



Informatica Umanistica A.A. 2007/2008

LEZIONE 4

eXtensible Markup Language (XML) INTRODUZIONE



Lezione 4 SOMMARIO

Lezione 3

1. HTML - Tabelle
2. CSS

Lezione 4

1. Introduzione XML
2. Cos'è un documento XML
3. XML – HTML
4. Un po' di storia...
5. XML – DTD – XSL
6. Dichiarazione XML
7. Elementi
8. Attributi
9. Entità
10. CDATA
11. PCDATA

1. Tabelle

- Le tabelle costituiscono un modo per organizzare "dati" in righe e colonne;
- Le "celle" delle tabelle possono contenere:
 - Testo
 - Immagini
 - Link
 - Altre tabelle
 -
- Le tabelle permettono anche di "strutturare" le pagine, di raggruppare celle, contenere titoli e didascalie;

1. Tabelle

Elementi principali

- Il TAG fondamentale: **<table>...</table>**
- Didascalia: **<caption>...</caption>**
Indica una didascalia della tabella. Se si vuole inserire, va messo subito dopo <table>
- Gli elementi di <table> sono
 - **Riga <TR>...</TR>** (Table Row)
 - **Cella <TD>...</TD>** (Table Data) deve essere all'interno di una <TR>

Esempio:

```
<table>  
  <caption> Tabella con una cella</caption>  
  <tr> <td> </td> </tr>  
</table>
```

1. Tabelle - Attributi di tabelle, righe e celle: <table>, <tr> e <td>

- **Larghezza** della tabella <TABLE **WIDTH**="?"> (in **pixel o percentuale %**)
- Posso aggiungere a righe <tr> e celle <td> attributi quali:
 - **ALIGN** (alignment): allinea il testo della cella a destra (right), sinistra (left) e centrato (center).
 - **VALIGN** (vertical alignment): allinea il testo della cella sul margine superiore (top), sul margine inferiore (bottom), e in mezzo (middle)
 - **COLSPAN**: quante colonne pesa una singola <td> (= colonne da occupare con una singola cella)
 - **ROWSPAN**: quante righe pesa una singola <td> (= righe da occupare con una singola cella)
- Larghezza della cella in percentuale <TD **WIDTH**="%">
- **Colore di sfondo** della cella <TD **BGCOLOR**="#0088dd">

1. Tabelle - Esempio

```
<table width="50%" height="30%" cellspacing="4" cellpadding="2" border="1">
  <caption align="bottom">Questa è una tabella di esempio</caption>
  <tr>
    <th>Intestazione 1</th>
    <th>Intestazione 2</th>
  </tr>
  <tr>
    <td align="center">Cella 1 colonna 1 testo centrato</td>
    <td valign="top">Cella 1 colonna 2 allineata sopra</td>
  </tr>
  <tr>
    <td>Cella 2 colonna 1</td>
    <td width="70%">Cella 2 colonna 2</td>
  </tr>
</table>
```

1. Tabelle – Esempio

```
<table width="50%"  
height="30%" cellpadding="2" cellspacing="4" border="1">
```

```
<th>Intestazione 1</th>
```

```
<th>Intestazione 2</th>
```

Intestazione 1	Intestazione 2
Cella 1 colonna 1 testo centrato	Cella 1 colonna 2 allineata sopra
Cella 2 colonna 1	Cella 2 colonna 2

Questa è una tabella di esempio

```
<caption align="bottom">Questa è una tabella di esempio</caption>
```

1. Tabelle – Esempio colspan

Colspan=2

prima cella della seconda riga seconda cella della seconda riga

```
<table border="1">  
  <tr>  
    <td colspan="2">Colspan=2</td>  
  </tr>  
  <tr>  
    <td>prima cella della seconda riga</td>  
    <td>seconda cella della seconda riga</td>  
  </tr>  
</table>
```

Nella prima riga ho solo 1 cella,
larga due colonne

1. Tabelle – Esempio rowspan

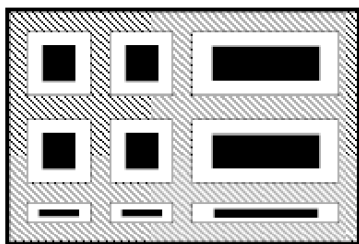
Rowspan=2	seconda cella della prima riga
	unica cella della seconda riga

```
<table border="1">
  <tr>
    <td rowspan="2">Rowspan=2</td>
    <td>seconda cella della prima riga</td>
  </tr>
  <tr>
    <td>unica cella della seconda riga</td>
  </tr>
</table>
```

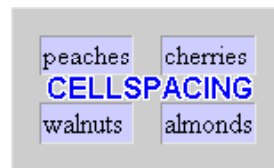
Nella prima riga ho 2 celle, di cui solo la prima occupa 2 righe

