

HTML, CSS E XML

PREMESSA: CODIFICA DEL TESTO

- I file **HTML**, **XML** e **CSS** sono file di Testo, sono cioè file composti esclusivamente da caratteri.
- Nel mondo digitale i caratteri sono codificati assegnando ad ognuno di essi un numero secondo una convenzione ben definita, ma arbitraria.
- Per poter elaborare correttamente un file di testo il computer deve sapere che codice è stato usato nel comporlo.

CODIFICA

- Anche i calcolatori più potente hanno un limite, mentre gli insiemi di informazioni possono essere limitati o illimitati
- La codifica è l'operazione che consente trasformare le informazioni in dati numerici che calcolatori elettronici possono leggere ed elaborare dati.
- Un bit può assumere solo due valori (0 e 1)
- Per rappresentare insiemi costituiti da più di due stati/simboli si usano serie di bit

CODIFICA

- Una stringa di bit e' costituita da un certo numero di bit (normalmente 8 o multipli di 8) ed e' comunemente detta parola (*word*)
- Con n bit si possono rappresentare 2^n valori diversi e quindi si possono rappresentare 2^n informazioni diverse
- La lunghezza della parola, quindi, definisce quante informazioni possono essere codificate

CODIFICA DEL TESTO

- Un testo e' una sequenza di caratteri alfabetici, separatori e caratteri speciali
- Ad ogni carattere e' associata una diversa configurazione di bit.
- Esempio: 21 lettere dell'alfabeto + 10 numeri + 10 punteggiatura = 41 simboli
 - $2^5 = 32$ combinazioni
 - $2^6 = 64$ combinazioni → ok
- Possiamo usare una codifica a 6 bit.

STANDARD ASCII

- ASCII significa *American Standard Code for Information Interchange*
- La codifica ASCII prevede l'utilizzo di 128 caratteri diversi
- Ogni carattere e' associato ad un numero da 0 a 127 cioè tutti i numeri rappresentabili con 7 bit.
- La codifica ASCII estesa prevede 256 simboli e 8 bit (*1byte*) per ogni carattere
- Quindi un testo di 1000 caratteri richiede 1Kbyte per essere rappresentato

STANDARD ASCII E HTML

- Per rappresentare i caratteri non codificabili utilizzando 7 bit si è introdotta una codifica particolare detta **entity**.
- Un entity è così composta:

&nnnn;

- Quando il parser incontra una parola che inizia con **&** legge i successivi caratteri fino ad incontrare **;** e tenta di interpretare il tutto come un carattere secondo la tabella di codici definita dal W3C.

STANDARD UNICODE

- Lo standard ISO10646/Unicode si basa su una codifica a 32 bit che consente oltre due miliardi di possibili caratteri
- **UTF-8** usa 7 bit per carattere per codificare i primi 128 caratteri (da 0 a 127) corrispondenti all'ASCII standard. Se il decodificatore incontra un codice superiore a 127 per decodificare il carattere viene utilizzato anche il byte successivo. In questa maniera vengono utilizzati da uno a quattro byte per rappresentare un carattere (da 8 a 32 bit) e i caratteri rappresentabili sono circa 270 milioni
- **UTF-16** è una codifica simile ma la base di partenza è 16 bit.

HTML

STANDARD UNICODE

- **HTML** è l'acronimo di **H**yper**T**ext **M**arkup **L**anguage ("Linguaggio a marcatori per gli Iper testi").
- Non è un linguaggio di programmazione non ha, cioè, meccanismi che consentono di prendere delle decisioni ("in questa situazione fai questo, in quest'altra fai quest'altro"), e non è in grado di compiere delle iterazioni ("ripeti questa cosa, finché non succede questo"), né ha altri costrutti propri della programmazione.
- Si tratta invece di un linguaggio di contrassegno (o 'di marcatura'), che permette di indicare come disporre gli elementi all'interno di una pagina: le indicazioni vengono date attraverso degli appositi marcatori, detti "**tag**".

I MARCATORI (TAG)

- I **tag** vanno inseriti tra parentesi uncinate: `<TAG>`
- La chiusura del tag viene indicata con una barra: `</TAG>`
- Il contenuto che il tag modifica va inserito tra l'apertura e la chiusura del tag medesimo:
Questa ``parola`` è in grassetto.
- che nel rendering verrà reso:
Questa **parola** è in grassetto.
- Alcuni tag non hanno (o possono non avere) contenuto (**empty tag**). Ad esempio l'interruzione di linea la indico così:

`
`

GLI ATTRIBUTI

- Le caratteristiche e i valori degli attributi dei tag HTML sono definiti nei documenti predefiniti che

```
<TAG attributo_1=
```

- Alcuni attributi (come class, id, ecc.), altri sono

```
<IMG width="20" height="20" src="miaImmagine.gif" alt="alt" />
```

- Una caratteristica importante del codice HTML è che i tag possono essere annidati l'uno dentro l'altro.
- È quindi opportuno usare l'indentazione. Grazie ad essa il codice HTML risulta più leggibile.

```
<TAG1 attributo = "valore">
  contenuto 1
  <TAG2>
    contenuto 2
  </TAG2>
</TAG1>
```

ENTITY

- Per rappresentare i caratteri non codificabili con lo standard ASCII si è introdotta una codifica particolare detta **entity**.
- Un entity è così composta:

&nnnn;

- Quando il parser HTML incontra una parola che inizia con **&** legge i successivi caratteri fino ad incontrare **;** e tenta di interpretare il tutto come un carattere secondo la tabella di codici definita dal W3C.

