogli di stile) introduzione

- Prima dei Cascade Style Sheets (CSS) o "Fogli di stile a cascata", in HTML le opportunità di definire l'aspetto delle pagine web erano limitate a specifici TAG (ad esempio il tag FONT) e si basavano sull'uso di tabelle per la definizione del layout.

- I CSS sono lo strumento definito dal W3C per definire l'aspetto delle pagine web. Sono essenzialmente un'insieme di regole per istruire il browser su come debba essere presentata la pagina web

- Con i CSS per separare il contenuto (es la scritta "titolo 1"), dalla formattazione (es il colore rosso e il grassetto) il contenuto della pagina diventa:

  ```
  <p class="formattaTitoli"> titolo 1 </p>
  ```

  e la colorazione del testo è affidata alla classe "formattaTitoli", descritta in un'altra parte del documento, o anche in un file separato. Per cambiare l'aspetto delle pagine è sufficiente editare la classe "formattaTitoli".

CSS - TAG Style

- Le regole dei CSS possono essere specificate direttamente nel codice (X)HTML tramite il tag STYLE posto nell'HEAD del documento (X)HTML. Se ne dovrebbe limitare l'uso a casi particolari.

```
<head>
    [...]
    <style type="text/css">
      <!-- body { font:80% Verdana,Helvetica, sans-serif; }
      p { font: 1.00em/1.20em verdana, helvetica, sans-serif }
      [...] -->
    </style>
    [...]
</head>
```

CSS - TAG Link

- I fogli di stile possono essere salvati in file specifici, con estensione .css, per essere poi collegati al codice (X)HTML tramite il tag LINK

```
<head>
    [...]
    <link rel="stylesheet" type="text/css" media="screen"
    href="/css/foglio_di_stile.css" />
    [...]
</head>
```

- Con il TAG link si può specificare se la visualizzazione debba essere fatta a schermo piuttosto che per la stampa:

```
<link rel="stylesheet" type="text/css" media="screen" href="/css/view.css" />

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

- La sintassi delle regole di applicazione dello stile segue lo schema:
  selettore { proprieta':valore; proprieta': valore }

  Esempi:
  - titolo { font-family: Arial; font-size: 18pt; color: #0000FF; text-align: center;}
    Formatta l'elemento titolo
  - argomento titolo { font-family: Arial }
    Formatta l'elemento <titolo> che sta dentro l'elemento <argomento>

- E’ possibile andare a capo:

```
selettore {
  proprietà1: valore1;
  proprietà2: valore2;
  proprietà3: valore3
}
```

- Esempio: Il seguente codice definisce il colore e le dimensioni del testo per gli elementi h1 e h2:

```
h1 { font-size: 25px; color: #660000; }
h2 { font-size: 15px; color: #990000; }
```

Il selettore font-size serve a specificare la dimensione dei font, il selettore color serve a definire il colore del testo.

Proprietà	Valori	Descrizione
background-color	red,green, blue, #666633,..	indica il colore dello sfondo di un elemento
background-image	url	specifica l'immagine da visualizzare come sfondo di un elemento
border-style	none, dotted, double,..	specifica lo stile del bordo da applicare a un elemento
color	red,green, blue, #666633,...	specifica il colore di un elemento
display	block, inline, none	specifica le modalità di visualizzazione di un elemento
font-family	times, helvetica, arial	indica il tipo di carattere da utilizzare
font-size	small, large, 10pt, 12pt,..	indica le dimensioni dei caratteri
font-weight	normal, bold, 100, 200, ..	indica lo spessore dei caratteri
height, width	30px, 40px, 75%, ..	indicano l'ampiezza e la larghezza di un elemento
position	absolute, relative, static	indica il tipo di posizionamento da applicare
text-decoration	none, underline, blink, ..	indica alcune decorazioni da applicare al testo, come ad esempio la sottolineatura
top, left	30px, 40px, 50px	indicano le coordinate dell'angolo superiore sinistro di un elemento
z-index	1,2,3,..	specifica il posizionamento degli elementi nello spazio (sovrapposizione)

CSS - Selettori

- I selettori servono a specificare a quale elemento del documento la regola debba essere applicata. I principali selettori sono:
  - Tag HTML
  - Classi
  - Identificatori
  - Pseudo-classi e Pseudo-elementi
  - Selettori in cascata
  - Raggruppamento

Tag HTML

- Ogni tag HTML è un selettore. Ad esempio:

```
p { text-indent: 2em; }
a { font-weight: bold; }
```

- Il codice sopra riportato specifica al browser che i paragrafi devono avere il capoverso con un rientro di due caratteri e che i link devono essere in grassetto, migliorandone l'identificazione all'interno del testo.

Classi

- Spesso si ha la necessità di associare ad uno stesso tag HTML diversi stili, per fare ciò è possibile ricorrere alle classi.

```
/* carattere piccolo */
    p.nota { font-size: small }
/* area con sfondo giallo */
    p.evidenzia { background-color: yellow; }
```

  Le due classi sono *nota* ed *evidenzia* e vengono specificate separando il nome del tag e il nome della classe con un punto.

- La classe viene specificata nel codice HTML attraverso l'attributo CLASS:

```
<p class="nota">Questo paragrafo è una nota</p>
<p class="evidenziato">Questo paragrafo è evidenziato in giallo</p>
```

  La sintassi utilizzata specifica che la classe è associata al TAG P.

- In questo modo è possibile associare lo stesso nome a classi associate a TAG differenti:

  - CSS

```
p.nota { font-size: small } /* carattere piccolo */
span.nota { color: #999999; } /* testo in grigio */
```

  - (X)HTML

```
<p class="nota">Questo paragrafo è una nota</p>
<p>Testo normale (<span class="nota">nota nel testo</span>) testo
   normale ...</p>
```

- Le classi possono anche essere associate a nessun TAG HTML, in questo modo la stessa classe può essere associata a più selettori:
  - CSS

```
.nota { font-size: small } /* carattere piccolo */
```

  - (X)HTML

```
<p class="nota">Questo paragrafo è una nota</p>
<div class="nota">Questa sezione è una nota</div>
```

- A un selettore è possibile associare una e una sola classe. Ad esempio non è valido il seguente codice:

```
p.classe1.classe2 { ... }
```

Pseudo-classi

- Pseudo-classi e pseudo-elementi sono classi ed elementi speciali, automaticamente riconosciuti dai browser che supportano i CSS. Non è dunque necessario specificare nulla nel codice HTML.

- Le pseudo-classi creano distinzioni fra elementi, ad esempio i link visitati e i link attivi sono distinti attraverso due pseudo-classi.

- La sintassi è la seguente:

  selettore:pseudoClasse { proprietà: valore }

- Definizione comune dei link (LOVE / HATE)

```
a:link { color:  blue; }
a:visited { color: purple; }
a:hover { color: red; }
a:active { color: darkred; }
```

  Il codice CSS indica al browser che i link devono essere blu, i link visitati devono essere viola, i link sui quali il puntatore del mouse si trova devono essere rossi (effetto roll-over), i link attivi devono essere rosso scuro.

- Nel codice XHTML non è necessario specificare nulla:

```
<a href="URL">questo link cambierà colore in base al suo stato</a>
```

Proprietà e Valori

- Per la specifica dei valori delle proprietà viene utilizzata la seguente sintassi (la stessa utilizzata dal W3C):
  - <Abc>: valore della proprietà
  - A B C: keyword richieste nell'ordine specificato
  - A|B: deve apparire A o B
  - A||B: deve apparire A o B oppure ambedue
  - [Abc]: le parentesi quadre sono utilizzate per raggruppare gli oggetti
  - Abc*: Abc appare zero o più volte
  - Abc+: Abc appare una o più volte
  - Abc?: Abc è opzionale
  - Abc{A,B}: Abc deve apparire almeno A volte e al massimo B volte

font-family

- Sintassi
  font-family: <nome font>* || <font generico>
  - <nome font>
    - qualsiasi nome di font (per esempio: verdana, helvetica, georgia)
  - <font generico>
    - serif
    - sans-serif
    - monospace
    - cursive[sconsigliato poiché di difficile lettura]
    - fantasy[sconsigliato poiché di difficile lettura]
- Esempio:
  p { font-family: verdana, helvetica, "Trebuchet MS", sans-serif }
  Il font adottato sarà il primo disponibile partendo da sinistra. E' bene specificare sempre un font generico, applicato qualora nessuno dei font elencati fosse disponibile. Font con nomi composti da due o più parole vano fra virgolette: ad esempio "Trebuchet MS"

font-style

- Sintassi
  font-style: normal | italic | oblique;

- Esempio:
  quote { font-style: italic }

- Questa regola rende il testo italico (corsivo). Il valore oblique inclina in senso opposto. Il valore normal non introduce variazioni.

font-variant

- sintassi
  font-variant: normal | small-caps

- Esempio:
  .maiuscoletto { font-variant: small-caps }

  Si noti che .maiuscoletto equivale alla definizione della classe “maiuscoletto” nel CSS. In HTML faremo riferimento ad essa con class = “maiuscoletto”. Il valore small-caps crea l'effetto "Maiuscoletto", il valore normal non introduce variazioni.

font-weight

- sintassi
  font-weight: normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900

- Esempio:
  strong { font-weight: 900; }

- Specifica il peso del font. I valori bolder e lighter sono relativi al valore di default. Gli altri valori sono pesi assoluti. 100-900 è una scala che va dal peso minimo al massimo. Non tutti e nove i valori da 100 a 900 sono disponibili per tutti i font, in questo caso il peso adottato è il più vicino disponibile.

font-size

- sintassi
  font-size: <dimensione assoluta> | <dimensione relativa> | <lunghezza> | <percentuale>
  - <dimensione assoluta>
    xx-small | x-small | small | medium | large | x-large | xx-large
  - <dimensione relativa>
    larger | smaller
  - <lungezza> e <percentuale>
    vedi sezioni seguenti
- Esempio:
  p { font-size: 0.80em }

font

- sintassi
  font: <valore>

- <valore>
  [<font-style> || <font-variant> || <font-weight>]? <font-size>
  [/ <line-height>]? <font-family>

- Esempio:
  p.paragrafo-speciale { font: bold 1.00em/1.50em verdana,
  helvetica, "Trebuchet MS", sans-serif }

- La proprietà font può essere utilizzata come un metodo veloce per definire le diverse proprietà font-family, font-style, ecc. Si noti la possibilità di specificare anche la proprietà <line-height> (una proprietà per specificare l'interlinea). Quando si utilizza la proprietà font è necessario specificare almeno i valori per le proprietà font-size e font-family.

color

- sintassi
  color: <colore>

- Esempio:
  a { color: #009 }
  em {color: red }

- La proprietà color permette di specificare il colore del testo di un selettore

background

- La proprietà background permette di specificare il colore e l'immagine di sfondo. L'immagine può essere posizionata come si desidera, può essere ripetuta oppure no, può scrollare col testo oppure essere fissa. La sintassi è:
  background-color: <valore>

  - <valore>
    <colore> || <immagine> || <ripetizione> || <scrolling> || <posizione>

  - <colore>
    i possibili valori sono i soliti descritti per la proprietà color

  - <immagine>
    url dell'immagine

  - <ripetizione>: repeat | repeat-x | repeat-y | no-repeat
    l'immagine di sfondo può essere ripetuta, ripetuta orizzontalmente, verticalmente, non essere ripetuta (immagine di sfondo unica)

- <scrolling>: scroll | fixed
  l'immagine si muove col testo oppure rimane fissa.

- <posizione>: [<percentuale> | <lunghezza>]{1,2} | [top | center | bottom] || [left | center | right]
  la posizione dell'immagine di sfondo può essere specificata con riferimento all'angolo superiore sinistro, sia attraverso unità percentuali o di lunghezza sia attraverso keyword.

- Esempio:

  body { background:#FFFFFF url(/images/sfondo.gif) no-repeat fixed top right; }

word-spacing

- La proprietà word-spacing permette di impostare lo spazio fra le parole, aumentando oppure diminuendo tale spazio. La sintassi è:
  word-spacing: <valore>

- <valore>
  normal | <lunghezza>

- Esempio:
  - .parole-distanziate { word-spacing: 1.00em }
  - .parole-ravvicinate { word-spacing: -0.20em }

letter-spacing

- La proprietà letter-spacing permette di impostare lo spazio fra le singole lettere di una parola, aumentandolo oppure diminuendolo. La sintassi è:
  letter-spacing: <valore>

- <valore>
  normal | <lunghezza>

- Esempio:
  - .lettere-distanziate { letter-spacing: 0.50em }
  - .lettere-ravvicinate { letter-spacing: -0.15em }

- .

line-height

- La proprietà line-height permette di impostare l'interlinea. La sintassi è:
  line-height: <valore>

- <valore>
  normal | <lunghezza> | <percentuale>

- Esempio:
  .interlinea-ridotta { line-height: 100% }

- Le percentuali si riferiscono all'altezza dei font e non al valore standard dell'interlinea, valori negativi non sono permessi. L'interlinea può essere specificata anche tramite la proprietà font.

text-indentation

- La proprietà text-indentation permette di impostare il rientro del capoverso.
  La sintassi è:
  text-indentation: <valore>

- <valore>
  <lunghezza> | <percentuale>

- Esempio:
  p { text-indentation: 2em; }

  em è la dimensione tipografica di una M maiuscola.

text-decoration

- La proprietà text-decoration permette decorare il testo. La sintassi è:
  text-decoration: <valore>

- <valore>
  none | [underline || overline || line-through || blink]

- Esempio:
  #barra-di-navigazione A { text-decoration: none }
  /* link non sottolineato */

- Esempio:

Giacomo Leopardi
L'infinito
Sempre caro mi fu quest'ermo colle,
e questa siepe, che da tanta parte

….

Codice HTML