1. Tabelle Bordi e Margini

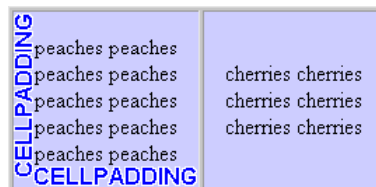
Table border 



- Cellspacing 
- Cellpadding 
- Cell content 



Cellspacing: spazio tra celle



Cellpadding: spazio tra contenuto e bordo della cella

1. Tabelle

Bordi e Margini: cellspacing e cellpadding

1.

a.

peaches	cherries
walnuts	almonds

<TABLE CELLSPACING="2">

b.

peaches	cherries
walnuts	almonds

<TABLE CELLSPACING="10">

2.

a.

peaches	cherries
walnuts	almonds

<TABLE CELLPADDING="1">

b.

peaches	cherries
walnuts	almonds

<TABLE CELLPADDING="10">

1. Tabelle

Bordi e Margini – in pixel

- o **Bordo:**
<table border="?" bordercolor="?">
- o **Spazio tra celle:**
<TABLE CELLSPACING="?">
- o **Spazio all'interno:**
<TABLE CELLPADDING="?">

1. Tabelle

Esempio (3 righe – 2 colonne)

```
<table><caption> eventuale didascalia </caption>
<tr>
  <td><strong>Cella 1, Riga 1</strong></td>
  <td><strong>Cella 2, Riga 1</strong></td>
</tr>
<tr>
  <td><em>Cella 1, Riga 2</em></td>
  <td><em>Cella 2, Riga 2</em></td>
</tr>
<tr>
  <td><em>Cella 1, Riga 3</em></td>
  <td><em>Cella 2, Riga 3</em></td>
</tr>
</table>
```

1° riga

2° riga

3° riga

1. Tabelle

Riassunto Elementi Tabelle

```
<table width="" border="" cellspacing=""
cellpadding="" bgcolor="" bordercolor="">
<caption align="" > didascalia </caption>
<tr bgcolor="" align="" >
  <td bgcolor="" rowspan="" colspan="" width
="" height="" align="" >
[...]
```

1. Tabelle

Esempio Tabella a grandezza fissa

```
<table width="75%" height="20%" border="3">
<tr>
<td><strong>Cognome</strong></td>
<td><strong>Nome</strong></td>
<td><strong>Indirizzo</strong></td>
</tr>
<tr>
<td><em> Ferron</em></td>
<td><em> Michela</em></td>
<td><em>Via .... </em></td>
</tr>
<tr>
<td><em>Massa</em></td>
<td><em>Paolo</em></td>
<td><em>Via ....</em></td>
</tr>
</table>
```

NB: Il tag `` sembra identico al tag ``:
In realtà il testo marcato con `strong` verrà letto con un tono di voce più alto dai browser vocali, cosa che non avviene per ``

1. Tabelle

Link nelle tabelle

```
<table width="50%" border="0">
<tr>
<td width="30%" align="center"><strong>
Link</strong></td>
<td width="70%"
align="center"><strong>Categoria</strong></td>
</tr>
<tr>
<td align="center" ><a
href="http://www.google.it">Google.it</a></td>
<td align="center" ><em>Motore di ricerca </em></td>
</tr>
<tr>
<td align="center" ><a
href="http://www.yahoo.it">Yahoo.it</a></td>
<td align="center" ><em>Motore di ricerca</em></td>
</tr>
</table>
```

1. Tabelle

Ultimo esempio

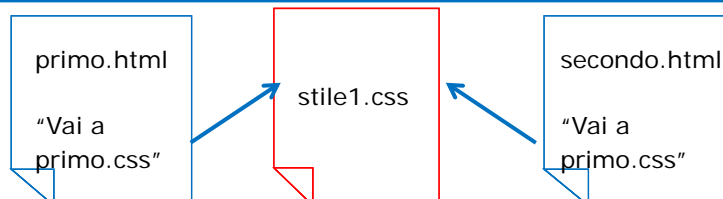
```
<table width="75%" border="3" bordercolor="#220066"
  bgcolor="#ccbff" cellpadding="5" cellspacing="10">
<tr>
  <td width="30%" bgcolor="#7755ff"> <strong> Link
  </strong></td>
  <td width="70%" bgcolor="#7755ff"><strong> Categoria
  </strong></td>
</tr>
<tr>
  <td><a href="http:\\www.google.it">Google.it</a></td>
  <td><em>Motore di ricerca</em></td>
</tr>
<tr>
  <td><a href="http:\\www.yahoo.it">Yahoo.it</a></td>
  <td><em>Motore di ricerca </em></td>
</tr>
</table>
```


2. Fogli di stile a cascata:

CSS - Cascading Style Sheet

- O **fogli di stile**, definiscono la **presentazione grafica** di documenti HTML, XHTML e XML.
- "A cascata": nel mio documento HTML posso incorporare più fogli di stile, ognuno dei quali è in grado di prevalere sull'altro grazie a delle regole gerarchiche

