

## Il linguaggio HTML

Ombretta Gaggi  
Università di Padova

### Un po' di storia - 1

- **HTML** (HyperText Markup Language) è un linguaggio per la costruzione di ipertesti. È un linguaggio di markup: descrive in modo molto preciso al browser come deve apparire la pagina web.
  - 1992: prima versione del linguaggio HTML
  - ~ 1994: HTML 3.0 diventa uno standard
  - Dicembre 1997: Recommendation HTML 4.01 del **W3C**
- L'elasticità e la semplicità di HTML ha favorito lo svilupparsi di editor **WYSIWYG**. L'utilizzo di questi ultimi ha però portato a risultati graficamente molto accattivanti, ma di bassa qualità (tecnica del "costruisci adesso paga il prezzo più tardi"):
  - pagine non accessibili, spreco di banda, code forking

Tecnologie Web - 2



### La guerra dei browser

- HTML appartiene alla famiglia dei linguaggi **SGML** (non XML), ma è molto più semplice
  - appena introdotto era troppo povero, non permetteva frame né immagini
- Per accaparrarsi più utenti, i browser hanno cominciato ad inserire nel linguaggio nuovi tag proprietari
  - **img** (Netscape) vs **object** (MS Internet Explorer)
  - **blink** (Netscape): testo lampeggiante
  - **marquee** (Internet Explorer): testo scorrevole
- La guerra dei browser ha portato un arricchimento delle possibilità offerte ma anche grossi problemi di incompatibilità
- HTML 4.01 non risolve del tutto il problema

Tecnologie Web - 3



### Un po' di storia - 2

- 1998: il W3C lancia il **Web Standard Project (WaSP)** per spingere Netscape, Microsoft e le altre case produttrici di browser a supportare pienamente gli standard
- Successivamente il WaSP si pone anche l'obiettivo di far produrre codice valido agli editor
- 2000: **IE5** per Macintosh supporta abbastanza bene gli standard XHTML, CSS e XML
  - document switch
  - Text zoom
- 2001: campagna aggiornamento browser. Obiettivo: eliminare i vecchi, tipo Netscape 4, IE 4, etc.
  - <http://www.alistapart.com/articles/tohell/>

Tecnologie Web - 4



## Un po' di storia - 3

- ❑ Gennaio 2000: viene definito lo standard **XHTML 1.0** successivamente revisionato nel 2002
- ❑ Luglio 2006: **XHTML 2.0** Working draft
  - non retrocompatibile
- ❑ Luglio 2008: diventa raccomandazione **XHTML Basic 1.1**, una versione pensata per i dispositivi palmari e cellulari
- ❑ XHTML 1.1 è essenzialmente una riformulazione della prima versione, in cui i tag e gli attributi vengono divisi in moduli per facilitarne l'uso



## Problemi di HTML

- ❑ Crescita disordinata
  - incompatibilità
- ❑ Contenuto e aspetto non vengono considerati separatamente



## HTML5

- ❑ Creato in risposta a XHTML 2, che sembrava fare le stesse cose della versione precedente del linguaggio in modo diverso, mentre si voleva qualcosa più orientato alle applicazioni web
- ❑ **WHATWG**, *Web Hypertext Application Technology Working Group*, coordinati da **Ian Hickson** (2004). È un gruppo in grado di lavorare più velocemente
  - Web Forms 2.0
  - Web Apps 1.0
- ❑ **W3C HTML 5 Working Group**: parte dalle specifiche prodotte dal WHATWG



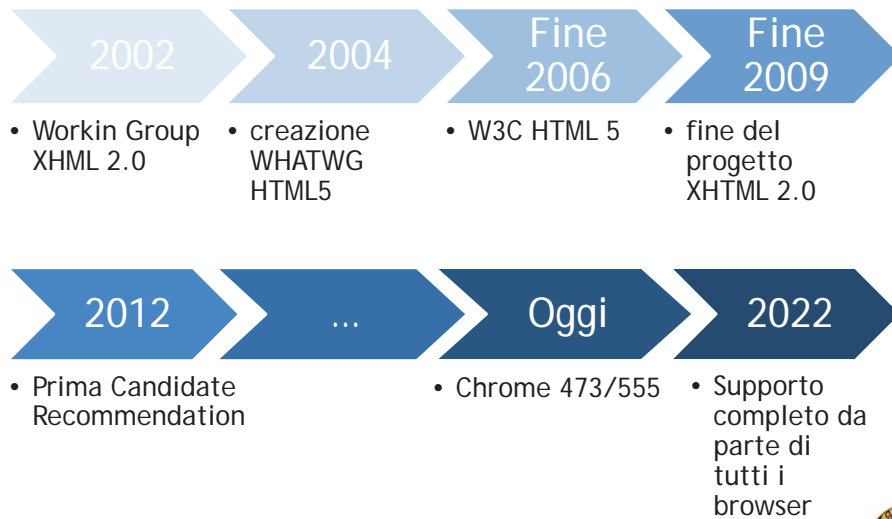
## HTML5

"In case of conflict, consider user over authors over implementers over specifiers over theoretical purity."

*HTML5 Working Group*



## Storia HTML5



Tecnologie Web - 9



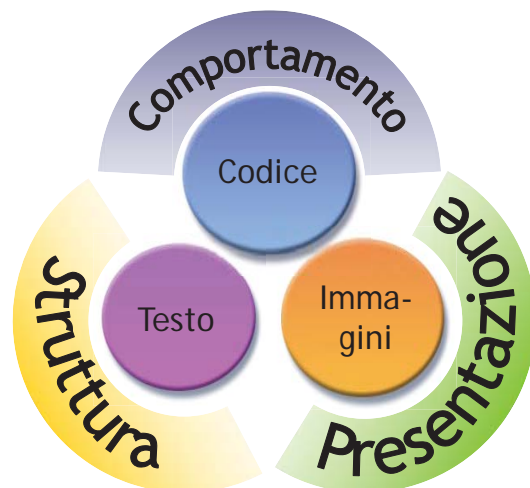
## Supporto dei browser



Tecnologie Web - 10



## Elementi di un sito web



Tecnologie Web - 11

J. Veen, 2001



## Problemi di HTML

- Crescita disordinata
  - incompatibilità
- Contenuto e aspetto non vengono considerati separatamente
  - Pagine XHTML + fogli di stile CSS
- Il numero notevole di pagine web presenti oggi rende difficile qualunque modifica al linguaggio HTML che non sia retrocompatibile

Tecnologie Web - 12



## XHTML vs HTML

- ❑ XHTML è l'evoluzione del linguaggio HTML. XHTML 1.0 è la versione successiva di HTML 4.01, quindi lo standard corrente
- ❑ XHTML è HTML riformulato come XML quindi è più coerente e aiuta lo sviluppo di codice valido. Questo elimina parte dei problemi di presentazione di HTML
- ❑ Essendo un linguaggio XML è interoperabile
- ❑ Elimina il problema del *code forking* perché supportato da diversi tipi di dispositivi
  - browser
  - browser per dispositivi mobili
  - screen reader
- ❑ I vecchi browser lo supportano abbastanza bene