```
<html>
<head>
    <title>Old style</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body bgcolor="#FFFFFF" text="#000000">
            .....
    <h1><font face="Georgia, Times New Roman, Times, serif" color="#004080">
    Giacomo Leopardi</font></h1>
    <h2><font face="Georgia, Times New Roman, Times, serif“ color="#004080">
    L'infinito</font></h2>
    <p><font face="Verdana, Arial, Helvetica, sans-serif" size="2">Sempre caro mi fu
    quest'ermo colle,</font></p>
    <p><font face="Verdana, Arial, Helvetica, sans-serif" size="2">e questa siepe, che
    da tanta parte</font></p>

</body>
</html>
```

- In XHTML tutta la formattazione è definita in un CSS

Codice XHTML

```
<?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">
<head>
        <title>New Style</title>
        <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"
/>
        <link rel="stylesheet" href="newstyle.css" type="text/css" />
</head>
<body>
        <h1>Giacomo Leopardi</h1>
        <h2>L&#39;infinito</h2>
        <p>Sempre caro mi fu quest&#39;ermo colle,</p>
        <p>e questa siepe, che da tanta parte</p>
……..
</body>
</html>
```

- Casi di esempio in aula HTML con CSS

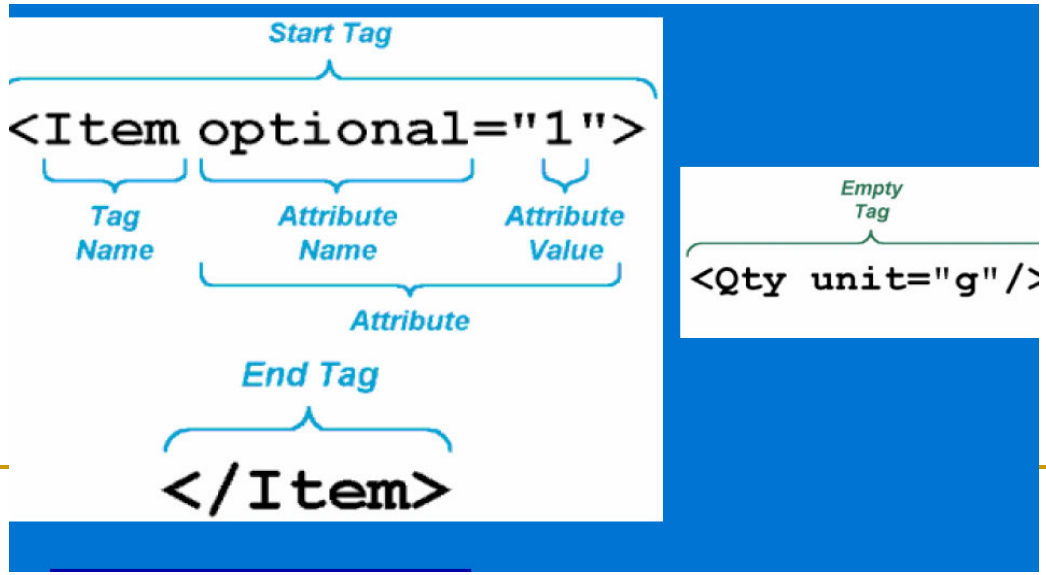
XML

TAG

- In XML I documenti devono essere “well-formed”: i TAG devono essere aperti e chiusi. La notazione per i TAG è la seguente:

  < tag1 >Testo del tag < /tag1 >

- Un TAG XML può contenere al suo interno un attributo, il valore dell’attributo deve essere obbligatoriamente racchiuso tra virgolette doppie: < div id=“header” > < /div >. In HTML possono anche essere “nudi”: <TABLE BORDER=1>



- In XML i TAG non possono sovrapporsi:

  <primotag> aaaaa <secondotag> bbbbbb
  </primotag> </secondotag>

  la forma accettabile è:

  <primotag> aaaaa <secondotag> bbbbbb
  </secondotag> </primotag>

Tipi di dato e DTD

- I tipi di dato che vengono usati in un documento XML sono definiti in un DTD: document type definition
- Il DTD contiene l'insieme di tag, attributi ed entities che possono essere usati all'interno di un documento. Più in generale, il DTD definisce la grammatica del linguaggio di markup (del file XML) definita dal disegnatore del linguaggio, comprensiva di:
  - nomi dei tag;
  - restrizioni rispetto alla nidificazione dei tag;
  - definizione di attributi dei tag.
- I DTD danno l'estendibilità dell'XML. Usano una sintassi diversa dai documenti XML. Un parser XML che valida un documento legge il documento ed il DTD, e controlla la corrispondenza nei confronti del DTD.

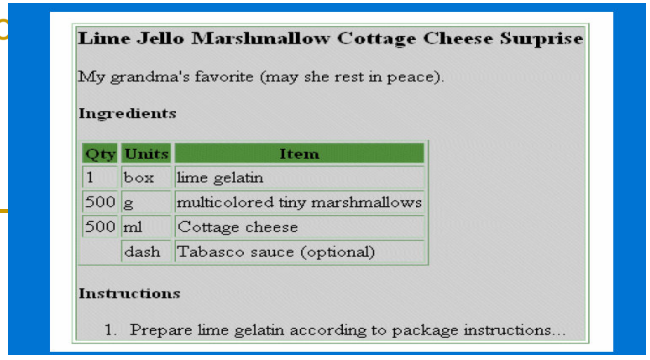
- Un DTD può essere definito esternamente al documento. In tal caso si deve inserire standalone="no" nello header.

- Si può anche definire un DTD interno al documento con:

  <!DOCTYPE *root_element* [Document Type Definition (DTD): elements/attributes/entities/notations/processing instructions/comments/PE references]>

- Riprendiamo l'esempio di pagina HTML già trattato:

```
<HTML>
<HEAD>
    <TITLE>Lime Jello Marshmallow Cottage Cheese Surprise</TITLE>
</HEAD>
<BODY>
    <H3>Lime Jello Marshmallow Cottage Cheese Surprise</H3>
        My grandma's favorite (may she rest in peace).
    <H4>Ingredients</H4>
        <TABLE BORDER="1">
            <TRBGCOLOR="#308030"><TH>Qty</TH><TH>Units</TH><TH>Item</TH></TR>
            <TR><TD>1</TD><TD>box</TD><TD>lime gelatin</TD></TR>
            <TR><TD>500</TD><TD>g</TD><TD>multicolored tiny marshmallows</TD></TR>
            <TR><TD>500</TD><TD>ml</TD><TD>cottage cheese</TD></TR>
            <TR><TD></TD><TD>dash</TD><TD>Tabasco sauce (optional)</TD></TR>
        </TABLE>
        <P>
            <H4>Instructions</H4>
            <OL>
            <LI>Prepare lime gelatin according to package instructio
            <!-- and so on -->
</BODY>
</HTML>
```



Lime Jello Marshmallow Cottage Cheese Suprise

My grandma's favorite (may she rest in peace).

Ingredients

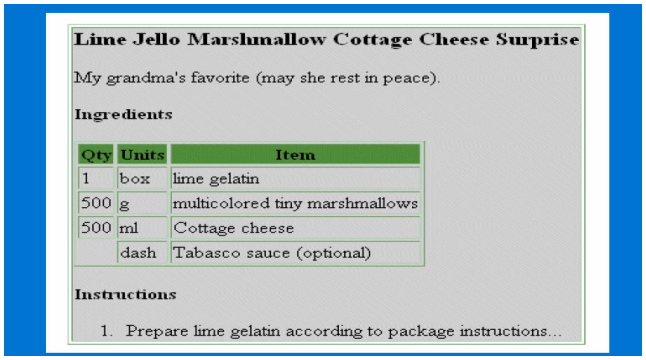
Qty	Units	Item
1	box	lime gelatin
500	g	multicolored tiny marshmallows
500	ml	Cottage cheese
	dash	Tabasco sauce (optional)

Instructions

1. Prepare lime gelatin according to package instructions...

- Lo stesso esempio in XML può essere scritto:

```
<?xml version="1.0" ?>
<Recipe>
    <Name>Lime Jello Marshmallow Cottage Cheese Surprise</Name>
    <Description>My grandma's favorite (may she rest in peace).</Description>
    <Ingredients>
        <Ingredient>
            <Qty unit="box">1</Qty>
            <Item>lime gelatin</Item>
        </Ingredient>
        <Ingredient>
            <Qty unit="g">500</Qty>
            <Item>multicolored tiny marshmallows</Item>
        </Ingredient>
        <Ingredient>
            <Qty unit="ml">500</Qty>
            <Item>Cottage cheese</Item>
        </Ingredient>
        <Ingredient>
            <Qty unit="dash"/>
            <Item optional="1">Tabasco sauce</Item>
        </Ingredient>
    </Ingredients>
    <Instructions>
        <Step>Prepare lime gelatin according to package instructions</Step>
        <!-- And so on... -->
    </Instructions>
</Recipe>
```



Lime Jello Marshmallow Cottage Cheese Surprise

My grandma's favorite (may she rest in peace).

Ingredients

Qty	Units	Item
1	box	lime gelatin
500	g	multicolored tiny marshmallows
500	ml	Cottage cheese
	dash	Tabasco sauce (optional)

Instructions

1. Prepare lime gelatin according to package instructions...

- Il DTD associato specifica quali elementi esistono, con quali attributi, con quali relazioni reciproche ed in quale ordine si trovano:

```
<!-- DTD per le ricette -->
<!ELEMENT Recipe (Name, Description?, Ingredients?,Instructions?)>
<!ELEMENT Name (#PCDATA)>
<!ELEMENT Description (#PCDATA)>
<!ELEMENT Ingredients (Ingredient)*>
<!ELEMENT Ingredient (Qty, Item)>
<!ELEMENT Qty (#PCDATA)>
<!ATTLIST Qty unit CDATA #REQUIRED>
<!ELEMENT Item (#PCDATA)>
<!ATTLIST Item optional CDATA "0“ isVegetarian CDATA "true">
<!ELEMENT Instructions (Step)+>
```

- Si noti che nell'esempio è descritta la struttura dei dati ma non è descritto come deve essere mostrata la ricetta:

```
<?xml version="1.0"?>
Header
<Ingredient>
    <Qty unit="box">1</Qty>
    <Item>lime gelatin</Item>
</Ingredient>
```

- <!ELEMENT Recipe (Name, Description?, Ingredients?, Instructions?)> definisce un tag di nome Recipe che contiene gli elementi tra parentesi.
  - “?” significa che l’elemento è opzionale e può apparire 1 o 0 volte
  - “+” significa 1 o più
  - “*” significa 1,0 o più volte

- <!ELEMENT Name (#PCDATA)> indica che l’elemento può contenere solo caratteri e nient’altro (parsed character data). Non può avere figli.

- <!ATTLIST Item optional CDATA "0“ isVegetarian CDATA "true"> definisce un tag con due possibili attributi con relativo valore di default.
  - Un attributo può essere reso obbligatorio con #REQUIRED.
  - Se un attributo non è richiesto allora si può indicare con #IMPLIED

- Es: DTD: <!ELEMENT orario (#PCDATA)>
  XML: <orario> 24 Maggio 2000 20:30</orario>
  o in alternativa:
  DTD: <!ELEMENT orario (EMPTY)>
  <!ATTLIST orario giorno CDATA #REQUIRED mese CDATA #REQUIRED anno CDATA #REQUIRED
  …..

- Un attributo può assumere solo certi valori predefiniti:
  <!ATTLIST corso_laurea (Elettronica | Informatica | Meccanica | Civile | Ambientale) “Informatica”>

- Si possono definire entità (standard: < > ") aggiuntive:
  <!ENTITY DSI “Dip. Sistemi e Informatica”>&DSI
  verrà sostituito dalla stringa associata

- I tipi di attributi possono essere:
  - Stringhe (CDATA o character data): possono assumere qualsiasi valore tranne: < > & ‘ “. Per rappresentare questi simboli si devono usare ENTITY
  - Token (ID, IDREF, ENTITY, NMTOKEN): impongono dei limiti ai valori degli attributi: tutti gli ID devono essere diversi; IDREF deve riferirsi ad un ID esistente. Un parser controlla la corrispondenza
  - Attributi enumerati

- Dentro <!ELEMENT…> si possono creare espressioni complesse: se si definisce <!ELEMENT x (a,(b | c | d)*, e)*> allora sono validi tutti i seguenti elementi:

  <x> <a /> <b /> <e /></x>

  <x></x>

  <x> <a /> <e /> <a /><c /><d /><e /></x>

- Un TAG può essere vuoto se per esempio al posto di #PCDATA si usa EMPTY. Può contenere sia testo che altri TAG o qualsiasi altra cosa se è definito come ANY. Il DTD completo del caso precedente potrebbe essere:

  <!ELEMENT x (a,(b | c | d)*, e)*>
  <!ELEMENT a EMPTY>
  <!ELEMENT b EMPTY>
  <!ELEMENT c EMPTY>
  <!ELEMENT d EMPTY>
  <!ELEMENT e EMPTY>

- Il “|” indica una scelta. Può essere fatta tra altri ELEMENT o anche dati #PCDATA:

```
<?xml version="1.0"?>

<!DOCTYPE student [<!ELEMENT student (#PCDATA | id )*>
<!ELEMENT id (#PCDATA)>]>

<student>
Ecco del testo
<id>9216735</id>
Il testo puo’ essere ovunque prima e dopo l’elemento.
Puo’ anche non esserci l’ id.
</student>
```

- Esempio

video.dtd

```
<!-- DTD per video -->
<!ELEMENT video (Titolo, Riassunto?,Stream?)>
<!ELEMENT Titolo (#PCDATA)>
<!ELEMENT Riassunto (#PCDATA)>
<!ELEMENT Stream (Episodio)*>
<!ELEMENT Episodio (Shot*,Avvenimento)>
<!ELEMENT Avvenimento (#PCDATA)>
<!ELEMENT Shot (Inizio,Fine,Keyframe,Audio?,Mosaic?)+>
<!ELEMENT Inizio (#PCDATA)>
<!ELEMENT Fine (#PCDATA)>
<!ELEMENT Keyframe (#PCDATA)>
<!ELEMENT Audio (#PCDATA)>
<!ELEMENT Mosaic (#PCDATA)>
```

prova.xml