Quando accedo alla pagina web, il browser viene indirizzato al file CSS per recuperare le informazioni sulle modalità di visualizzazione del contenuto






2. Separare contenuto (html) e presentazione (css): vantaggi

Per l'utente

- **siti web leggeri e tempi di caricamento brevi:** un solo file CSS può controllare la presentazione grafica di tutte le pagine
- l'utente può **personalizzare il layout** (dimensioni del testo, colori)
- **trasformazione elegante delle pagine** a seconda dello strumento (sintetizzatore vocale, display Braille o testuale, cellulare)



2. Separare contenuto (html) e presentazione (css): vantaggi

Per il webmaster

aggiornamento più veloce delle pagine perché:

- il documento HTML è pulito e ordinato
- lo stile di tutte le pagine si può cambiare modificando un unico file CSS
- **Esempio:**
<http://www.csszengarden.com/tr/italiano/>

2. Come specifico il CSS per il mio HTML?

1) Inserendo nel tag `<head>` della pagina un collegamento ad un foglio di stile esterno con estensione `.css`

```
<head> <title>Esempio</title>
```

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

```
</head>
```

2) Inserendo tra gli specifici tag `<style>` e `</style>` le dichiarazioni CSS.

```
<head> <title>Esempio</title>
```

```
<style type="text/css">
```

```
...codice css...
```

```
</style>
```

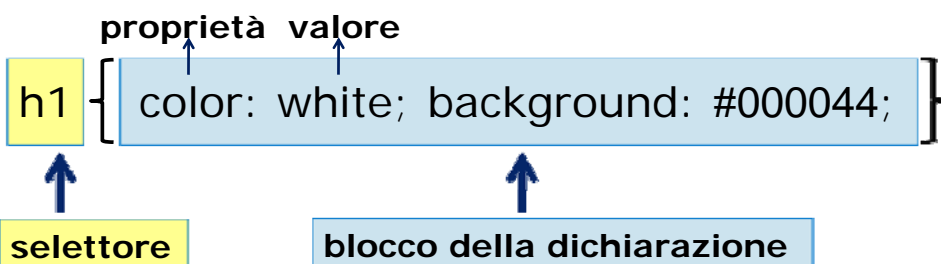
```
</head>
```

NOI USIAMO LA 1)

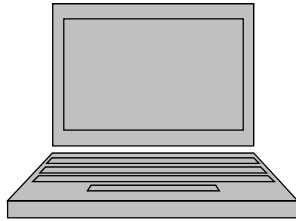
2. Com'è fatto un foglio di stile CSS

Un foglio di stile è composto da due tipi di **dichiarazioni**: **regole** e **commenti** (i commenti non vengono "letti" dal browser: `/* commento */`)

Struttura di una regola



2. CSS – Ma come digito le parentesi graffe?



```
Alt+123 → {  
Alt+125 → }
```

1. Parentesi graffa aperta {

Tenendo premuto il tasto **Alt** digitare **123** in sequenza sul tastierino numerico a destra

2. Parentesi graffa chiusa }

Tenendo premuto il tasto **Alt** digitare **125** in sequenza sul tastierino numerico a destra

2. Cosa posso specificare con un CSS

Proprietà CSS sono molte, circa 60.

- **background**: definisce lo sfondo di un elemento;
- **border**: definisce il bordo di un elemento;
- **color**: definisce il colore del testo di un elemento;
- **float**: definisce un blocco flottante, ovvero che permette la disposizione di altri elementi ai suoi lati;
- **font**: definisce le proprietà del carattere;
- **margin** e **padding**: definiscono lo spazio circostante gli elementi (esterno, interno);
- **text-align**: definisce l'allineamento del testo.

2. Cosa posso specificare con un CSS: i selettori

- L'aspetto grafico (formattazione) di qualcosa in una pagina HTML.
- Di cosa?
 - 1) Elementi di un certo **tipo** (tag, es: "h1") con i **selettori di tipo**
 - 2) Elementi di una certa **classe** con i **selettori di classe**
 - 3) Elementi identificati con un **id** con i **selettori di identificatori**

Li specifico tramite un selettore: una dichiarazione che **seleziona** la parte o le parti di un documento soggette ad una specifica regola

2. Selettori di tipo

- sono i selettori più generici, e indicano che la regola deve essere applicata a tutti gli elementi del tipo indicato
 - **h1** {...} → Applica la regola a tutto ciò che nell'html è marcato con il tag <h1>
 - **ul** {...}
 - **p** {...}
 - **a** {...}
 - **div** {...}
 - ***** {...} → *** {...}** è un selettore particolare: seleziona **tutti gli elementi** del documento

2. Selettori di tipo: esempi

primo.html → stile1.css

```
<html> <head>
<link rel="stylesheet" type="text/css"
href="stile1.css"> </head>
<body>

<H1> Titolo </H1>

<P> Primo paragrafo </P>

<UL> Lista non ordinata
<LI> primo elemento</LI>
<LI> secondo elemento</LI>
</UL>

</body>
</html>
```

```
h1 { background-color: yellow;
color: blue; }

p { font-family: arial, verdana, sans-
serif;
font-size: 12px;
text-decoration: underline;
background-color: yellow;
color: blue;
border : 1px solid red; }

ul {list-style-type: square; }
```