## Inoltre... (ricordiamo)

- ❑ L'uso degli standard Web porta i seguenti vantaggi:
  - Compatibilità con i browser
  - Compatibilità con le future tecnologie
  - Controllo centralizzato della presentazione
  - Indipendenza dal dispositivo
  - Migliore posizionamento nei motori di ricerca
  - Pagine leggere
  - Accessibilità
  - Migliore posizionamento sul mercato come sviluppatore web



## Perché non usare gli standard?



## XHTML Strict vs XHTML transitional

- ❑ Esistono diverse versioni di XHTML
  - XHTML Strict
  - ~~XHTML Transitional~~
  - ~~XHTML Frameset~~
- ❑ XHTML Transitional è una forma transitoria creata per facilitare agli sviluppatori il passaggio ai nuovi standard
- ❑ XHTML Strict è la forma più pura che aiuta a produrre codice in cui struttura e presentazione sono fortemente separati
  - svantaggi: non sempre supportato bene dai vecchi browser



## La sintassi XHTML

- ❑ XHTML è un linguaggio XML quindi:
  - i tag e gli attributi sono case sensitive (tutto in minuscolo)
  - i tag devono sempre essere chiusi (anche se sono vuoti)
    - ❑ <br /> e non <br>
    - ❑ per compatibilità con i vecchi browser va usata la forma <p></p> per i tag non vuoti (anche se privi di contenuto) e <br /> per gli elementi vuoti
  - i tag devono essere aperti e chiusi nell'ordine corretto
  - l'ordine con cui si inseriscono gli attributi è irrilevante
  - i valori degli attributi vanno riportati tra "virgolette doppie"
  - tutti gli attributi devono avere un valore
  - un elemento in linea non può contenere un elemento di blocco

I browser cercano di visualizzare *al meglio* codice non valido, ma questo può portare ad interpretazioni arbitrarie



## HTML5

- ❑ Uno delle critiche mosse ad XHTML era la rigidità della sua sintassi
- ❑ HTML5 supporta sia la sintassi di tipo HTML che la sintassi di tipo XML
- ❑ Il problema del codice non valido, viene risolto istruendo i browser su come comportarsi in caso di codice non valido
  - definisce una gestione degli errori standard
  - obiettivo molto ambizioso e, allo stato attuale non raggiunto



utilizzare sempre codice valido



## Alcune semplificazioni

- |   |  |
|---|--|
| ❑ docType   | ❑ <!DOCTYPE html>                            |
| ❑ <meta http-equiv="Content-Type" content="text/html; charset=utf-8" /> | ❑ <meta charset="utf-8" >                    |
| ❑ <script type="text/javascript" src="script.js"></script>              | ❑ <script src="script.js"></script>          |
| ❑ <link type="text/css" rel="stylesheet" href="print.css" />            | ❑ <link rel="stylesheet" href="print.css" /> |



## Elementi che vengono ignorati

- ❑ I browser ignorano completamente:
  - le interruzioni di linea non identificate con <br/> e non contenute in un tag <pre>
  - tabulazioni e spazi multipli
  - tag <p> nidificati
  - tag sconosciuti
  - commenti
    - ❑ **ATTENZIONE:** all'interno di un commento non è possibile inserire la stringa "--" (doppi trattini)



## Struttura base di un documento XHTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="it"
    lang="it">
  <head>
    <title>la mia prima pagina web</title>
  </head>
  <body>
    <p>Ecco la mia prima pagina in html destinata
    al web.</p>
    <!-- commento HTML -->
  </body>
</html>
```

Specifica di riferimento

Intestazione

Contenuto della pagina Web

Tecnologie Web - 21



## Struttura base di un documento HTML5

```
<!DOCTYPE html>
<html lang="it">
  <head>
    <title>la mia prima pagina web</title>
  </head>
  <body>
    <p>Ecco la mia prima pagina in html destinata
    al web.</p>
    <!-- commento HTML -->
  </body>
</html>
```

DOCTYPE

Intestazione

Contenuto della pagina Web

Tecnologie Web - 22



## Prologo XML

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

- Il W3C raccomanda di usare il prologo XML (opzionale) per specificare
    - la versione XML
    - il tipo di codifica dei caratteri
- purtroppo però molti browser non lo gestiscono correttamente, causando visualizzazioni scorrette o crash

Tecnologie Web - 23



## Dichiarazione di tipo documento

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

```
<!DOCTYPE html>
```

- Questa *prima* riga solitamente viene generata in modo automatico dall'editor HTML specifico, non è obbligatoria, ed ha il compito di informare il browser che si tratta di un documento html relativo alle specifiche (in questo caso **XHTML 1.0 Strict**) e qual è la lingua primaria
- È necessaria quando si vuole validare la pagina web tramite un validatore, ad esempio: <http://validator.w3.org/>
- Se non si specifica un doctype valido, i browser passano in modalità "*quirks mode*"

Tecnologie Web - 24



## Namespace e lingua

- Namespace: collezione di tipi di elementi e nomi di attributi

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

```
<html lang="it">
```

- Per ragioni di accessibilità si deve definire la lingua principale del documento
- Si può fare lato server o con (xml:)lang (che può essere usato anche in linea)



## Content Type

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

character set: un altro  
valore molto usato è  
UTF-8

MIME Type

- Anche il content type può essere definito lato server tramite l'uso di uno script
- HTML5: `<meta charset="UTF-8" />`



## Intestazione

- La parte contenuta tra i tag `<head>` e `</head>` viene chiamata intestazione o semplicemente, sezione `head`.
- In questa sezione si trovano tutti i tag che impartiscono direttive al browser quali: titolo (obbligatorio), comandi meta, richiami ai fogli di stile, script.
- Tutto ciò che si trova all'interno della struttura head non sarà visualizzato ma interpretato dal browser. La sezione `head` quindi è una zona destinata ad uso esclusivo dei soli comandi che impartiscono direttive ben specifiche.



## All'interno della sezione Head

- `title` (obbligatorio), `link`, `meta`, `base`, `style` e `script`
- `link` definisce un collegamento ad una risorsa esterna (CSS, shortcut icon, etc)
  - attributi più comuni: `href`, `rel` (relazione)