COMMENTI

- Un strategia importante, per rendere il nostro codice più leggibile è quella di inserire dei "commenti" nei punti più significativi:
- Un commento è un'indicazione significativa per il webmaster, ma invisibile al browser. Inserendo i commenti in punti specifici del documento ci permette di mantenere l'orientamento anche in file molto complessi e lunghi.
- La sintassi è la seguente:

```
<!-- questo è un commento -->
```

IL W3C

- L'organizzazione che si occupa di standardizzare la sintassi del linguaggio HTML (il W3C: [World Wide Web Consortium](http://www.w3.org/)).
- Ha rilasciato diverse versioni di questo linguaggio (HTML 2.0, HTML 3.2, HTML 4.0....);
- Allo stato attuale abbiamo a che fare con 3 versioni:
 - HTML 4.01 (24/12/1999)
 - XHTML 1.0 (01/08/2002)
 - HTML 5 (Bozza di lavoro: 19/10/2010)

CONTROLLO SINTASSI

- Il linguaggio HTML, pur essendo dotato di una sua sintassi precisa e codificata, non presuppone un controllo sintattico rigido.
- Per ragioni storiche (e strategiche) il fine principale di un browser è quella di fare vedere comunque qualcosa all'utente non formalizzandosi sul fatto che una pagina sia ben formata o meno.
- Se vi dimenticate di chiudere un tag, non verranno prodotti dei messaggi di errore; se non rispettate la sintassi probabilmente non otterrete la visualizzazione della pagina che desiderate, ma nient'altro.

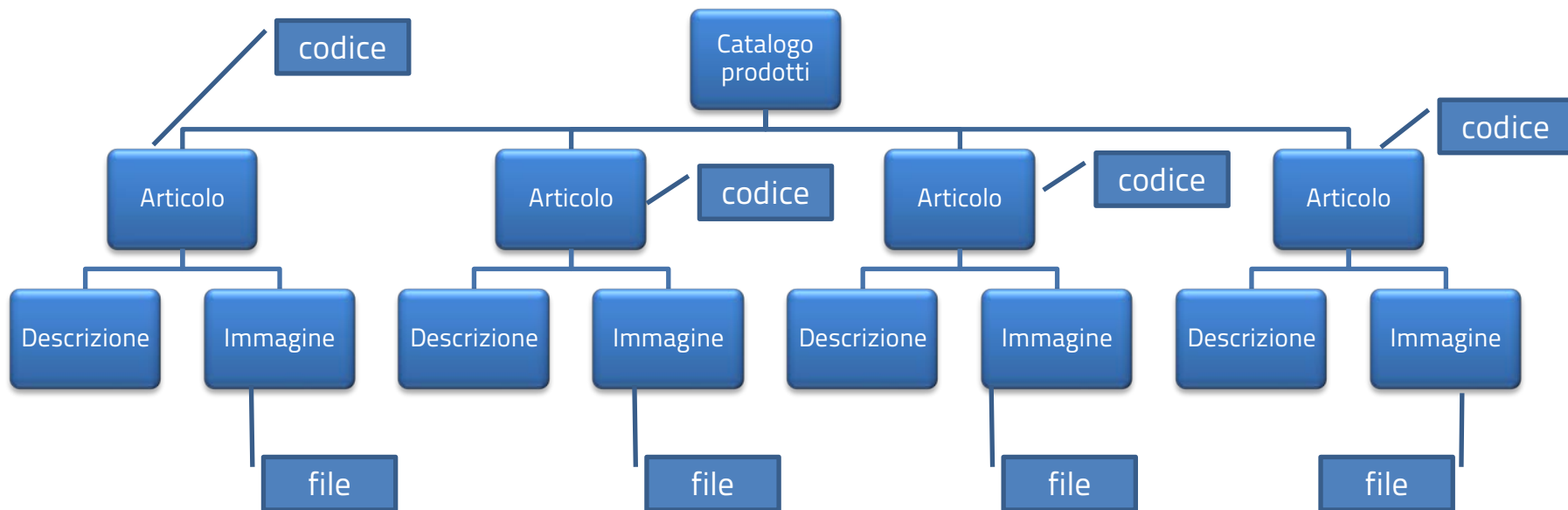
DEFINIZIONE DI XML

- Nel dicembre '97 il W3C rilascia le specifiche di XML come Proposed Recommendation.
- Gli obiettivi iniziali della nascita di XML erano rivolti alla soluzione di un problema di standard per il Web, ma ben presto ci si accorse che XML non era limitato al solo contesto Web.
- Esso risulta essere abbastanza generale per poter essere utilizzato nei più disparati contesti: dalla definizione della struttura di documenti allo scambio di informazioni tra sistemi diversi, dalla rappresentazione di immagini alla definizione di formati di dati.