```
<!DOCTYPE video SYSTEM "video.dtd">

<video>
<Titolo>Ronin</Titolo>
<Riassunto>Film d'azione con Robert De Niro</Riassunto>
<Stream>
<Episodio>
<Shot>
<Inizio>1</Inizio>
<Fine>2345</Fine>
<Keyframe>http://www.dsi.unifi.it/keyframe/ronin1.jpg
</Keyframe>
</Shot>
<Avvenimento>
Vengono presentati i personaggi
</Avvenimento>
</Episodio>
</Stream>
</video>
```

XML Schema

- I DTD non sono documenti XML quindi non sono manipolabili in modo automatico. Hanno limitazioni, es:

  <altezza>1.85</altezza>

  <altezza>biondo</altezza>

  sono entrambi validi se definiti con

  <!ELEMENT altezza (#PCDATA)>

- Uno Schema XML è un documento XML conforme ad un DTD che definisce la struttura di uno schema… questi DTD sono associati al parser…Con gli Schema XML diventa possibile differenziare:

  <altezza>1.85</altezza>

  <altezza>biondo</altezza>

  il primo potrebbe essere valido, il secondo no

- Esempio di schema W3C:

```
<xsd:element name="bookID" type="catalogID" />
<xsd:simpleType name="catalogID" base="xsd:string">
<xsd:pattern value="\ d{3} - \ d {4} - d {3} "/>
</xsd:simpleType>
```

  Nell'esempio precedente viene definito un TAG bookID di tipo catalogID. catalogID viene definito come tipo semplice (ovvero senza sotto-elementi) deve seguire il pattern: *3 cifre, trattino, 4 cifre, trattino, 3 cifre*. I pattern seguono le regole del Perl

- Gli Schema di Microsoft normalmente sono contenuti in file .xml
  W3C invece usa .xsd

- Tools che convertono DTD in Schema:
  http://www.w3.org/XML/Schema.html

  In Internet Explorer >= 5.x c'è un parser XML che gestisce MS Schema. Xerces valida documenti W3C Schema. Si può usare anche XMLSpy

CSS e XSL

- L’informazione su come rappresentare i dati è memorizzata da un’altra parte ed è scritta con un linguaggio di stile: CSS (Cascading Style Sheets) e e XSL (eXtensible Stylesheets Language).
- I CSS sono gli stessi già usati per l’HTML. XSL è utilizzabile solo per XML (MS Internet Explorer >= 5.0 e Mozilla).
- Il CSS puo' essere dichiarato internamente o esternamente in un documento XML. Di solito si conserva in un file esterno e si include con l'istruzione:

  <?xml-stylesheet type=”text/css” href=”stile.css”?>

CSS

- I CSS come definiti per HTML si possono applicare direttamente ai documenti XML. Basta ridefinire i TAG XML con la rappresentazione relativa. La sintassi delle regole di applicazione dello stile segue lo schema:

  selettore { proprieta':valore; proprieta': valore }

  Esempi:

  - titolo { font-family: Arial; font-size: 18pt; color: #0000FF; text-align: center;}

    Formatta l'elemento titolo
  - argomento titolo { font-family: Arial }

    Formatta l'elemento <titolo> che sta dentro l'elemento <argomento>

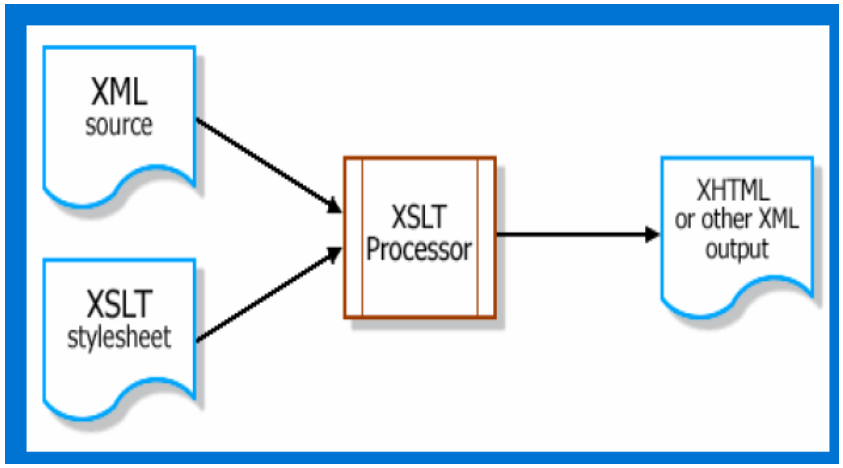
    immagine[file=”im1.jpg”] {font-family: Arial }

    Formatta l'elemento <immagine> con attributo [file=”im1.jpg”]

XSL

- La notazione dello XSL è l'XML. In pratica è un documento XML che dice come trasformare un altro documento XML. Usando XSL diversi si ottengono più rappresentazioni dallo stesso XML

- Un file XSL è composto da una serie di regole (*template*) che sono applicate ad un file XML da un parser XSL. XSL deve ancora essere approvato (XSL 1.0 candidate recommendation). La parte di trasformazione (XSLT) è già stata approvata. XSL comprende XSLT (linguaggio per effettuare la trasformazione di documenti XML)



XHTML

Concetti base del linguaggio XHTML

- XHTML è la riformulazione di HTML come applicazione XML. Ciò significa essenzialmente una cosa: un documento XHTML deve essere valido e ben formato.

- Una pagina XHTML può essere sottoposta ad un validatore. Il validatore controlla:
  - che il codice sia ben formato, ovvero rispetti la corretta grammatica e sequenza di tag
  - che il codice sia valido, ovvero che sia conforme ai tipi di dato definiti nel DTD. Il W3C definisce quale tipo di dato deve essere definito all'interno di un documento XHTML 1.1

    http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd

- Con l'introduzione di XHTML, piuttosto che sfornare una nuova versione del linguaggio, un HTML 5.0, il W3C ha compiuto un'opera di ridefinizione:
  - Niente nuovi tag, attributi o metodi. Questi rimangono essenzialmente quelli di HTML 4.0.
  - Il vocabolario rimane uguale, ma cambiano le regole sintattiche. Se vengono inseriti elementi non supportati (font, larghezza per le celle di tabelle o margini per il body, per citare solo alcuni esempi) il documento non è valido.
  - Con XHTML, almeno nella sua versione Strict, si torna ad un linguaggio che definisce solo la struttura. La formattazione si fa con i CSS..

- XHTML risponde a due esigenze fondamentali:
  - portare HTML nella famiglia XML con i benefici che ciò comporta in termini di estensibilità e rigore sintattico
  - mantenere la compatibilità con i software che supportano HTML 4.0

- HTML era “figlio” del linguaggio di markup SGML: Standard Generalized Markup Language
- XML è un sottoinsieme di SGML caratterizzato dal fatto di avere restrizioni sulla grammatica dei tag
- XHTML è “figlio” di XML, segue pertanto la sua filosofia di correttezza e validità. In XHTML il contenuto deve essere rigorosamente separato dalla presentazione. Si usano TAG che hanno un contenuto semantico: h1 ad esempio rappresenta un header, non come deve essere visualizzato

Versioni di XHTML

- Le versioni di XHTML disponibili e raccomandate dal W3C sono tre.

- **XHTML 1.0 (**pub. 2000, rev. 2001).
  - Consiste nella riscrittura in XML di HTML 4.0 e si basa sulle tre DTD già definite per questo linguaggio:
    - DTD Strict
    - DTD Transitional
    - DTD Frameset

- **XHTML Basic**
  - E' una versione "ridotta" del linguaggio, specificamente pensata per dispositivi mobili (PDA, cellulari), contiene solo gli elementi che si adattano a questi dispositivi (esclude, ad esempio, i frames che non hanno ovviamente senso in tale contesto). (sostituisce WML come linguaggio per le applicazioni WAP.

- **XHTML 1.1**
  - Basata sulla DTD Strict della versione 1.0, rappresenta la prima formulazione pratica del concetto di modularità. L'insieme di tag che definiscono la struttura di un documento sono raggruppati in una serie di moduli indipendenti, che possono essere implementati o esclusi secondo le necessità. E' la base della futura estensione di XHTML con altri moduli personalizzati.

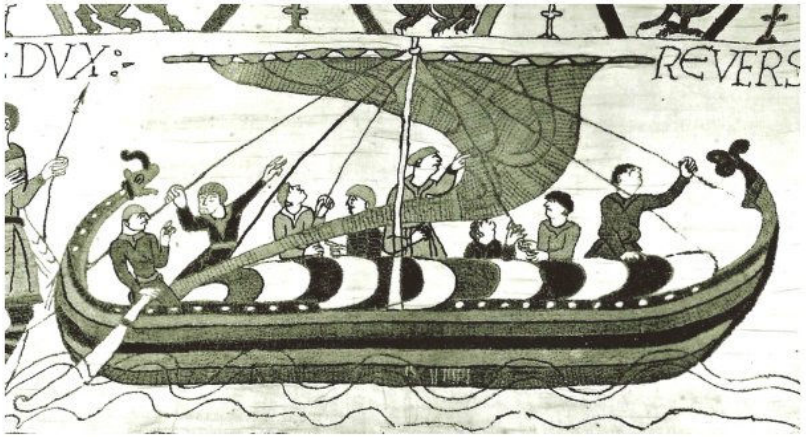
- In XHTML in un foglio di stile possono essere definiti elementi diversi
  - un tag HTML (come BODY, P, DIV)
  - una classe
  - un identificatore
- In XHTML la pagina generica è strutturata attraverso il TAG div che definisce i vari box che costituiscono il layout della pagina stessa
- Le proprietà di visualizzazione sono descritte attraverso l'uso dei CSS (CSS2)

Esempio