```
<link rel="stylesheet" type="text/css" href="/file.css" />
<link rel="shortcut icon" href="/img.ico" />
<link rel="next" title="Next page" href="/next.html" />
```
- `base` definisce la posizione di base per i collegamenti

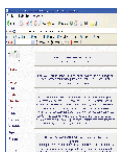
```
<base href="/miacartella/" />
```





## I tag META

- ❑ La sezione head contiene una serie di comandi, chiamati **MetaComandi**, che non producono alcuna variazione visiva sulla pagina, ma sono indispensabili per altre attività quali la validazione e la lettura da parte dei motori di ricerca.
- ❑ I metacomandi inseriscono informazioni aggiuntive sul contenuto del documento che si sta creando, come ad esempio l'autore.
- ❑ Non esistono limitazioni sul numero di metatag inseriti
- ❑ Ci sono due tipi di tag meta:
  - **http-equiv**
  - **name**



## Tag meta http-equiv

- ❑ L'informazione viene processata come se fosse presente in un header di risposta proveniente da un server HTTP, quindi prima del documento
- ❑ Possono condizionare la manipolazione del documento
  - <http://vancouver-webpages.com/META>

```
<meta http-equiv="refresh" content="15" />  
<meta http-equiv="refresh" content="1;URL=nuovaURL" />  
<meta http-equiv="expires" content="5" />
```



## Tag meta name

- ❑ Inseriscono informazioni riguardanti il documento
- ❑ È possibile creare valori propri (ex. suggeriti dai motori di ricerca)
- ❑ **description**: breve descrizione dei contenuti della pagina web. È particolarmente utile quando la pagina contiene poco testo (ex. statistiche).
- ❑ **keywords**: lista di parole chiavi separate da una virgola
- ❑ **copyright**
- ❑ **author**
- ❑ **robots**: utilizzato per prevenire l'indicizzazione di una pagina. Valori: index, noindex, follow, nofollow (i link della pagina), all, none
- ❑ **rating**: classificazione del contenuto



## Corpo del documento

- ❑ La parte contenuta tra i tag **<body>** e **</body>** viene chiamata corpo del documento o semplicemente, sezione **body**.
- ❑ Questa sezione contiene la pagina vera e propria, o almeno quello che si vedrà a video. Qui vengono inserite le immagini, i suoni, i filmati, e il testo, link e quant'altro.
- ❑ La sezione **body** contiene quindi tutti i tag che descrivono la **struttura** del documento: **non** devono essere usati elementi relativi alla **presentazione** visuale
  - si devono usare gli elementi per il loro significato e non per come vengono visualizzati dai browser (**<em>** vs **<i>**)





## Attributi del tag <body> (XHTML Transitional)

- ❑ **ATTENZIONE:** *questi attributi vengono riportati solo a scopo di documentazione e non devono essere usati nel progetto di questo corso*
- ❑ **background:** inserisce un'immagine come sfondo della pagina. Possono essere utilizzate immagini JPEG, GIF o PNG (non sempre supportato)
- ❑ **bgproperties:** l'immagine adoperata a riempimento può essere tenuta ferma durante lo spostamento verticale (scrolling) sulla pagina (no supportato da tutti i browser)
- ❑ **bgcolor:** utilizza un colore come sfondo della pagina
- ❑ **link:** indica il colore di tutti i link della pagina
- ❑ **vlink:** indica il colore di tutti i link, dopo che questi sono stati visitati
- ❑ **alink:** indica il colore di tutti i link attivi, il colore al momento del click su di esso



## Gli attributi comuni

- ❑ Possono essere utilizzati nella maggior parte dei tag. Sono divisi in 3 classi: core, i18n, e gli attributi evento
- ❑ Gli attributi **core** sono: **class** (specifica la classe di appartenenza), **id** (funziona come un'etichetta per fare riferimento ad un tag in modo univoco), **title** (aggiunge un titolo ad un elemento) e **style** (istruzioni CSS in linea)
- ❑ L'attributo **id** può essere usato come ancora per un link, per relazionare un elemento con la sua presentazione in CSS, oppure con JavaScript
- ❑ Gli attributi **internationalization** (i18n) sono **dir** (direzione, ltr o rtl) e **xml:lang**
- ❑ Gli attributi evento rappresentano gli eventi JavaScript: **onclick**, **ondblclick**, **onmousedown**, **onmouseup**, etc.



## I tag generalisti

- ❑ **<div> ... </div>**  
L'elemento **<div>** è un contenitore generico per l'associazione con fogli di stile e crea un nuovo blocco. Tutti gli attributi e le associazioni applicate al tag div saranno estese a tutto il blocco di codice interessato.

`<div class="center">`  
Questa riga di testo ed anche eventuali altri elementi,  
se presenti,  
subiranno in questo caso l'allineamento centrato.  
`</div>`

- ❑ **<span> ... </span>**  
L'elemento **<span>** non ha alcuna caratteristica se non quella di fare da supporto per gli stili. Diversamente da div, è un elemento in linea.

In un testo nero `<span class="green">` una parte pu<sup>o</sup>grave;  
essere colorata di verde `</span>`.



## id vs class

- ❑ In assenza di regole di stile ad essi associati, **div** e **span** non alterano in alcun modo la visualizzazione del contenuto
- ❑ **class** definisce un gruppo di appartenenza mentre **id** identifica un elemento in modo univoco
- ❑ Dare un **id** ad un elemento permette di usarlo:
  - come selettore in un foglio di stile
  - all'interno di uno script
  - come ancora di destinazione di un link
  - come strumento generico nel trattamento dei dati
- ❑ Un id deve cominciare con una lettera o con il carattere "\_". Per utilizzarlo all'interno di Javascript non sono ammessi spazi, apostrofi e punteggiatura.



## Il testo

- Il testo viene inserito tra i tag `<p>` e `</p>`.
- `<p>` La lettera p sta per paragrafo. In questo modo si formano dei paragrafi simili a quelli di questa slide. `</p>`
- `<p>` Tra un paragrafo e l'altro il browser inserisce un po' di spazio. All'interno dello stesso paragrafo è possibile andare a capo con il tag. `<br />`  
In questo caso lo spazio di interlinea non viene inserito. `</p>`
- `<hr />` inserisce una linea orizzontale
- All'interno del codice html si possono inserire dei commenti che non vengono visualizzati dal browser. È sufficiente inserirli tra i tag `<!--` e `-->`.



## Caratteri speciali

- Dato che le parentesi `<angolate>` servono a distinguere i tag xhtml dalle parole del testo, come faccio ad inserire una di queste nel testo della mia pagina?
- Lo stesso problema si pone con tutta una serie di caratteri speciali come lo spazio (in genere ignorato) o le vocali accentate, che vengono indicate con dei codici.

spazio	<code>&amp;nbsp;</code>	ò	<del><code>&amp;ograve;</code></del>
à	<del><code>&amp;agrave;</code></del>	ù	<del><code>&amp;ugrave;</code></del>
è	<del><code>&amp;egrave;</code></del>	ì	<del><code>&amp;igrave;</code></del>
"	<code>&amp;quot;</code>	&	<code>&amp;amp;</code>
<	<code>&amp;lt;</code>	>	<code>&amp;gt;</code>
€	<code>&amp;euro;</code>		

<http://www.usabile.it/unicode.htm>



## Cosmesi del testo - 1

- Ci sono due tipi di markup: *strutturale* e *di presentazione*
- All'interno del markup strutturale, possiamo distinguere una insieme di tag che condizionano in qualche modo il contenuto all'interno di essi
- Stili logici**: descrivono il significato, il contesto o l'uso dell'elemento che racchiudono
- Sostituiscono tag della prima formulazione di HTML troppo legati ad aspetti presentazionali
- L'aspetto di presentazione dipende dal browser utilizzato (e ovviamente può essere modificato tramite CSS), tuttavia ci sono delle convenzioni comuni



## Cosmesi del testo - 2

- Per rendere una pagina più leggibile si fa spesso ricorso ad una specie di cosmesi del testo per dare enfasi ad una parte del paragrafo
- `<em> </em>` = enfasi
- `<strong> </strong>` = forte enfasi
- Il modo in cui vengono visualizzati può essere manipolato tramite un foglio di stile
- Sono pensati per sostituire `<i>` e `<b>`, in quanto migliorano l'accessibilità (sono leggibili da uno screen reader) e contribuiscono a separare contenuto e presentazione



## Intestazioni

- Esistono sei livelli di intestazione: [h1](#), [h2](#), [h3](#), [h4](#), [h5](#), [h6](#).