DEFINIZIONE DI XML

- XML è un meta-linguaggio per definire la struttura di documenti e dati.
- Un documento XML è un file di testo che contiene una serie di **tag**, **attributi** e **testo** secondo regole sintattiche ben definite.
- Un documento XML è intrinsecamente caratterizzato da una struttura gerarchica. Esso è composto da componenti denominati **elementi**. Ciascun elemento rappresenta un componente logico del documento e può contenere altri elementi (sottoelementi) o del testo.
- Gli elementi possono avere associate altre informazioni che ne descrivono le proprietà. Queste informazioni sono chiamate attributi.
- L'organizzazione degli elementi segue un ordine gerarchico o arboreo che prevede un elemento principale, chiamato root element o semplicemente root o radice.

DEFINIZIONE DI XML



```
<?xml version="1.0" ?>
<catalogo>
  <articolo codice="Codice del primo articolo">
    <descrizione>
      Blocco di testo della descrizione.
    </descrizione >
    <immagine file="immagine1.jpg"></immagine>
  </articolo>
  <articolo codice="Codice del secondo articolo">
    < descrizione >
      Blocco di testo della descrizione.
    </descrizione>
  </articolo>
  <articolo codice="Codice del terzo articolo">
    <descrizione>
      Blocco di testo della descrizione.
    </descrizione>
  </articolo>
</catalogo>
```

XML È UN METALINGUAGGIO

- A differenza dell'HTML in cui i tag sono predefiniti, XML ci lascia liberi di definire i tag che vogliamo.
- Per specificare un attributo per un elemento inseriamo il nome dell'attributo con il relativo valore all'interno del tag di apertura dell'elemento.
- L'organizzazione gerarchica degli elementi viene rappresentata in XML tramite il loro annidamento.
- Alcuni elementi possono essere vuoti, cioè possono essere privi di contenuto testuale. A differenza di quanto avviene per l'HTML, che consente l'utilizzo di elementi senza tag di chiusura, XML prevede che vengano sempre specificati i tag di apertura e chiusura.
- Un Dtd è un documento che descrive i tag utilizzabili in un documento XML, la loro reciproca relazione nei confronti della struttura del documento e altre informazioni sugli attributi di ciascun tag.

REGOLE DI BASE

- XHTML è la riformulazione di HTML come applicazione XML. Ciò significa essenzialmente una cosa: un documento XHTML deve essere valido e ben formato.
- Niente nuovi tag, attributi o metodi, cambiano le regole sintattiche.
- Le versioni di XHTML attualmente disponibili e pubblicate come raccomandazioni dal W3C sono tre.
 - XHTML 1.0 (26 gennaio 2000)
 - DTD Strict
 - DTD Transitional
 - DTD Frameset
 - XHTML Basic (Dispositivi mobili)
 - XHTML 1.1

XHTML

- Un documento deve essere convalidato rispetto ad una delle tre DTD XHTML del W3C.
- Presenza di un elemento radice, corretto annidamento degli elementi, chiusura obbligatoria dei tag vuoti, etc.
- Ogni documento XML deve contenere un elemento radice. Si tratta dell'elemento che contiene al suo interno tutti gli altri. In un documento XHTML l'elemento radice deve essere `<html>`.
- L'elemento radice `<html>` deve contenere la dichiarazione di un namespace XML (spazio dei nomi) tramite l'attributo `xmlns`. Il namespace usato deve essere "<http://www.w3.org/1999/xhtml>".
- In un documento XHTML l'elemento radice deve essere preceduto da un elemento `<!DOCTYPE>`. All'interno di questo elemento è necessario specificare la DTD di riferimento e il suo URI.

XHTML

- Elementi devono correttamente annidati:

~~<i>un test</i>~~

<i>un test</i>

- Elementi e attributi devono essere in minuscolo
- Tutti gli elementi non vuoti devono essere chiusi
- I valori degli attributi devono essere posti tra virgolette:

~~~~

XHTML

- Ogni attributo deve avere un valore:

~~<option selected>test</option>~~

<option selected="selected">test</option>

- Gli elementi vuoti devono terminare con />

~~~~



- Per identificare un elemento si deve usare l'attributo id e non name.

## DTD STRICT

- È la DTD più rigida, centrata esclusivamente sulla struttura del documento.
- Elimina diversi elementi ed esclude tutti gli attributi che definiscono la presentazione. Per questo scopo vanno usati i **CSS**.
- Tag non supportati: <applet>, <basefont>, <center>, <dir>, <font>, <frame>, <frameset>, <iframe>, <isindex>, <menu>, <noframes>, <s>, <strike>, <u>

# DTD TRASITIONAL

- Supporta tutti gli elementi e gli attributi di presentazione di HTML 4.0, anche quelli ritenuti sconsigliati.
- Garantisce la massima compatibilità con i vecchi browser.
- Consente anche sintassi fortemente sconsigliate che potrebbero non essere più supportate in futuro.