2. Selettori di tipo: esempi - pseudoclassi

- **a:link** { background: #fff; color: #f00; text-decoration: underline; }
- **a:visited** { background: #fff; color: #f77; text-decoration: underline; }
- **a:hover, a:active, a:visited** { background: #f00; color: #fff; text-decoration: none; }

Pseudo- classi:

Applicano uno stile per un elemento al verificarsi di certe condizioni

2. Selettori di tipo: esempi pseudoclassi

primo.html → stile1.css

```
<html> <head>
<link rel="stylesheet"
type="text/css"
href="stile1.css"> </head>
<body>

<A HREF="http://www.unitn.it">
Link a unitn.it </A>

</body>
</html>
```

```
a {font-size: 30pt;}
a:link { background: #fff; color:
#f00; text-decoration: underline;
}
a:hover { background: #f00;
color: #fff; text-decoration:
none; }
a:visited { background: #fff;
color: #000; text-decoration:
underline; }
```

2. Selettori di tipo: esempi

HTML

```
<div>Vuolsi cosi' cola'</div>
<div>dove si puote</div>
<div>cio' che si vuole</div>
```

CSS

```
div {
border: 3px solid #ff0000;
width: 30%;
}
```

NB: il tag **<div>** (division) identifica una **sezione** in un documento html; si usa **per raggruppare degli elementi in blocchi e formattarli** con un particolare stile

```
Vuolsi cosi' cola'
dove si puote
cio' che si vuole
```


2. Selettori di tipo: esempi

HTML

```
<div>Vuolsi cosi' cola'</div>  
<div>dove si puote</div>  
<div>cio' che si vuole</div>
```

CSS

```
div {  
  border: 3px dotted green;  
  margin : 5px;  
  width: 30%;  
}
```



Vuolsi cosi' cola'
dove si puote
cio' che si vuole

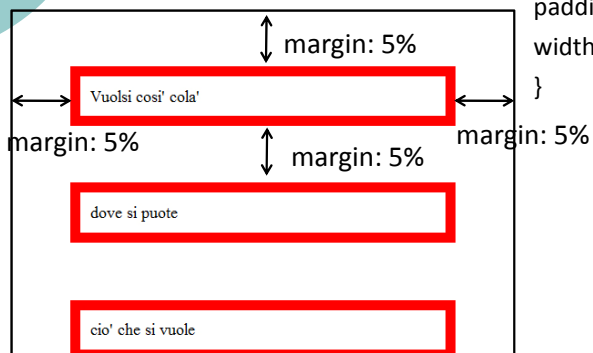
2. Selettori di tipo: esempi

HTML

```
<div>Vuolsi cosi' cola'</div>  
<div>dove si puote</div>  
<div>cio' che si vuole</div>
```

CSS

```
div {  
  border: 10px solid #ff0000;  
  margin: 5%;  
  padding: 10px;  
  width: 30%;  
}
```



2. Selettore del discendente

Seleziona il **discendente** di un elemento.

Un elemento è discendente di un altro se è **contenuto al suo interno**, a qualsiasi livello.

Es.: liste

- `ul li { color: #000000; }`
- `ul li a { text-decoration: underline; }`

Seleziona tutti i link `<a>` contenuti nei list item ``, contenuti a loro volta in liste non ordinate `` (unordered list)

2. Selettori del discendente: esempi

primo.html → stile1.css

```
<html> <head>
<link rel="stylesheet" type="text/css"
href="stile1.css"> </head>
<body>

<UL>
<LI><A HREF="http://www.unitn.it" >
Link a Unitn.it </A>
</LI>
<LI> secondo elemento della lista
</LI>
</UL>

</body>
</html>
```

```
ul li a {text-decoration:
underline;
font-size: 20px;
font-family: verdana, arial,
sans-serif;}
```

Domanda: succederà qualcosa al secondo elemento della lista?

2. Selettore di classi (class)

- Le classi definiscono un **insieme di oggetti omogenei**. Per associare un elemento a una è sufficiente specificarne il nome nel file HTML attraverso l'attributo **class**.

HTML:

- `<p class="testorosso">...</p>`
`<div class="testorosso">...</div>`
`<table class="testorosso">...</table>`
- Tutti questi tag appartengono alla classe "testorosso"

CSS (qui la classe si indica con il punto ". "):

- **`.testorosso { color: red; }`**

Seleziona tutto ciò che nell'html è indicato con la classe "testorosso"

2. Selettori di classi: esempi

primo.html → stile1.css

```
<html> <head>
<link rel="stylesheet" type="text/css"
href="stile1.css"> </head>
<body>

<P CLASS="testorosso">Paragrafo
testorosso</p>
<DIV CLASS="testorosso">Div
testorosso</DIV>
<H1>H1 generico (no testorosso)</H1>

</body>
</html>
```

```
.testorosso { color: red; }
```

A tutto ciò che nell'html è indicato con la classe "testorosso" verrà applicata la regola

2. Selettori di identificatori (id)

- Simili alle classi, ma **identificano in modo univoco un elemento**. Se assegno ad un paragrafo l'id "testorosso", non potrò più usare questo valore nel resto della pagina.