- Si devono utilizzare rispettando l'ordine e pensando alla struttura del documento e non a come vengono visualizzati di default. La visualizzazione infatti può essere modificata

<body>

<h1>Heading 1 (h1)</h1>

<h2>Heading 2 (h2)</h2>

<h3>Heading 3 (h3)</h3>

<h4>Heading 4 (h4)</h4>

<h5>Heading 5 (h5)</h5>

<h6>Heading 6 (h6)</h6>

</body>

Heading 1 (h1)

Heading 2 (h2)

Heading 3 (h3)

Heading 4 (h4)

Heading 5 (h5)

Heading 6 (h6)



## Regole per scrivere dei buoni titoli

- Scrivi un titolo unico per ogni pagina
- Cerca di essere conciso e descrittivo
- Evita titoli vaghi e generici
- Utilizza la maiuscola per la prima lettera della frase o la prima lettera di ogni parola
- Crea contenuti degni di click ed evita i [clickbait](#)
- Pensa all'intento di ricerca
- Includi la parola chiave principale quando ha senso farlo
- Massimo 60 caratteri



## Il testo

- La marcatura del testo ha generato, nel tempo, molte cattive abitudini:
  - uso del tag `<br />`,
  - modifiche allo stile dei paragrafi per simulare le intestazioni,
  - ...
- La marcatura del testo deve corrispondere al significato semantico dell'elemento in essa racchiuso

<body>

<p class="titolo">....</p>

<p>...</p>

<p class="sottotitolo">....</p>

<p>.. </p>

</body>



## Citazioni

- Per riportare un passo e citare l'autore di devono usare i tag [blockquote](#), [q](#) o [cite](#)
- [blockquote](#) introduce un'ampia citazione che occupa un'intero blocco.
- [q](#) introduce una citazione più ristretta in linea
- la fonte può essere indicata tramite gli attributi [cite](#) (obbligatoriamente un URI) o [title](#), oppure con il tag `<cite>`
- ATTENZIONE:** in HTML5 [cite](#) assume un significato diverso, indica il titolo di un lavoro (libro, film, ...)



## Abbreviazioni, acronimi, indirizzi

- ❑ `abbr` indica le abbreviazioni
- ❑ `acronym` indica gli acronimi → obsoleto in HTML5
- ❑ `address`: identifica un indirizzo
- ❑ Servono davvero questi tag?
  - Nell'obiettivo di spostarci sempre più verso un web semantico, è bene cercare di strutturare quanto più possibile il testo, aggiungendo informazioni su di esso



## Altri tag per l'inserimento di testo particolare

- ❑ `code`: permette di inserire del codice all'interno di HTML.
- ❑ `var`: identifica delle variabili in un codice
- ❑ `samp`: identifica un particolare output di un programma
- ❑ `pre`: permette di inserire testo preformattato, dove spazi, tabulazioni e a capo hanno un valore
- ❑ `ins`: identifica un inserimento redazionale. Solitamente è visualizzato sottolineato
- ❑ `del`: identifica una cancellazione redazionale. Solitamente è visualizzata barrata
- ❑ Possono essere usati sia come elementi in linea che di blocco e può esservi associato l'attributo `cite` per identificare l'autore



## Visualizzazione degli stili logici

Tag	Descrizione	Solitamente visualizzato come
<code>&lt;abbr&gt;&lt;/abbr&gt;</code>	Abbreviazione	Testo normale
<code>&lt;acronym&gt;&lt;/acronym&gt;</code>	Acronimo	Testo normale
<code>&lt;cite&gt;&lt;/cite&gt;</code>	Riferimento a un altro documento	Corsivo
<code>&lt;code&gt;&lt;/code&gt;</code>	Codice	Font a larghezza fissa
<code>&lt;del&gt;&lt;/del&gt;</code>	Testo cancellato	Testo normale / Testo barrato orizzontalmente
<code>&lt;dfn&gt;&lt;/dfn&gt;</code>	Definizione di un'istanza	Testo normale
<code>&lt;em&gt;&lt;/em&gt;</code>	Enfaticizzato	Corsivo
<code>&lt;ins&gt;&lt;/ins&gt;</code>	Testo inserito	Testo normale
<code>&lt;kbd&gt;&lt;/kbd&gt;</code>	Testo da tastiera	Font a larghezza fissa
<code>&lt;q&gt;&lt;/q&gt;</code>	Citazione breve	Corsivo
<code>&lt;samp&gt;&lt;/samp&gt;</code>	Testo di esempio	Font a larghezza fissa
<code>&lt;span&gt;&lt;/span&gt;</code>	Parte	Testo normale
<code>&lt;strong&gt;&lt;/strong&gt;</code>	Evidenziato	Grassetto
<code>&lt;var&gt;&lt;/var&gt;</code>	Variabile	Font a larghezza fissa / Corsivo



## Stili fisici

- ❑ Sono markup di presentazione che forniscono istruzioni precise sulla visualizzazione di un elemento
- ❑ Sono fortemente sconsigliati

Tag	Descrizione	Funzione
<code>&lt;b&gt;&lt;/b&gt;</code>	Grassetto	Testo in grassetto
<code>&lt;big&gt;&lt;/big&gt;</code>	Grande	Testo leggermente più grande del testo adiacente
<code>&lt;blink&gt;&lt;/blink&gt;</code>	Lampeggiante	Testo lampeggiante (Netscape)
<code>&lt;font&gt;&lt;/font&gt;</code>	Carattere	Specifica il tipo di carattere, la dimensione e il colore (sconsigliato)
<code>&lt;i&gt;&lt;/i&gt;</code>	Corsivo	Testo in corsivo
<code>&lt;s&gt;&lt;/s&gt;</code>	Testo barrato orizzontalmente	Testo barrato orizzontalmente (sconsigliato)
<code>&lt;small&gt;&lt;/small&gt;</code>	Piccolo	Testo leggermente più piccolo del testo adiacente
<code>&lt;strike&gt;&lt;/strike&gt;</code>	Testo barrato orizzontalmente	Testo barrato orizzontalmente (sconsigliato)
<code>&lt;sub&gt;&lt;/sub&gt;</code>	Pedice	Testo di dimensione più piccola, sopra la linea di base del testo adiacente
<code>&lt;sup&gt;&lt;/sup&gt;</code>	Apice	Testo di dimensione più piccola, sotto la linea di base del testo adiacente
<code>&lt;tt&gt;&lt;/tt&gt;</code>	Telescrivente	Carattere a larghezza fissa
<code>&lt;u&gt;&lt;/u&gt;</code>	Sottolineato	Testo sottolineato (sconsigliato)



## I font (obsoleti)

- HTML permetteva di cambiare il colore, le dimensioni e il tipo di carattere adoperato per la scrittura. XHTML Strict non permette l'uso di questo tag.
- <font>**: il tag font cambia il font utilizzato dal testo racchiuso nei tags `<font>` `</font>`

### Attributi:

- color**, **face**, **size**, rispettivamente per colore, tipo di carattere e dimensioni dello stesso:  
`<font color="colore">`  
`<font face="nome del carattere">`  
`<font size="3">`
- face può non essere rispettato se non si dispone di quel particolare carattere

### Esempio:

- `<font color="#008000" size="3" face="Comic Sans MS">`  
**Sto scrivendo con un font di dimensione 3, colore verde e carattere Comic Sans MS**`</font>`



## I font (obsoleti): attributo size

- L'attributo size definisce la dimensione del testo su una scala da 1 a 7, inventata dai browser, quindi senza riferimenti con le misure tipografiche convenzionali
- Il valore di default è 3
- La definizione può avvenire anche in modo relativo, ma in ogni caso mai più piccola di 1 o maggiore di 7



## Elenchi non ordinati

- Elenchi puntati da utilizzare quando vogliamo dei punti per il nostro elenco, senza un ordine ben preciso.
- <ul>**: ogni elemento di lista è compreso all'interno di un elemento **<li>** (List Item).

CODICE	RISULTATO
Oggi devo comprare: <code>&lt;ul&gt;</code> <code>&lt;li&gt;Mele&lt;/li&gt;</code> <code>&lt;li&gt;Pere&lt;/li&gt;</code> <code>&lt;li&gt;Angurie&lt;/li&gt;</code> <code>&lt;li&gt;Limoni&lt;/li&gt;</code> <code>&lt;/ul&gt;</code>	Oggi devo comprare: <ul style="list-style-type: none"><li>Mele</li><li>Pere</li><li>Angurie</li><li>Limoni</li></ul>



## Elenchi ordinati

- Elenchi numerati da utilizzare quando vogliamo dei punti che abbiano una gerarchia o un ordine ben preciso.
- <ol>**: ogni elemento di lista è compreso all'interno di un elemento **<li>** (List Item).

CODICE	RISULTATO
Per piantare un chiodo devo: <code>&lt;ol&gt;</code> <code>&lt;li&gt;Prendere martello e chiodo&lt;/li&gt;</code> <code>&lt;li&gt;Sollevare il martello&lt;/li&gt;</code> <code>&lt;li&gt;Colpire ripetutamente il chiodo col martello finché questo non è piantato&lt;/li&gt;</code> <code>&lt;/ol&gt;</code>	Per piantare un chiodo devo: <ol style="list-style-type: none"><li>Prendere martello e chiodo</li><li>Sollevare il martello</li><li>Colpire ripetutamente il chiodo col martello finché questo non è piantato</li></ol>



## Elenchi di definizioni

- Elenchi in cui non si utilizza alcun tipo di punto, utili soprattutto per definire dei termini.
- **<dl>**: il termine da definire è indicato dall'elemento **<dt>** e la definizione dall'elemento **<dd>**.
- Ci possono essere più **<dd>** per un unico **<dt>** e viceversa.

### CODICE