# OTO FRAMESET

- È identica alla Transitional, ma va usata quando si utilizzano i frame.
- L'unica differenza è in pratica la sostituzione del tag `<body>` con `<frameset>` nella pagina principale

# STRUTTURA DELLA PAGINA

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

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

```
<head>
```

```
  <title>La mia prima pagina XHTML</title>
```

```
</head>
```

```
<body>
```

```
  <h1>Benvenuto!</h1>
```

```
  <p>Questo &egrave; il mondo di XHTML!</p>
```

```
</body>
```

```
</html>
```

# PROLOGO

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

- Dichiarazione XML (non obbligatoria):
  - **version** (il solo valore possibile è "1.0", in quanto non esistono altre versioni del linguaggio).
  - **encoding**: Serve a specificare la codifica del testo in cui è scritto il documento.
- Per massima compatibilità si può omettere la dichiarazione XML e usare il tag meta per indicare la codifica della pagina:

```
<meta http-equiv="Content-Type"  
  content="text/html; charset=UTF-8" />
```

# PROLOGO

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

- Un FPI (Identificatore Formale Pubblico) riferito ad una delle tre DTD XHTML
- L'URI della DTD
- Essa, dunque, ha lo scopo di stabilire a quale delle tre DTD XHTML intendiamo conformarci e di dire al browser dove essa è collocata.
- Il DOCTYPE non ha alcun effetto sulla presentazione della pagina. Serve solo al validatore per stabilire le regole della convalida.

# ELEMENTO RADICE (ROOT)

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

- <HTML> è obbligatorio
- L'elemento <html> può assumere questi attributi:
  - dir Determina la direzione del testo
  - lang Specifica il linguaggio di base dell'elemento quando è interpretato come HTML
  - xml:lang Specifica il linguaggio di base dell'elemento quando è interpretato come XML
  - xmlns Specifica il namespace predefinito per XHTML
- L'unico attributo obbligatorio è xmlns. Il W3C, come visto, specifica anche il valore obbligatorio di tale attributo: "http://www.w3.org/1999/xhtml".



# HEAD

```
<head>  
  <title>La mia prima pagina XHTML</title>  
</head>
```

- La funzione principale della sezione <head> è quella di contenere informazioni che non vengono direttamente visualizzate nella pagina, ma che sono comunque di grande rilievo. Ecco l'elenco degli elementi che possono apparire nella testata:
  - <base> Usato per definire l'URL di base della pagina. Utilissimo per la risoluzione dei link relativi.
  - <link> Contiene informazioni su documenti esterni collegati. Usato soprattutto per i CSS.
  - <meta> Specifica informazioni di vario tipo sul documento.
  - <noscript> Usato per visualizzazioni alternative nei browser che non supportano gli script.
  - <object> Racchiude un oggetto.
  - <script> Contiene script di programmazione .
  - <style> Definisce le regole di formattazione per il documento corrente
  - <title> Specifica il titolo del documento che compare nella barra del titolo del browser

# BODY

```
<body>
  <h1>Benvenuto!</h1>
  <p>Questo &egrave; il mondo di XHTML!</p>
</body>
```

- Il corpo del documento è la sezione in cui si sviluppa il contenuto. È racchiusa, come in HTML, tra i tag `<body>...</body>`.
- Gli elementi che possono comparire all'interno del corpo sono in genere suddivisi in due categorie:
  - **elementi blocco** ed gli elementi blocco sono quelli che definiscono la struttura del documento. Possono contenere altri elementi blocco, elementi inline o testo. Quando sono inseriti danno origine ad una nuova riga nel flusso del documento.
  - **elementi inline**: quando sono inseriti non danno origine a una nuova riga e possono contenere solo dati (essenzialmente testo) o altri elementi inline.

## ELEMENTI BLOCCO

- Il primo elemento della gerarchia dovrebbe essere <div>, che definisce in pratica una sezione del documento. Al suo interno trovano posto gli altri elementi. Bisogna evitare annidamenti errati, che i browser fanno passare senza problemi, ma che il validatore segnala impietosamente in quanto violano le regole delle DTD.
- Esempio:

```
<p><div>Qui inserisco il mio testo</div></p>  
<div><p>Qui inserisco il mio testo</p></div>
```