HTML:

- `<p id="testorosso">...</p>`
`<div id="intestazione">...</div>`

CSS (qui l'identificatore si indica con il cancelletto "#"):

- `#testorosso { color: red; }`

- `#intestazione { font-size: 16px; }`

Seleziona in modo univoco l'unico elemento identificato nell'html con l'id "testorosso"

2. Selettori di identificatori: esempi

primo.html → stile1.css

```
<html> <head>
<link rel="stylesheet" type="text/css"
href="stile1.css"> </head>
<body>

<P ID="testorosso">paragrafo con
ID=testorosso</P>
<DIV ID="intestazione">Div con
id=intestazione </DIV>

</body>
</html>
```

```
#testorosso { color: red; }
#intestazione {
font-weight:bold;
}
```

I due identificatori "testorosso" e "intestazione" sono unici e diversi tra loro

2. CSS - esempi

HTML	CSS
<pre><p> Cantami, o Diva, del Pelide Achille
 l'ira funesta che infiniti addusse
 lutti agli Achei, molte anzi tempo all'Orco
 generose travolse alme d'eroi,
 ...</p></pre>	<pre>p:first-letter { font-size: 200%; } p:first-line { font-style: italic; }</pre>

p:first-letter e **p:first-line** sono **pseudo-elementi**: servono per aggiungere un effetto speciale ad alcuni selettori, in questo caso la prima lettera e la prima riga.

2. CSS – ultimo esempio

HTML	CSS
<pre><p> Cantami, o Diva, del Pelide Achille
 l'ira funesta che infiniti addusse
 lutti agli Achei, molte anzi tempo all'Orco
 generose travolse alme d'eroi,
 ...</p> <p>"Mare mare..."
 Immagine online. Flickr.com. 2 Maggio 2009. link all'immagine </p></pre>	<pre>body { background-image: url('http://farm2.static.fli ckr.com/1022/998882346 _e57fadaa57.jpg?v=0'); background-repeat: no- repeat; }</pre>

Attenzione al copyright! Usate immagini vostre o con licenze che ne permettono l'utilizzo (es. creative commons). **In generale salvate le immagini nella vostra cartella, non prendetele dal web!** (possono essere cancellate o sostituite senza che voi lo sappiate)



2. CSS – Esercizi

<http://www.antonibucchiarone.it/Esercizi/Esercizi-Lezione3.pdf>



XML - Acronimo

XML (*eXtensible Markup Language*):

è l'abbreviazione di Linguaggio Estensibile di Markup ed è l'acronimo dell'espressione inglese Extensible Markup Language.

Guida:

<http://www.html.it/xml/guida/index.html>

Cosa è XML?

- È un "**markup language**" molto simile all'HTML
- È pensato per **descrivere dati e la loro struttura** (non la loro presentazione)
- Non esistono tag predefiniti: **lo sviluppatore definisce i suoi tag personali**
 - Per questo motivo l'XML si può considerare un **meta-linguaggio di markup** = linguaggio che permette di definire altri linguaggi di markup
- Utilizza un documento di definizione dei tipi (**Document Type Definition - DTD**) per descrivere i dati.
 - Documento che definisce i tag utilizzabili in un documento XML, la loro struttura e altre info sugli attributi dei tag

Principali differenze fra HTML e XML

XML e HTML sono stati pensati per scopi diversi.