```
<dl>
  <dt>Uomo</dt>
  <dd>Essere vivente, amante
  e desiderante. Bipede
  implume dotato di una
  intelligenza più o meno grande
  che può usare o meno</dd>
</dl>
```

### RISULTATO

**Uomo**  
Essere vivente, amante e desiderante. Bipede implume dotato di una intelligenza più o meno grande che può usare o meno



## Elenchi e navigazione

- Una barra di navigazione è essenzialmente un elenco di link

```
<ul>
  <li>Home</li>
  <li>Azienda</li>
  <li>Prodotti</li>
</ul>
```

```
<dl>
  <dt>Home</dt>
  <dt>Azienda</dt>
  <dd>Sede 1</dd>
  <dd>Sede 2</dd>
  <dd>Sede 3</dd>
  <dt>Prodotti</dt>
  <dd>Prodotto 1</dd>
  <dd>Prodotto 2</dd>
</dl>
```

- Deve essere definito un id per modificare il layout tramite CSS



## Immagini

- Per inserire un'immagine si utilizza il tag ****, dove **xxx** è il nome dell'immagine e **yyy** la sua estensione. Le immagini consentite dal linguaggio html sono .gif .jpg e .png.

```
<body>
  <p>Questa è la mia prima pagina web </p>
  
</body>
```

### Attributi:

- **alt** = testo alternativo
- **longdesc** = URI ad una pagina con una descrizione dell'immagine
- **width** (larghezza) ed **height** (altezza) fanno conoscere la precisa dimensione dell'immagine al browser prima di scaricarla
- **align**, **hspace**, **vspace**



## Link

- Per inserire un link si usa il tag **<a>**, ancora.
- Sorgente del link può essere un pezzo di testo (*hot word*) ma anche elementi più complessi come le immagini (*thumbnail*). Destinazione del link può essere una pagina o una sua parte.

protocollo      porta      parte del testo

```
<a href="http://ind_server:8080/path/doc.html#frammento">
  Sorgente del link
</a>
```

- I riferimenti possono essere assoluti o relativi (eventualmente utilizzando il tag **base**).
- **target** indica il frame di destinazione (se non esiste apre una nuova finestra). Nella versione **Strict** non è valido, **HTML5** sì
- **HTML5**: **media** (media query), **download**



## Accesso ad un frammento

- ❑ Si possono indirizzare dei frammenti di un documento in due modi:
  1. tramite la definizione di un **name** (OBSOLETO)  
`<h1 name="title" />`  
...  
`<a href="#title" > Vai alla sezione title </a>`
  1. tramite la definizione di un **id** (non supportato nei vecchi browser)  
`<h1 id="title" />`  
...  
`<a href="#title" > Vai alla sezione title </a>`



## Accesso ai link da tastiera

- ❑ È molto importante che i link siano accessibili anche agli utenti non in grado di utilizzare il mouse
- ❑ **accesskey** e **tabindex** indicano rispettivamente un carattere per portare il focus sul link e l'ordine di tabulazione