```
<div id="content">
  <div id="head"><h1>Thomas M Alisi - Intervallo</h1></div>
  <div id="text">
    <p>Le attivit&agrave; riprenderanno appena possibile. [<a href="old.php">versione precedente</a>] </p>
    <p>New site coming soon. [<a href="old.php">old version</a>]  </p>
  </div>
  <div id="footer">&copy; <a href="mailto:alisi@micc.unifi.it">TMA</a> '06<br />
  last edit: '06.1.3</div>
</div>
```

```
#content {
        width: 600px;
        margin: 25px auto;
        text-align: left;
}
#head {
        border-bottom: 1px solid #333333;
}
#text {
        background: url(../img/bt.jpg) no-repeat top left;
        margin-top: 10px;
        padding-top: 450px;
}
#footer {
        border-top: 1px solid #333333;
        margin-top: 25px;
        padding-top: 25px;
        font-size: 80%;
}
```

Thomas M Alisi - Intervallo



Detail from the Bayeux Tapestry
© Reproduced with kind permission of the Victoria & Albert Museum, London

Le attività riprenderanno appena possibile. [versione precedente]
New site coming soon. [old version]

© TMA '06
last edit: '06.1.3

CSS2 – Selettori

- I selettori servono a specificare a quale elemento del documento la regola debba essere applicata. I principali selettori sono:
  - Tag HTML
  - Classi
  - Identificatori
  - Pseudo-classi e Pseudo-elementi
  - Selettori in cascata
  - Raggruppamento

Identificatori

- Gli identificatori sono simili alle classi, ma servono a definire un'entità unica all'interno del documento. Se le classi, come la parola stessa indica, definiscono un insieme di elementi concettualmente omogenei (ad esempio le note di un testo possono essere accorpate nella classe note), l'identificatore deve definire un elemento che non si ripete all'interno del documento, ad esempio la nota per il copyright.

- Il nome del identificatore deve essere preceduto dal carattere '#'.

```
#nota-copyright { font-size: xx-small }
```

- L'identificatore viene specificato nel codice HTML attraverso l'attributo ID:

```
<p id="nota-copyright">Questo paragrafo è la nota per il copyright</p>
```

- Nel codice non devono esistere due tag HTML associati allo stesso identificatore, non possono dunque esistere due tag HTML con lo stesso valore per l'attributo ID.

```
<p id="nota-copyright">Questo paragrafo è la nota per il copyright</p>
<p id="nota-copyright">E' errato inserire questo paragrafo</p>
```

Selettori in cascata

- Ci sono numerosi altri modi più o meno elaborati per specificare dei selettori, alcuni dei quali non completamente supportati da browser largamente diffusi (ad esempio Internet Explorer 6 e inferiori).

- Un metodo ben supportato e molto utile e quello di utilizzare selettori in cascata:

  selettore1 selettore2 { proprietà: valore }

- In questo caso la regola si applica al selettore2 se questo è contenuto nel selettore1. Si consideri il seguente codice CSS:

  ```
  code i { color: red; }
  ```

  Questo codice specifica che il tag I, se inserito nel TAG CODE, deve contenere del testo di colore rosso.

- Consideriamo il seguente codice HTML:

  ```
  <p><code>testo normale <i>testo corsivo</i> testo normale</code></p>
  <p>testo normale <i>testo corsivo</i> testo normale</p>
  ```

testo normale *testo corsivo* testo normale

testo normale *testo corsivo* testo normale

Raggruppamento

- Qualora più selettori dovessero rispettare le stesse regole, è possibile raggrupparli

```
h1, h2, h3, h4, h5, h6 { font-family: sans-serif }
```

- Il codice è equivalente al seguente:

```
h1 { font-family: sans-serif }
h2 { font-family: sans-serif }
h3 { font-family: sans-serif }
h4 { font-family: sans-serif }
h5 { font-family: sans-serif }
h6 { font-family: sans-serif }
```

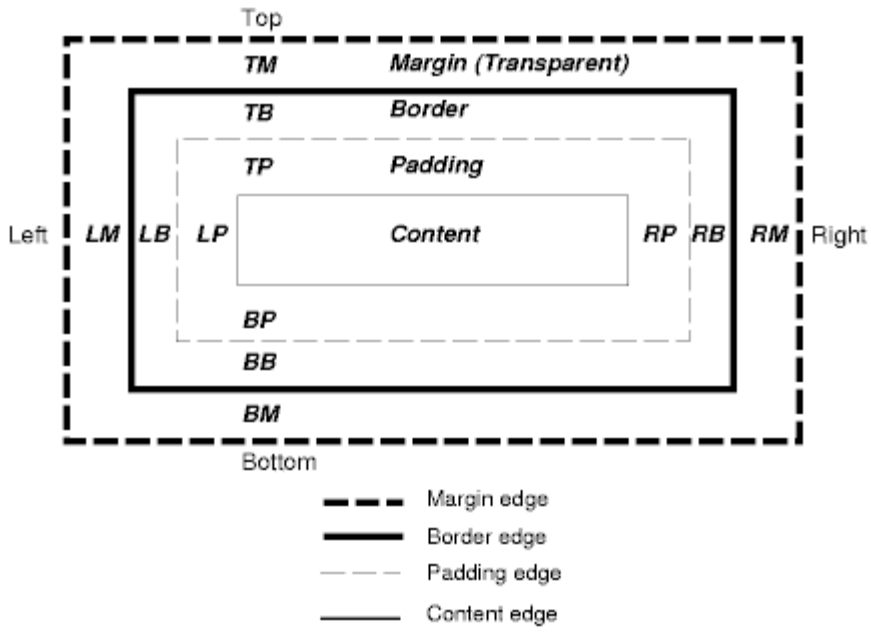
CSS2 Box Model

- The CSS2 box model describes the rectangular boxes that are generated for elements in the document tree and laid out according to the visual formatting model.

- In the visual formatting model, each element in the document tree generates zero or more boxes according to the box model.

- The layout of these boxes is governed by:
  - * box dimensions and type.
  - * positioning scheme (normal flow, float, and absolute).
  - * relationships between elements in the document tree.
  - * external information (e.g., viewport size, intrinsic dimensions of images, etc.).

- Each box has a content area (e.g., text, an image, etc.) and optional surrounding padding, border, and margin areas; the size of each area is specified by properties defined below.



- The perimeter of each of the four areas (content, padding, border, and margin) is called an "edge", so each box has four edges:
  - content edge or inner edge (the content edge surrounds the element's rendered content)
  - padding edge (the padding edge surrounds the box padding). If the padding has 0 width, the padding edge is the same as the content edge. The padding edge of a box defines the edges of the containing block established by the box.
  - border edge (the border edge surrounds the box's border). If the border has 0 width, the border edge is the same as the padding edge.
  - margin edge or outer edge (the margin edge surrounds the box margin). If the margin has 0 width, the margin edge is the same as the border edge.

- The dimensions of the content area of a box -- the content width and content height -- depend on several factors: whether the element generating the box has the 'width' or 'height' property set, whether the box contains text or other boxes, whether the box is a table, etc.

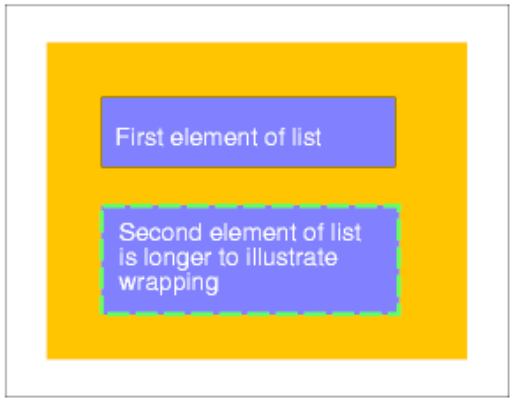
- The box width is given by the sum of the left and right margins, border, and padding, and the content width. The height is given by the sum of the top and bottom margins, border, and padding, and the content height.

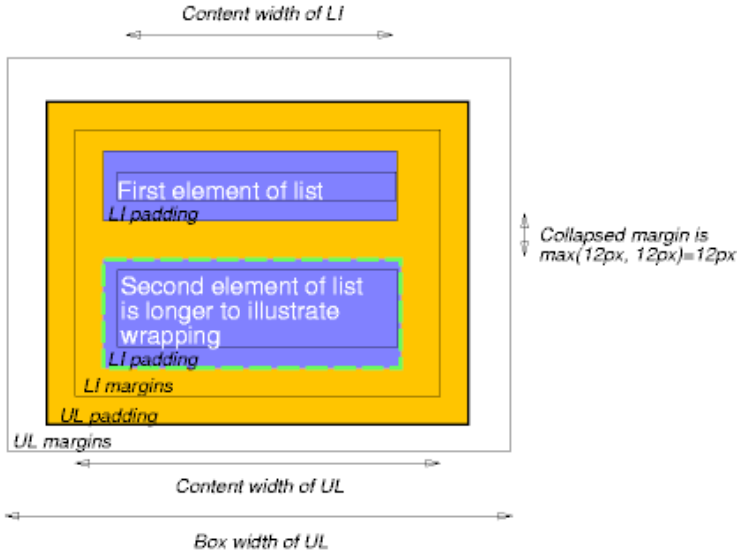
- The background style of the various areas of a box are determined as follows:
  - * Content area: The 'background' property of the generating element.
  - * Padding area: The 'background' property of the generating element.
  - * Border area: The border properties of the generating element.
  - * Margin area: Margins are always transparent.

Example:

- CSS

```
ul {
        background: green;
        margin: 12px 12px 12px 12px;
        padding: 3px 3px 3px 3px;
}
li {
        color: black;
        background: gray;
        margin: 12px 12px 12px 12px;
        padding: 12px 0px 12px 12px;
        list-style: none;
}
li.withborder {
        border-style: dashed;
        border-width: medium;
        border-color: black;
}
```





- XHTML