HTML

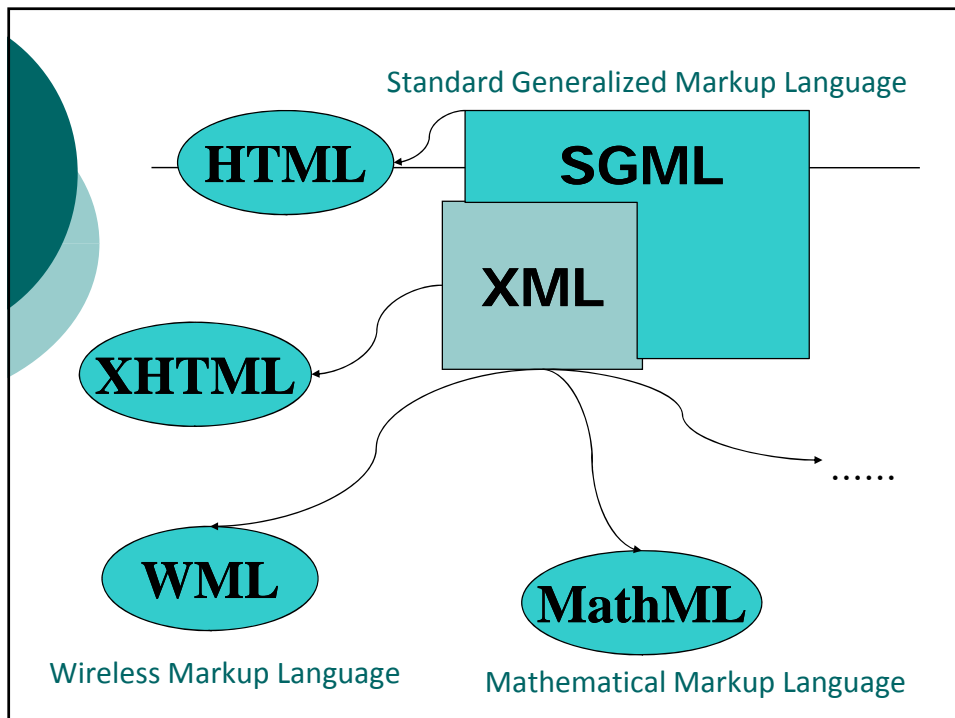
- HTML è un linguaggio di markup
- I tag dell'HTML sono definiti dal W3C
- HTML è pensato per visualizzare dati e porre l'attenzione su come presentarli.
- HTML si occupa di visualizzare dati sul Web

XML

- XML è un meta-linguaggio di markup
- I tag dell'XML sono definiti da noi
- XML è pensato per descrivere dati (loro struttura) e porre l'attenzione su cosa sono.
- XML è pensato per descrivere le informazioni e per trasportare dati.
- XML non è un sostituto di HTML: è più generico

HTML vs XML

- In HTML informazione e sua visualizzazione **non sono** necessariamente **separate**
- In XML informazione e sua visualizzazione **sono separate**





Un po' di storia

1997: la convergenza su XML

- Internet e il Web hanno bisogno di **standard** per evolversi
- A questo scopo nel 1994 è stato istituito il **W3C** per definire gli standard per il Web
- Tuttavia negli anni '90 la rapida diffusione del Web ha scatenato una **guerra commerciale** tra i browser Netscape e Microsoft, che introducevano **estensioni proprietarie** per l'HTML (es. tag <blink> per Netscape, tag <marquee> per IE)
- Necessità di un linguaggio di markup che offrisse libertà nella definizione dei tag
- 1996: XML Working Group del W3C
- 1997: escono le prime specifiche di XML



Un documento XML è composto da tre parti

- **XML** (eXtensible Markup Language)
 - i dati (struttura fisica - *document instance*)
 - *Quale informazione e' contenuta nel file?*
- **DTD** (Document Type Definition)
 - lo schema dei dati (struttura logica)
 - *Quali tag posso usare? Dove? Come?*
- **XSL** (eXtensible Stylesheet Language)
 - la presentazione dei dati (*stylesheet*)
 - *Come visualizzo questi dati?*

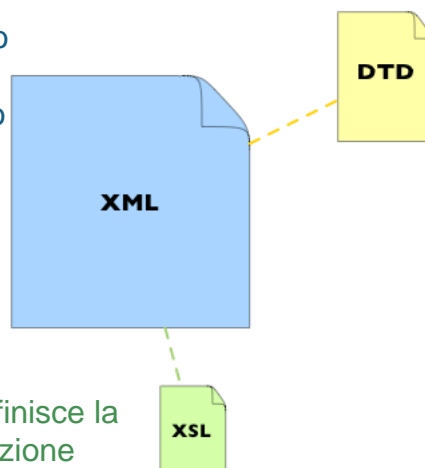
XML – DTD – XSL

Nel file XML c'è il contenuto
“vero e proprio”
semanticamente strutturato

La DTD (o lo Schema)
definiscono la sintassi
dell'XML



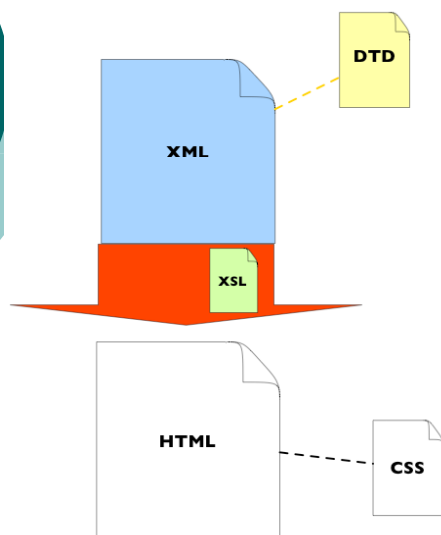
Il file XSL definisce la
visualizzazione



XML – HTML

XML, come HTML è
un dialetto di SGML
orientato al web, ma
**non è un'alternativa
a HTML.**

Il file **XML** viene
trasformato da un
programma **in un
altro linguaggio**,
tipicamente HTML ma
non solo (testo, PDF,
..., qualsiasi formato)



Piccolo esempio

```
<lettera>
  Cara Beatrice, ti scrivo per comunicarti ...
  <saluti> cari saluti,
  <firma> Dante </firma>
</saluti>
</lettera>
```

Questo e' il file XML (dati). **I tag sono liberi e non definiti dal W3C.**

"**lettera**" e' l'elemento **radice (root)**.