# ELEMENTI BLOCCO

| Elemento     | Descrizione                                                                                                                   | DTD |
|--------------|-------------------------------------------------------------------------------------------------------------------------------|-----|
| <address>    | Definisce un blocco di testo destinato a indirizzi, firme, indicazioni sull'autore. Non può contenere altri elementi blocco.  | STF |
| <blockquote> | Usato per riportare citazioni da altri documenti. Il testo inserito viene indentato. Può contenere tutti gli elementi blocco. | STF |
| <center>     | <b>Centra il testo che racchiude. Sconsigliato in HTML 4.0.</b>                                                               | TF  |
| <dir>        | <b>Crea una lista di tipo directory. Sconsigliato in HTML 4.0</b>                                                             | TF  |
| <div>        | Definisce un blocco di contenuto generico o una sezione del documento. Può contenere tutti gli elementi blocco.               | STF |
| <dl>         | Crea una lista di definizione. Può contenere solo gli elementi <dt> e<dd>.                                                    | STF |
| <fieldset>   | Usato per raggruppare campi di un form.                                                                                       | STF |
| <form>       | Definisce un form. Può contenere i classici elementi dei form ma anche elementi blocco.                                       | STF |
| <h1>..<h6>   | Definiscono titoli e sottotitoli. Non possono contenere altri elementi blocco.                                                | STF |
| <hr>         | Inserisce una linea divisoria orizzontale. E' un elemento vuoto.                                                              | STF |
| <isindex>    | <b>Inserisce un elemento simile alle caselle di testo. Sconsigliato in HTML 4.0</b>                                           | TF  |
| <menu>       | <b>Definisce una lista di tipo menu. Sconsigliato in HTML 4.0</b>                                                             | TF  |
| <noframes>   | Inserisce contenuto alternativo per i browser che non supportano i frames.                                                    | STF |

|                               |                                                                                                                                             |     |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------|-----|
| <code>&lt;noscript&gt;</code> | Inserisce contenuto alternativo per i browser che non supportano gli script.                                                                | STF |
| <code>&lt;ol&gt;</code>       | Lista ordinata. può contenere solo l'elemento <code>&lt;li&gt;</code>                                                                       | STF |
| <code>&lt;p&gt;</code>        | Definisce un paragrafo. Non può contenere altri elementi blocco, ma solo testo o elementi inline.                                           | STF |
| <code>&lt;pre&gt;</code>      | Definisce testo preformattato che mantiene le impostazioni dello spazio bianco.                                                             | STF |
| <code>&lt;table&gt;</code>    | Definisce una tabella per l'inserimento di dati tabulari.                                                                                   | STF |
| <code>&lt;dd&gt;</code>       | Descrizione di un termine in una lista di definizione.                                                                                      | STF |
| <code>&lt;dt&gt;</code>       | Definizione di un termine in una lista di definizione.                                                                                      | STF |
| <code>&lt;frameset&gt;</code> | Definisce un frameset.                                                                                                                      | F   |
| <code>&lt;li&gt;</code>       | Elemento di una lista ordinata o non ordinata.                                                                                              | STF |
| <code>&lt;tbody&gt;</code>    | Definisce il corpo di una tabella. Con <code>&lt;thead&gt;</code> e <code>&lt;tfoot&gt;</code> serve a raggruppare le righe di una tabella. | STF |
| <code>&lt;td&gt;</code>       | Cella di tabella.                                                                                                                           | STF |
| <code>&lt;tfoot&gt;</code>    | Definisce il "piede" di una tabella.                                                                                                        | STF |
| <code>&lt;th&gt;</code>       | Intestazione di cella.                                                                                                                      | STF |
| <code>&lt;thead&gt;</code>    | Definisce la testata di una tabella.                                                                                                        | STF |
| <code>&lt;tr&gt;</code>       | Riga di tabella.                                                                                                                            | STF |

# ELEMENTI INLINE

- Quando sono inseriti non danno origine a una nuova riga e possono contenere solo testo o altri elementi inline.

```
<p>Questo tasto è<b>grassetto</b></p>
```

- La parte delimitata dai tag `<b>...</b>` non sarà posta su una nuova riga.
- Esempi come questo:

```
<b><p>Testo in grassetto</p></b>
```

- sono tollerati dai browser, ma non reggono al giudizio della validazione in quanto un elemento inline non può contenerne uno di tipo blocco.

# ATTRIBUTI DI BODY

- Gli attributi per il testo, i link, il colore di sfondo e i margini dell'elemento <body> sono espressamente vietati solo nella DTD Strict, ma erano già considerati sconsigliati in HTML 4.0.

~~alink~~

~~background~~

~~bgcolor~~

~~link~~

~~text~~

~~vlink~~



OBSOLETI!!

# VALIDARE LE PAGINE

- L'utilizzo delle dichiarazioni doctype consente l'uso dei validatori, applicazioni web che consentono di verificare se le pagine costruite soddisfano lo standard (DTD) dichiarato.
- Si può usare il [validatore del W3C](#)
- O il validatore alternativo del sito [htmlhelp.com](http://htmlhelp.com).
- Una volta validati i siti possono esporre l'icona che certifica la validazione.





# CSS

# I FOGLI DI STILE

- HTML serve informare il browser di quali sono le componenti necessarie a mostrare un documento e ad articolare il documento in blocchi semantici.
- I fogli di stile (**Cascading Style Sheets**) definiscono come i vari elementi che compongono un documento verranno resi su un media specifico (schermo, stampante, dispositivo mobile).