```
<a href="http://ind_server:8080/path/doc.html#frammento"
    tabindex="1" accesskey="s">
    Sorgente del link
</a>
```



## Link non ipertestuali

- ❑ Richiedono una configurazione del browser che deve aprire il programma corretto
- ❑ Mail
  - `mailto:username@domain`
  - `<a href="mailto:utente@dominio.it">scrivimio</a>`
  - Esistono delle funzioni aggiuntive supportate solo da alcuni che permettono di preimpostare alcuni parametri
- ❑ FTP link
  - `<a href="ftp://server/pathname">...</a>`
- ❑ Altri
  - `<a href="file://server/pathname">...</a>`
  - `<a href="news:newsgroup">...</a>`



## Tabelle

- ❑ Le tabelle servono per tabulare colonne di dati.
- ❑ Purtroppo sono usate spesso come contenitori per testi ed immagini per migliorarne la disposizione nella pagina, portando a codice di bassa qualità.
- ❑ Una tabella si crea con il tag `<table>`. `<tr>` e `<td>` indicano, rispettivamente, le righe e le colonne. Interi tabelle possono poi essere a loro volta contenute in celle di altre tabelle, che vengono quindi nidificate come scatole cinesi.

```
<table>
  <tr>
    <td> qui andrà messo il contenuto della tabella </td>
  </tr>
</table>
```





## Tabelle per il layout

- ❑ Nel passato, lo scarso supporto del CSS da parte dei browser ha promosso l'uso di tabelle per l'impaginazione
- ❑ Questa pratica ha portato a diversi problemi
  - accessibilità con dispositivi non visuali
  - lentezza nel caricamento dei dati (la visualizzazione di una tabella richiede molti calcoli al browser)
  - struttura e contenuto non separati, ma entrambi presenti nei dati



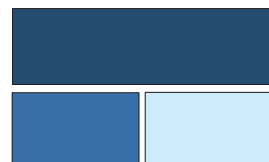
## Regole per le tabelle

- ❑ Non ci possono essere righe senza celle al suo interno.
- ❑ Le colonne non si definiscono in modo esplicito ma si definiscono le celle all'interno delle righe tramite gli elementi `td`.
- ❑ Si possono definire celle che occupano più di una colonna (`colspan`) o più di una riga (`rowspan`).
- ❑ È possibile creare delle intestazioni per le colonne (o per le righe) con gli elementi `th` al posto di `td`.
- ❑ Il tag `caption`, posto subito dopo il tag `table`, permette di inserire un titolo (in genere visualizzato sopra la tabella).
- ❑ L'attributo `summary` permette di descrivere il contenuto della tabella



## colspan e rowspan

```
<table>
  <tr>
    <td colspan="2">cella 1</td>
  </tr>
  <tr><td>cella 2</td>
    <td>cella3</td>
  </tr>
</table>
```



```
<table>
  <tr>
    <td rowspan="2">cella 1</td>
    <td>cella 2</td>
  </tr>
  <tr>
    <td>cella3</td>
  </tr>
</table>
```



## Raggruppare le righe

- ❑ È possibile raggruppare alcune righe suddividendo la tabella in header, body e footer.
- ❑ Quando la tabella viene interrotta in qualche modo (ex. stampa) header e footer vengono ripetuti (non in IE).

```
<table>
  <thead>
    <tr>....</tr> <tr>....</tr> <tr>....</tr>
  </thead>
  <tfoot>
    <tr>....</tr> <tr>....</tr> <tr>....</tr>
  </tfoot>
  <tbody>
    <tr>....</tr> <tr>....</tr> <tr>....</tr>
  </tbody>
</table>
```



## Esempio

```
<table>
  <thead>
    <tr><th>1 col</th><th>2 col</th><th>3 col</th></tr>
  </thead>
  <tbody>
    <tr><td>Dato 1</td><td>Dato 2</td><td>Dato 3</td></tr>
    <tr><td>Dato 4</td><td>Dato 5</td><td>Dato 6</td></tr>
    <tr class="alternative"><td>Dato 1</td><td>Dato 2</td><td>Dato 3</td></tr>
    <tr><td>Dato 4</td><td>Dato 5</td><td>Dato 6</td></tr>
    <tr><td>Dato 1</td><td>Dato 2</td><td>Dato 3</td></tr>
    <tr><td>Dato 4</td><td>Dato 5</td><td>Dato 6</td></tr>
  </tbody>
</table>
```



## Esempio

1 col	2 col	3 col
Dato 1	Dato 2	Dato 3
Dato 4	Dato 5	Dato 6
Dato 1	Dato 2	Dato 3
Dato 4	Dato 5	Dato 6
Dato 1	Dato 2	Dato 3
Dato 4	Dato 5	Dato 6

Diagram illustrating the structure of an HTML table with 3 columns and 7 rows. The first row is the header (thead) with 3 columns. The remaining 6 rows are the body (tbody). The first column is labeled '1 col', the second '2 col', and the third '3 col'. The first row is labeled 'th' (header), and the subsequent rows are labeled 'tr' (table row). The first column is labeled 'td' (table data). The first row is labeled 'tr class="alternative"'.



## Indirizzare le colonne

- Sebbene le tabelle si costruiscano per righe, è possibile indirizzare le colonne, per creare effetti di layout ad esse associati
- colgroup** consente di applicare attributi ad un set di colonne identificato dall'attributo **span** (simile a **rowspan** e **colspan**)
- col** permette di selezionare una singola colonna

```
<table>
  <colgroup span="2"
    class="alternative" />
  <tr> ... </tr>
</table>
```



## Esempio

```
<table>
  <colgroup>
    <col />
    <col class="alternative" />
    <col />
    <col class="alternative" />
  </colgroup>
```

This	That	The other	Lunch
Ladybird	Locust	Lunch	Lunch
Ladybird	Locust	Lunch	Lunch
Ladybird	Locust	Lunch	Lunch
Ladybird	Locust	Lunch	Lunch

Diagram illustrating the use of the **colgroup** element to group columns. The table has 4 columns. The first column is labeled 'This', the second 'That', the third 'The other', and the fourth 'Lunch'. The first row is labeled 'Caption'. The first column is labeled 'td'. The first row is labeled 'tr class="alternative"'.



## Attributi per le tabelle - 1 (sconsigliati)

- ❑ **height** e **width**: altezza e larghezza. Possono essere espressi anche in termini percentuali.
- ❑ **align** (**valign**): specifica l'allineamento: center, left e right.
- ❑ **background** e **bgcolor**: permettono rispettivamente di inserire un'immagine o un colore come sfondo della tabella o del singolo elemento della tabella.
- ❑ **border**: permette di impostare lo spessore del bordo perimetrale che contorna tutte le celle facenti parte della tabella.



## Attributi per le tabelle - 2 (sconsigliati)

- ❑ **bordercolor**, **bordercolordark**, **bordercolorlight**: permettono di selezionare il colore del bordo (solo IE).
- ❑ **cellpadding**: specifica la quantità di spazio vuoto lasciato tra i bordi delle celle di una tabella e il dato vero e proprio in esse contenuto. Il valore per default è 1.
- ❑ **cellspacing**: specifica la quantità di spazio vuoto da lasciare tra le singole celle di dati di una tabella. Il valore per default è 2.



## Posizionamento di una tabella

- ❑ Esistono tre metodi, uno solo accettato dal W3C:
  1. uso dell'attributo **align** nel tag **table**
  2. inserire la tabella in un tag **center**
  3. usare lo standard CSS eventualmente inserendo la tabella in un tag **div**



## I form

"Se l'economia fa girare il mondo, allora i form fanno girare il Web."

XHTML & CSS, P. Griffit



## Definizione di un form