Come viene visualizzato? Lo vediamo dopo

La dichiarazione XML

- La **dichiarazione XML** è **obbligatoria** e deve essere posta **all'immediato inizio del documento** (ovvero come primo carattere! Niente spazi prima!):

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
```

- Gli attributi sono:
 - **version**: (*obbligatorio*) la **versione di XML** usata.
 - **encoding**: (*opzionale*) nome della **codifica dei caratteri** usata nel documento (default: Unicode UTF-8 o 16)
 - **standalone**: (*opzionale*) (default: no)
 - **standalone=yes** → il file non fa **riferimento ad altri file esterni**, ovvero il solo file XML contiene dati e DTD
 - Noi useremo sempre standalone="yes"
 - standalone=no → e' necessario un file esterno (che contiene il DTD)

Piccolo esempio (2)

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
```

```
<lettera>
```

Caro Beatrice, ti scrivo per comunicarti...

```
<saluti> cari saluti,
```

```
<firma> Dante </firma>
```

```
</saluti>
```

```
</lettera>
```

- Qual è differenza con HTML?
- Qual è l'elemento radice?
- Come facciamo a sapere che "saluti" è un tag valido? O che ci deve essere una sola "firma"?

Gli elementi (non sono tag)

- Gli elementi sono le parti di documento dotate di un senso proprio.
- Un **elemento** è individuato da un **tag iniziale**, un **contenuto** ed un **tag finale**.
- Non confondere i tag con gli elementi!

```
<TITOLO> Primo file XML </TITOLO>
```

elemento TITOLO, racchiuso dai tag TITOLO e /TITOLO

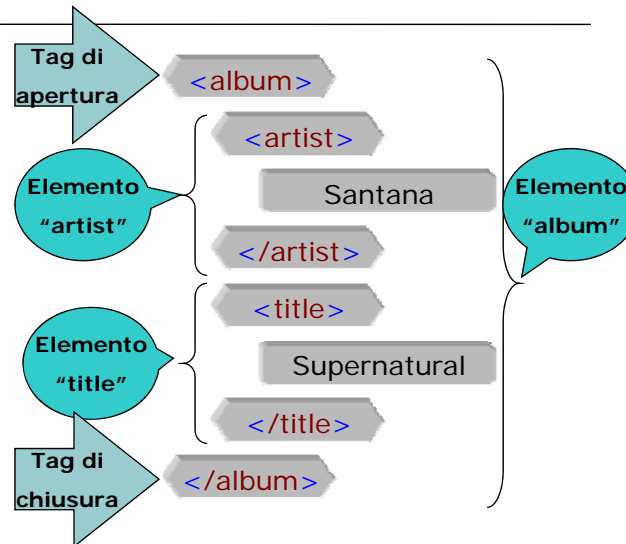
```
<saluti> cari saluti <firma> Marianna </firma></saluti>
```

elemento saluti (contiene un elemento firma)

Elementi - esempio

- Gli **elementi** sono alla base della strutturazione dei documenti XML.

- Un elemento è un **frammento di dati, limitato ed identificato** (tramite un nome) da un *tag*.



Commenti

- I documenti di markup possono contenere commenti, ovvero note da un autore all'altro, da un editore all'altro, ecc.
- Queste note non fanno parte del contenuto del documento, e le applicazioni di markup li ignorano.
- Sono molto comodi per passare informazioni tra un autore e l'altro, o per trattenere informazioni per se stessi, nel caso le dimenticassimo.

<!-- Questo è ignorato dal parser -->

Elementi – Regole per documenti ben formati

- I nomi degli elementi sono **case-sensitive** (devono coincidere in apertura e chiusura).
- **Ogni elemento aperto deve essere chiuso** entro la fine del documento.
- Gli elementi possono essere nidificati, e in tal caso **vanno chiusi esattamente nell'ordine inverso a quello di apertura**.
- Un documento XML deve avere **un unico elemento "radice"**, in cui tutti gli altri sono nidificati
- I valori si racchiudono tra **singoli (' ') o doppi (" ") apici**
- Gli elementi non devono avere due attributi con lo stesso nome
- I nomi degli elementi e degli attributi non devono contenere caratteri speciali (&, <, > ...)
- I nomi degli elementi non devono contenere spazi

Documenti ben formati esempi violazioni

1. <articolo
titolo=test>
...
</articolo>

2. <Articolo
titolo="test">
...
</articolo>

3. <paragrafo>
<testo> ABC
</paragrafo>
</testo>

4. <articolo
titolo="test">
...
<articolo>

5. <articolo
titolo="test"
titolo="test2">
...
</articolo>

6. <primo paragrafo>
<testo> ABC
</testo>
</primo paragrafo>

Elementi - Sintassi

- Il **tag di apertura** di un elemento ha la forma seguente:

<nome attributi>

- **nome** è il nome dell'elemento.
- **attributi** è una lista di attributi per l'elemento (che può anche non apparire).