# INSERIMENTO

- Un foglio di stile può essere esterno e interno:
  - È esterno un foglio di stile definito in un file separato dal documento.
  - Un foglio di stile si dice interno quando il suo codice è compreso in quello del documento.
- Un foglio esterno si carica:
  - Utilizzando l'elemento <LINK>.
  - Usando @import.
- Un foglio interno può essere compilato
  - utilizzando l'elemento <style>
  - Utilizzando l'attributo style di un singolo elemento

# FOGLI COLLEGATI

- Uso dell'elemento <LINK>:
  - La dichiarazione va sempre collocata all'interno della sezione <HEAD> del documento (X)HTML:

```
<html>
```

```
<head>
```

```
<title>Inserire i fogli di stile in un documento</title>
```

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

```
</head>
```

```
<body>
```

- L'elemento <link> presenta una serie di attributi di cui è importante spiegare significato e funzione:

| Attributo | Descrizione                                                                                                                                                                          |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| rel       | descrive il tipo di relazione tra il documento e il file collegato. È <b>obbligatorio</b> . Per i CSS due sono i valori possibili: <b>stylesheet</b> e <b>alternate stylesheet</b> . |
| href      | serve a definire l'URL assoluto o relativo del foglio di stile. È <b>obbligatorio</b>                                                                                                |
| type      | identifica il tipo di dati da collegare. Per i CSS l'unico valore possibile è <b>text/css</b> . L'attributo è <b>obbligatorio</b>                                                    |
| media     | con questo attributo si identifica il supporto (schermo, stampa, etc) cui applicare un particolare foglio di stile. Attributo <b>opzionale</b> .                                     |

## FOGLIO COLLEGATI

- Un altro modo per caricare CSS esterni è usare la direttiva `@import` all'interno dell'elemento `<style>`:  

```
<style>  
  @import url(stile.css);  
</style>
```
- Questo sistema è utile per fare in modo che solo i browser più recenti carichino il foglio di stile, garantendo così che lo interpretino correttamente e forzando i browser obsoleti (come Netscape 4. ad esempio) a rendere la pagina senza foglio di stile.

# FOGLIO INCORPORATI

- I fogli incorporati sono quelli inseriti direttamente nel documento (X)HTML tramite l'elemento `<style>`. Anche in questo caso la dichiarazione va posta all'interno della sezione `<head>`:

```
<html>
  <head>
    <title>Inserire i fogli di stile in un documento</title>
    <style type="text/css">
      body {
        background: #FFFFCC;
      }
    </style>
  </head>
  <body>
    ...
```

- `<style>` può avere due attributi:
  - `type` (obbligatorio)
  - `media` (opzionale)

# FOGLIO IN LINEA

- L'ultimo modo per formattare un elemento con un foglio di stile consiste nell'uso dell'attributo 'style'.
- Esso fa parte della collezione di attributi (X)HTML definita Common: si tratta di quegli attributi applicabili a tutti gli elementi.
- La dichiarazione avviene a livello dei singoli tag contenuti nella pagina e per questo si parla di fogli di stile in linea. La sintassi generica è la seguente:

```
<elemento style="regole_di_stile">
```

# REGOLE

- Un foglio di stile è costituito da una serie di regole che stabiliscono come un elemento (identificato da un selettore) viene reso su un media.



- Esempio:

```
p{font: 12px Verdana, arial;}
```



# SINTASSI ABBREVIATA

- Ogni elemento presenta sui suoi quattro lati un certo margine rispetto a quelli adiacenti.
  - margin-top
  - margin-right
  - margin-bottom
  - margin-left
- La regola sarebbe questa:

```
div { margin-top: 10px;  
      margin-right: 5px;  
      margin-bottom: 10px;  
      margin-left: 5px;  
}
```
- Sintassi abbreviata:

```
div {margin: 10px 5px 10px 5px;}
```

# SELETTORI

# ELEMENTI

- È il più semplice dei selettori. È costituito da uno qualunque degli elementi di (X)HTML.

CSS  
1.0

```
h1 {color: #000000;}
```

```
p {background: white; font: 12px Verdana, arial, sans-serif;}
```

```
table {width: 200px;}
```

- È possibile nei CSS raggruppare diversi elementi al fine di semplificare il codice.

CSS  
1.0

```
h1 {background: white;}
```

```
h2 {background: white;}
```

```
h3 {background: white;}
```

```
h1, h2, h3 {background: white;}
```

CSS  
2.1

```
* { color: black; }
```

# ELEMENTI

- Elementi che nella struttura ad albero di un documento siano discendenti di un altro elemento specificato nella regola.

CSS  
1.0

```
div p {color: black;}  
p strong {color: red;}
```

- Nel primo esempio verranno selezionati tutti i paragrafi (<p>) discendenti di elementi <div>. Nel secondo tutti gli elementi <strong> che si trovino all'interno di un paragrafo.

- Elementi che nella struttura ad albero di un documento siano i figli diretti di un elemento.

CSS  
2.1

```
body > p {color: black;}  
<body>  
  <p>Primo paragrafo</p>  
  <div>  
    <p>Secondo paragrafo</p>  
  </div>  
</body>
```

IE7+

# ELEMENTI

- Elementi che nella struttura ad albero di un documento siano discendenti di un altro elemento specificato nella regola.