```
<form action="http://server/path/file.cgi" method="post" >  
  <!-- elementi del form -->  
</form>
```

- L'attributo `method` può avere due valori, `get` e `post`
- Metodo **get**: è il predefinito. Viene utilizzato per leggere dati. Il browser allega la **stringa di query** all'url del programma CGI
  - `http://server/path/file.cgi?parametro=valore`
  - limite alla lunghezza della stringa
  - vulnerabilità dell'accesso
- Metodo **post**: viene utilizzato per inviare dati. La stringa di query viene passata come input standard
  - maggiore facilità di gestione



## Formato della stringa di query

- Contiene i dati inviati cliccando il pulsante **Submit**
- Il nome e il valore di ciascun elemento della form sono codificati come assegnamenti
  - Ex. **nome=Mario&Cognome=Rossi**
- I caratteri speciali sono codificati sottoforma di numeri esadecimali preceduti da %
  - Ex. Lo spazio è rappresentato da %20: `Nome=Mario%20Rossi`
- Con il metodo **get** la pagina di destinazione può essere salvata come bookmark in modo da poter ripetere la query senza reinserire i dati
- Se si usa il metodo **get** la stringa viene inserita dal server in una variabile d'ambiente
- Se si usa il metodo **post** si deve leggere la stringa di query dall'input standard



## Campi e pulsanti dei form

- Gli elementi inseriti in una form si inseriscono con soli 3 tag:
  - `input`
  - `textarea`
  - `select`



## Attributi per gli elementi dei form

- **name**: serve per identificare l'input inviato al server. Ogni elemento viene inviato come una coppia nome/valore. Il nome si ricava da questo attributo, il valore è l'input inserito dall'utente in quel campo.
- **readonly="readonly"**: i campi con questo attributo non sono editabili dall'utente.
- **disabled="disabled"**: i campi con questo attributo non sono editabili dall'utente. **Il valore di questo campo non viene inviato al server.**

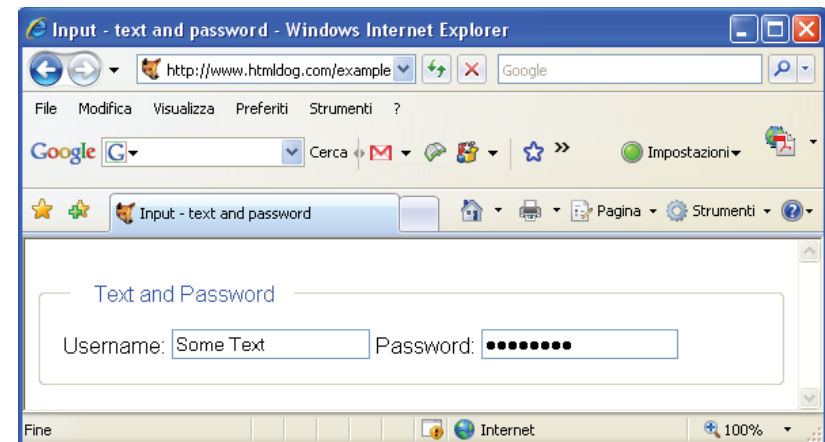


## Il tag input

- ❑ Questo tag permette da solo di creare diversi elementi di una form a seconda del contenuto dell'attributo **type**:
  - **text**: una singola riga di testo con **maxlength** elementi
  - **password**: una riga di testo offuscata
  - **checkbox**: un semplice on/off
  - **radio**: per selezionare una o più opzioni
  - **submit**: pulsante per inviare i dati del modulo
  - **reset**: pulsante per riportare i valori predefiniti nei campi del modulo
  - **hidden**: per dati non visibili o non editabili
  - **file**: per caricare file
  - **button**: per richiamare script lato client
  - **image**



## Esempio: testo e password



<http://www.htmldog.com/examples/inputtextboxes.html>



## Esempio: testo e password - codice

```
<form action="">  
  <fieldset>  
    <legend>Text and Password</legend>  
    <label for="username">Username:</label>  
    <input name="username" id="username"  
          value="Some Text"  
          maxlength="20"/>  
    <label for="password">Password:</label>  
    <input type="password" name="password" id="password"  
          value="Password"  
          maxlength="20" />  
  </fieldset>  
</form>
```



## fieldset e label

- ❑ In caso di form molto lunghi, il tag **fieldset** permette di raggruppare elementi logicamente correlati: questa operazione in genere agevola la loro compilazione.
- ❑ Il tag **legend** permette di inserire una intestazione. È situato subito dopo il tag di apertura **<fieldset>**.
- ❑ La visualizzazione predefinita riquadra l'insieme di elementi con un bordo con la legenda che interrompe il bordo superiore.
- ❑ Il tag **label** associa un'etichetta ad un campo del form (non necessariamente adiacente) con **id** il valore dell'attributo **for**.



## Esempio: checkbox e radio

Films you like

Drama	<input type="checkbox"/>
Action	<input checked="" type="checkbox"/>
Comedy	<input type="checkbox"/>
Horror	<input checked="" type="checkbox"/>
Sci-fi	<input type="checkbox"/>

Your age

19 or under	<input type="radio"/>
20 to 39	<input type="radio"/>
40 to 59	<input type="radio"/>
60 or over	<input checked="" type="radio"/>

<http://www.htmldog.com/examples/inputcheckboxes.html>



## Esempio: checkbox e radio - code