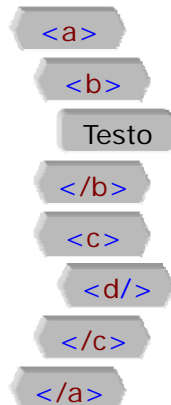
- Il **tag di chiusura** corrispondente ha la forma seguente:

</nome>

- **nome** è lo stesso identificatore usato nell'apertura.
- Alcuni elementi possono essere **privi di contenuto**; in questo caso è possibile omettere il tag di chiusura scrivendo quello di apertura nel modo seguente (*empty tag*):

<nome attributi />

Elementi - Gerarchia



- Gli elementi, nidificandosi, creano la **struttura ad albero** tipica dei documenti XML.
- All'interno di questa struttura si definiscono alcuni "rapporti di parentela" utili per individuare gli elementi:

- *a* è il nodo **radice**
- *b* e *c* sono figli di *a*, il *testo* è **figlio** di *b*, *d* è figlio di *c*
- *c* è il **padre** di *d*, *b* è il padre del *testo*, *a* è il padre di *b* e *c*
- *b* e *c* sono **fratelli**
- *b*, *c*, *d* e il *testo* sono discendenti di *a*, *d* è un **discendente** di *c*, il *testo* è un discendente di *b*
- *a* è un **predecessore** di *b*, *c*, *d* e del *testo*, *b* è un predecessore del *testo*, *c* è un predecessore di *d*.

Attributi

- Gli **elementi possono contenere uno o più attributi**, che sono termini aggiuntivi che definiscono con più precisione l'elemento, es. `<section id="uno">` (qual è l'attributo?)
- Il **valore** degli attributi è racchiuso tra virgolette (singole: 'uno', o doppie: "uno"). N.B. Le virgolette devono essere entrambe singole o entrambe doppie.
- Un attributo non può essere usato due volte all'interno dello stesso elemento.
- Un elemento può contenere più di un attributo.

`<romanzo file="Nicola.doc"> ...</romanzo>`

`<capitolo N="1">Capitolo primo</capitolo>`

Attributi - Regole

- I nomi degli attributi sono **case-sensitive** (c'è differenza tra maiuscole e minuscole).
- Lo stesso tag non può contenere due attributi con lo stesso nome.
- Non sono ammessi attributi senza valore (solo nome, es. `<section id="">`).
- Il valore degli attributi deve essere specificato **tra virgolette semplici o doppie**.
- Il valore può contenere **riferimenti ad entità** (es. ```).
- Il valore non può contenere markup, sezioni CDATA (character DATA) o virgolette uguali a quelle iniziali.

Attributi - Sintassi

- **Sintassi di base** da usare all'interno dei tag di apertura:

<nome attributo="valore">

- Una **lista di attributi** si ottiene elencando più attributi separati da uno o più spazi:

<nome att1="v1" att2="v2">

- Per includere **virgolette** nel valore, è necessario usare un tipo diverso da quello usato per delimitare il valore stesso:

<nome att1=' "virgolette" '>

- Si possono includere **riferimenti a entità** nel valore:

<nome att1="" salve " ">

Entità

- Le entità sono frammenti di documento memorizzati separatamente e richiamabili all'interno del documento.
- Esse permettono di riutilizzare lo stesso frammento in molte posizioni garantendo sempre l'esatta corrispondenza dei dati, e permettendo una loro modifica semplificata.

<testo>Oggi **è una bella giornata. </testo>**
(= Oggi **è** una bella giornata)

[Lista entità](#)

PCDATA (Parsed Character DATA)

- È il testo che si trova tra il tag di apertura e il tag di chiusura di un elemento (il contenuto vero e proprio del documento)
- Esso corrisponde alle parole, gli spazi e la punteggiatura che costituiscono il testo.
- Viene anche detto **PCDATA (Parsed Character DATA)** perché a differenza dei CDATA (*character data*, il cui contenuto testuale non viene analizzato) quello degli elementi è soggetto ad azione di parsing (= analisi, per lo più per identificare le entità e sostituirle).

<code><codice> <![CDATA[</code>	<code><testo></code>
<code><libro><capitolo></capitolo></code>	Oggi &grave; una bella
<code></libro></code>	giornata.
<code>]]> </codice></code>	<code></testo></code>

Riassumendo

I documenti XML si possono considerare formati da cinque blocchi (building blocks):

- Commenti
- Elementi
- Attributi (con i loro valori)
- Entità
- CDATA
- PCDATA