CSS  
1.0

```
div p {color: black;}  
p strong {color: red;}
```

- Nel primo esempio verranno selezionati tutti i paragrafi (<p>) discendenti di elementi <div>. Nel secondo tutti gli elementi <strong> che si trovino all'interno di un paragrafo.

- Elementi che nella struttura ad albero di un documento siano i figli diretti di un elemento.

CSS  
2.1

```
body > p {color: black;}  
<body>  
  <p>Primo paragrafo</p>  
  <div>  
    <p>Secondo paragrafo</p>  
  </div>  
</body>
```

IE7+

# ELEMENTI

- Elementi che nel codice del documento siano immediatamente vicini (adiacenti) ad un altro.

CSS  
2.1

```
h1 + p {color: red;}  
<h1>Titolo</h1>  
<p>Primo paragrafo</p>  
<p>Secondo paragrafo</p>
```

- In base a questa dichiarazione solo il primo dei due paragrafi avrà il testo rosso..

IE7+

# SELEZIONE DEGLI ELEMENTI IN BASE AI LORO ATTRIBUTI



- Attributo semplice

CSS  
2.1

```
input [ id ] {background: red;}
```

- applicherà uno sfondo rosso a tutti gli elementi input per cui sia stato impostato un attributo id, a prescindere dal valore di id.

- Attributo con valore

CSS  
2.1

```
input [ id = "text" ] { background: red; }
```

- applicherà un sfondo rosso a tutti gli elementi input che abbiano come valore dell'attributo id "text".

- Attributo il cui valore contiene una stringa

CSS  
3.0

```
img [ alt*= "foto" ] {margin: 10px;}
```

- La regola applicherà un margine di 10px a tutte le immagini in cui l'attributo alt contiene la stringa "foto".

- Attributo il cui valore inizia con una stringa

CSS  
3.0

```
img [ alt |= "figura" ] {margin: 10px;}
```

- selezionerà tutte le immagini in cui l'attributo alt inizia con la stringa "figura".

# CLASSI E ID

- In questa pagina abbiamo assegnato al paragrafo l'attributo `class="testorosso"`:  
`<p class="testorosso">....</p>`
- Possiamo ora creare un regola e assegnargli il nome testorosso:  

```
.testorosso {  
  font: 12px arial, Helvetica, sans-serif;  
  color: #FF0000;  
}
```
- In un documento potrò avere senza problemi questa situazione:  
`<p class="testorosso">....</p>`  
`<div class="testorosso">....</div>`  
`<table class="testorosso">...</table>`  
`<p class="testorosso">....</p>`
- E l'elemento seguirà la regola definita nella classe testorosso



# CLASSI E ID

- Per definire una classe si usa far precedere il nome da un semplice punto:

```
.testorosso {
```

```
font: 12px arial, Helvetica, sans-serif;  
color: #FF0000;  
}
```

- Se scriviamo:

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

lo stile verrà applicato solo ai paragrafi che presentino l'attributo class="testorosso".

- Sono possibili dichiarazioni di classi multiple:

```
p.testorosso.grassetto {color:red; font-weight:bold;}
```

Questa regola applicherà gli stili impostati a tutti gli elementi in cui siano presenti (in qualunque ordine) i nomi delle classi definiti nel selettore.

IE7+

# CLASSI E ID

- La sintassi di un selettore ID è semplicissima. Basta far precedere il nome dal simbolo di cancelletto #:



```
#titolo {  
  color: blue;  
}
```

- assegniamo il colore blue all'elemento che presenti questa definizione:

```
<h1 id="titolo">...</h1>
```

- Come per le classi è possibile usare una sintassi con elemento:



```
p#nome_id {  
  color: red;  
}
```

- Ma non ha senso perché l'id per sua natura dovrebbe essere unico.

# PSEUDO-CLASSI

- Una pseudo-classe non definisce un elemento ma un particolare stato di quest'ultimo. In buona sostanza imposta uno stile per un elemento al verificarsi di certe condizioni.
- A livello sintattico le pseudo-classi non possono essere mai dichiarate da sole, ma per la loro stessa natura devono sempre appoggiarsi ad un selettore.

```
a:link {color: blue;}
```

- La regola vuol dire: i collegamenti ipertestuali (<a>) che non siano stati visitati (:link) avranno il colore blue.

# PSEUDO-CLASSI

- :first-child

CSS  
2.1

– Seleziona e formatta un elemento che si trovi ad essere il primo elemento figlio di un altro elemento.

IE  
7+

- :link

CSS  
1.0

– Si applica solo all'elemento (X)HTML <a> che abbia anche l'attributo href. Definisce lo stile per questo elemento quando il collegamento punta ad un sito o ad una pagina non ancora visitati.

- :hover

CSS  
2.1

– Definisce lo stile per questo elemento quando il puntatore è sopra all'elemento.