```
<ul>
<li>First element of list</li>
<li class="withborder">Second element of list is longer to illustrate wrapping.</li>
</ul>
```

- Note that:
  - The content width for each LI box is calculated top-down; the containing block for each LI box is established by the UL element.
  - The height of each LI box is given by its content height, plus top and bottom padding, borders, and margins. Note that vertical margins between the LI boxes collapse.
  - The right padding of the LI boxes has been set to zero width (the 'padding' property). The effect is apparent in the second illustration.
  - The margins of the LI boxes are transparent -- margins are always transparent -- so the background color (green) of the UL padding and content areas shines through them.
  - The second LI element specifies a dashed border (the 'border-style' property).
- In this specification, the expression collapsing margins means that adjoining margins (no padding or border areas separate them) of two or more boxes (which may be next to one another or nested) combine to form a single margin.

CSS2 Visual formatting model

- In the visual formatting model, each element in the document tree generates zero or more boxes according to the box model. The layout of these boxes is governed by:
  - box dimensions and type.
  - positioning scheme (normal flow, float, and absolute).
  - relationships between elements in the document tree.
  - external information (e.g., viewport size, intrinsic dimensions of images, etc.).

- There are several type of boxes. A box's type affects, in part, its behavior in the visual formatting model. The 'display' property specifies a box's type.
  - Block-level elements: block-level elements are those elements of the source document that are formatted visually as blocks (e.g., paragraphs). Several values of the 'display' property make an element block-level: 'block', 'list-item', 'compact' and 'run-in' (part of the time; see compact and run-in boxes), and 'table'.
  - Inline-level elements and inline boxes: inline-level elements are those elements of the source document that do not form new blocks of content; the content is distributed in lines (e.g., emphasized pieces of text within a paragraph, inline images, etc.). Several values of the 'display' property make an element inline: 'inline', 'inline-table', 'compact' and 'run-in'. Inline-level elements generate inline boxes.

CSS2 Positioning schemes

- In CSS2, a box may be laid out according to three positioning schemes:
  - Normal flow. In CSS2, normal flow includes block formatting of block boxes, inline formatting of inline boxes, relative positioning of block or inline boxes, and positioning of compact and run-in boxes.
  - Floats. In the float model, a box is first laid out according to the normal flow, then taken out of the flow and shifted to the left or right as far as possible. Content may flow along the side of a float.
  - Absolute positioning. In the absolute positioning model, a box is removed from the normal flow entirely (it has no impact on later siblings) and assigned a position with respect to a containing block.

- The 'position' and 'float' properties determine which of the CSS2 positioning algorithms is used to calculate the position of a box. Position can be static, relative, absolute and fixed
  - static: the box is a normal box, laid out according to the normal flow. The 'left' and 'top' properties do not apply.
  - relative: the box's position is calculated according to the normal flow (this is called the position in normal flow). Then the box is offset relative to its normal position. When a box B is relatively positioned, the position of the following box is calculated as though B were not offset.
  - absolute: the box's position (and possibly size) is specified with the 'left', 'right', 'top', and 'bottom' properties. These properties specify offsets with respect to the box's containing block. Absolutely positioned boxes are taken out of the normal flow. This means they have no impact on the layout of later siblings.
  - fixed: the box's position is calculated according to the 'absolute' model, but in addition, the box is fixed with respect to some reference.

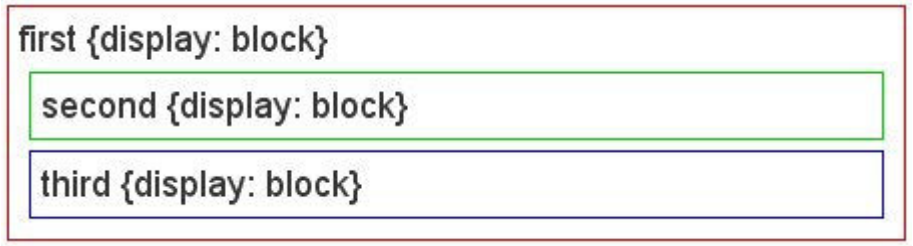
- Casi di esempio in aula: XHTML con CSS e div posizionati in maniera assoluta e relativa

- Normal flow: boxes in the normal flow belong to a formatting context, which may be block or inline, but not both simultaneously. Block boxes participate in a block formatting context. Inline boxes participate in an inline formatting context.
  - In a block formatting context, boxes are laid out one after the other, vertically, beginning at the top of a containing block. The vertical distance between two sibling boxes is determined by the 'margin' properties. In a block formatting context, each box's left outer edge touches the left edge of the containing block (for right-to-left formatting, right edges touch). This is true even in the presence of floats (although a box's content area may shrink due to the floats).
  - In an inline formatting context, boxes are laid out horizontally, one after the other, beginning at the top of a containing block. Horizontal margins, borders, and padding are respected between these boxes. When several inline boxes cannot fit horizontally within a single line box, they are distributed among two or more vertically-stacked line boxes.

Example:

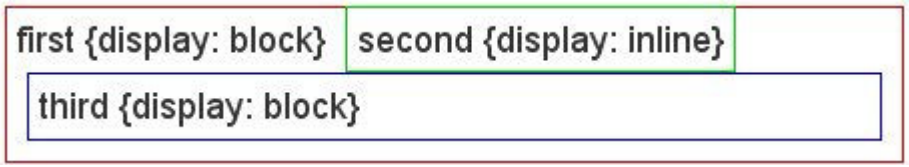
- Esempio normal flow: display block

```
<div class="first">
first {display: block}
<div class="second">second {display: block}</div>
<div class="third">third {display: block}</div>
</div>
```



- Esempio normal flow: display inline

```
<div class="first">
first {display: block}
<div class="second" style="display: inline">second {display: inline}</div>
<div class="third">third {display: block}</div>
</div>
```



- Esempio normal flow: display none

```
<div class="first">
first {display: block}
<div class="second" style="display: none">second {display: none}</div>
<div class="third">third {display: block}</div>
</div>
```

- Floats: a float is a box that is shifted to the left or right on the current line. The most interesting characteristic of a float (or "floated" or "floating" box) is that content may flow along its side (or be prohibited from doing so by the 'clear' property). Content flows down the right side of a left-floated box and down the left side of a right-floated box.

- Any floated box becomes a block box that is shifted to the left or right until its outer edge touches the containing block edge or the outer edge of another float. The top of the floated box is aligned with the top of the current line box (or bottom of the preceding block box if no line box exists). If there isn't enough horizontal room on the current line for the float, it is shifted downward, line by line, until a line has room for it.

- Since a float is not in the flow, non-positioned block boxes created before and after the float box flow vertically as if the float didn't exist. However, line boxes created next to the float are shortened to make room for the floated box. Any content in the current line before a floated box is reflowed in the first available line on the other side of the float.

Example:

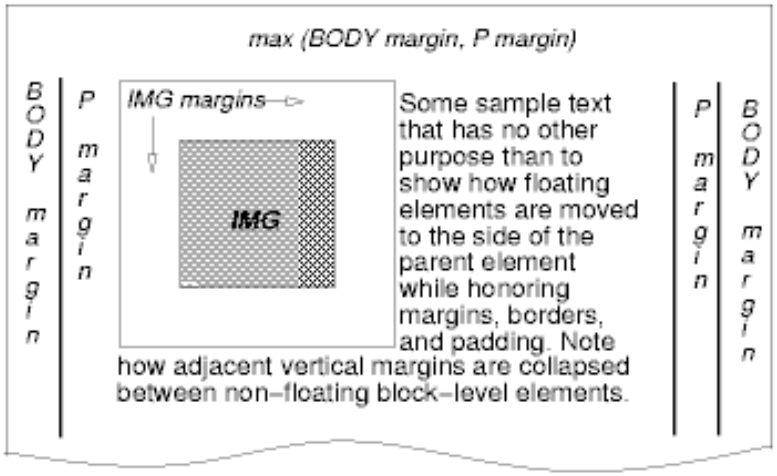
- CSS

  img { float: left }

  body, p, img { margin: 2em }

- XHTML

  Some sample text that has no other...



Riferimenti

- MMM forum http://morpheus.micc.unifi.it/forum/
- MICC learning http://morpheus.micc.unifi.it/learning/

- W3C CSS2 specs http://www.w3.org/TR/REC-CSS2/cover.html#minitoc (sez. 8+9)
- w3c XHTML overview http://www.w3.org/TR/xhtml11/Overview.html
- w3schools XHTML http://www.w3schools.com/xhtml/

- CSS Vault http://cssvault.com/
- CSS Beauty http://www.cssbeauty.com/
- webstandards http://webstandards.org/
- A list apart http://alistapart.com/
- Boxes and arrows http://www.boxesandarrows.com/
- Maxdesign tutorials http://css.maxdesign.com.au/
- layout-o-matic http://www.inknoise.com/experimental/layoutomatic.php
- colourlovers http://colourlovers.com/