IE  
7+

- :visited

CSS  
1.0

– Si applica solo all'elemento (X)HTML <a> che abbia anche l'attributo href. Definisce lo stile per questo elemento quando il collegamento punta ad un sito o ad una pagina già visitata.

# PSEUDO-CLASSI

- :first-letter

CSS  
1.0

– Imposta lo stile della prima lettera di un elemento contenente del testo.

IE  
7+

- :first-line

CSS  
1.0

– Imposta lo stile della prima riga di un elemento contenente del testo.

IE  
7+

- :before

CSS  
2.1

```
blockquote:before {content:"Nota"}
```

IE  
8+

- :after

CSS  
2.1

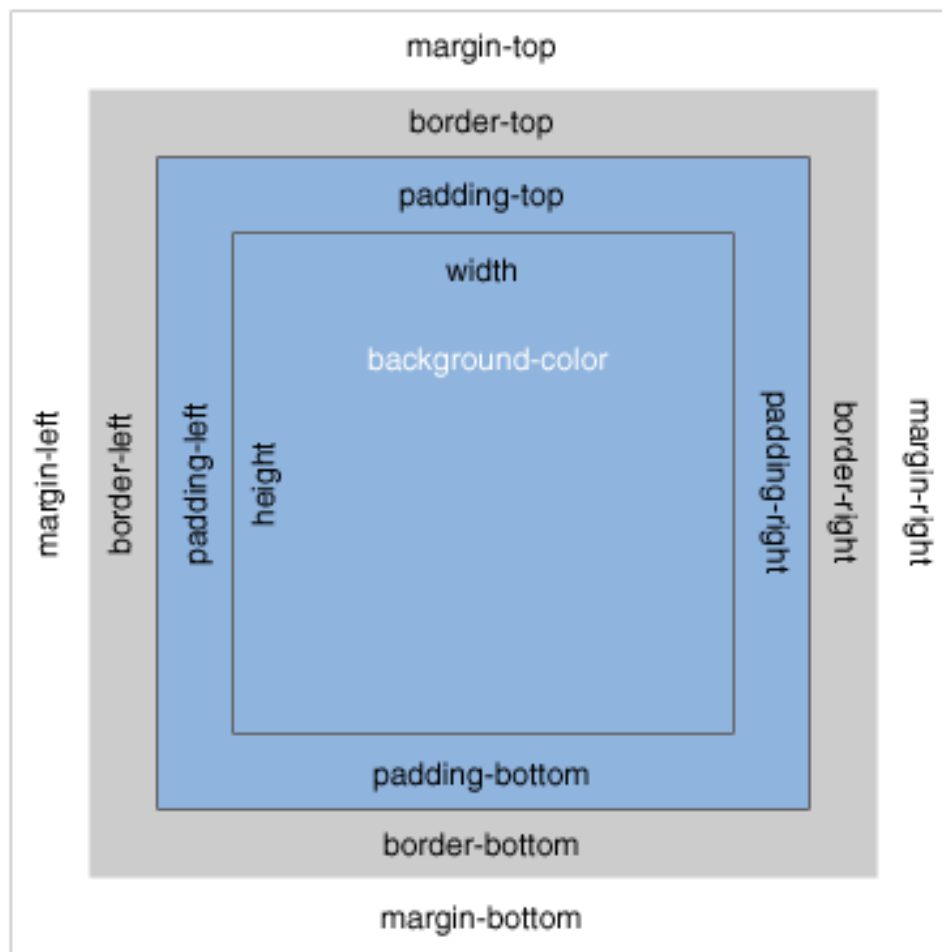
```
blockquote:after {content:"Nota"}
```

IE  
8+

# DETTAGLI

- <http://kimblim.dk/css-tests/selectors/>

# PROPRIETÀ DEI BLOCCHI



# BACKGROUND

- **background-color**

- Definisce il colore di sfondo di un elemento. Questa proprietà non è ereditata.

```
selettore {background-color: #FFF;}
```

```
selettore {background-color: transparent;}
```

- **background-image**

- Definisce l'URL di un'immagine da usare come sfondo di un elemento. Questa proprietà non è ereditata.

```
selettore { background-image: url(valore); }
```

```
selettore { background-image: none }
```

- **background-repeat**

- Consente di definire la direzione in cui l'immagine di sfondo viene ripetuta.

```
selettore {background-repeat: valore;}
```

- Valori: repeat, repeat-x, repeat-y, no-repeat

- **background-attachment**

```
selettore {background-attachment: valore;}
```

- Valori: scroll, fixed.



# BACKGROUND

- **background-position**

- Definisce il punto in cui verrà piazzata un'immagine di sfondo.

```
selettore {background-position: valoreOriz | valoreVert;}
```

- Valori: valori in percentuale, valori espressi con unità di misura, parole chiave top, left, bottom, right.

- **background**

- Per essere valida, la dichiarazione non deve contenere necessariamente riferimenti a tutte le proprietà viste finora, ma deve contenere almeno la definizione del colore di sfondo.

```
selettore { background: background-color  
                background-image  
                background-repeat  
                background-attachment  
                background-position;  
            }
```