```
<form action="">
  <fieldset> <legend>Films you like</legend>
    <div><label for="drama">Drama</label><input type="checkbox"
      name="drama" id="drama" value="drama" /></div>
    <div><label for="action">Action</label><input type="checkbox"
      name="action" id="action" value="action" /></div>
    ...
  </fieldset>
  <fieldset> <legend>Your age</legend>
    <div><label for="lt20">19 or under</label>
      <input type="radio" name="age" id="lt20" value="lt20" />
    </div>
    ... <input type="radio" name="age" id="f20to39" value="20to39">
    ...
  </fieldset>
</form>
```

## hidden

- I tag input di tipi hidden non vengono visualizzati nel form e non possono in alcun modo interagire con l'utente
- Possono essere usati per:
  - passaggio dati in modo da non richiederli all'utente in una sequenza di form (ex. *wizard*)
  - salvataggio di informazioni calcolate sulla base dei dati inseriti dall'utente (ex. id)
  - definizione di variabili



## checkbox vs radio

- I pulsanti di scelta **radio** permettono la selezione di un'unica voce, mentre i pulsanti **checkbox** permettono una scelta multipla
- `checked="checked"` permette di specificare lo stato iniziale del pulsante di scelta
- Se un pulsante **checkbox** non è selezionato non viene inviato al server, altrimenti viene inviato il valore `on` associato al nome del controllo, oppure il valore dell'attributo `value`, se presente
- Per i pulsanti di tipo **radio** è obbligatorio definire l'attributo `value`, che viene inviato al server in caso di selezione
  - Ex. age=lt20



## file

- Consente all'utente di selezionare un file dal proprio computer
- Se viene usato un tag input di tipo **file**, il tag form di apertura deve contenere l'attributo **enctype= "multipart/form-data"** che comunica al server che si stanno inviando dati non solo testuali
- Non può essere usato con **method= "get"**

Upload file

File name:



## Il tag textarea

- Permette all'utente di inserire testo più lungo di una riga  
`<textarea rows="20" cols="40" name="message">`  
scrivi qualcosa qui  
`</textarea>`
- Gli attributi **rows** e **cols** sono **obbligatori** e **textarea** ha un tag di apertura ed uno di chiusura

Contact us

Name:

Email address:

Message:



## Il tag select - 1

- Permette di creare elenco di dati, in genere visualizzato come menù a tendina, su cui effettuare una o più scelte.

```
<select name="book" id="book">
  <optgroup label="Camus">
    <option>The Outsider</option>
    <option>The Rebel</option>
    <option>The Plague</option>
  </optgroup>
  <optgroup label="Orwell">
    <option>Animal Farm</option>
    <option>Nineteen Eighty-Four</option>
    <option>Down and Out in Paris and London</option>
  </optgroup>
</select>
```

Favourite book

Name:

**Camus**

- The Outsider
- The Rebel
- The Plague

**Orwell**

- Animal Farm
- Nineteen Eighty-Four
- Down and Out in Paris and London



## Il tag select - 2

- Per impostazione, viene visualizzata solo la prima opzione. Con l'attributo **size**, si può modificare questa scelta.
- Viene inviato al server la coppia nome del tag **select**/contenuto del tag **option** scelto, o valore del suo attributo **value** se presente
  - Ex. book= "The Outsider"





## Tag "nocivi" (P. Griffiths)

- Sono tag che si occupano di aspetti di presentazione o tag non validi
- Presentazionali: b, i, big, small, marquee, blink, u, tt, sub, sup, center, hr, etc.
- Altri: applet e embed (si deve usare object), font, frame, frameset, iframe, etc.
- **ATTENZIONE:** HTML5 permette l'uso di iframe e di small



## XHTML 1.1

- **XHTML 1.0** è stato scritto per tradurre HTML in un linguaggio XML
  - È pensato come un linguaggio di transizione (3 DTD che facilitano questo passaggio)
- **XHTML 1.1** invece elimina definitivamente tutti gli elementi di presentazione
- Il linguaggio viene frammentato in diversi moduli indipendenti. Ogni modulo definisce una caratteristica del linguaggio. Ex:
  - struttura: head, body, title, ...
  - testo
  - form, ...
- Ogni modulo viene richiamato solo se necessario



## HTML5

## Uno sguardo a HTML5

- HTML5 è il primo linguaggio di markup realizzato da produttori di browser e non da esperti di informatica
- È pensato per accelerare lo sviluppo di applicazioni web che vanno a rimpiazzare prodotti desktop
  - Calendari
  - Gestori di email, documenti, foto
  - ...
- e tecnologie proprietarie
  - Microsoft Silverlight
  - Adobe Flex



## Innovazioni introdotte

- ❑ Regole sintattiche meno stringenti
- ❑ Gestione standard degli errori
- ❑ **canvas**: un'area di disegno interattiva
- ❑ **video**, **audio**: non rende necessaria la presenza di un plug-in
- ❑ Interazione con le API
- ❑ Gestione della posizione tramite **GeoLocation API** del W3C
- ❑ Javascript multithread
- ❑ Pagine modificabili dall'utente (**contenteditable**, **draggable**, **spellcheck**)
- ❑ Possibilità di usufruire delle pagine/applicazioni, anche in modalità off-line
- ❑ Possibilità di accedere in modo sicuro ad un database locale
- ❑ Non si parla più di elementi deprecati, ma obsoleti

Tecnologie Web - 93



## Tag Meta

- ❑ In HTML5 il tag meta non deve necessariamente essere chiuso
- ❑ Altre innovazioni utili:

```
<meta http-equiv="X-UA-compatible"
      content="IE=edge,chrome=1" />
```

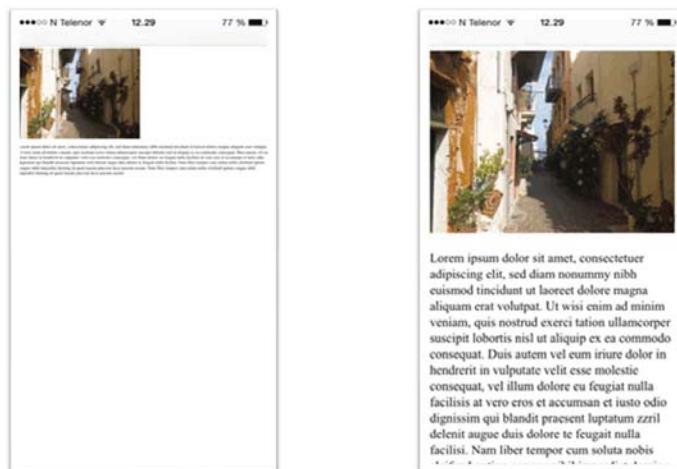
```
<meta name="viewport"
      content="width=device-width" />
```

- ❑ Il tag viewport può contenere nel content anche **initial-scale**, **minimum-scale**, **maximum-scale** e **user-scalable**

Tecnologie Web - 94



## Viewport



[http://www.w3schools.com/css/css\\_rwd\\_viewport.asp](http://www.w3schools.com/css/css_rwd_viewport.asp)

Tecnologie Web - 95



## Markup Strutturale - 1

- ❑ HTML 5 introduce markup in grado di descrivere meglio la struttura interna di un documento
- ❑ Nuovi tag introdotti:
  - **header**, **footer**: intestazione e piè pagina di una documento. Possono essere usati più volte nella stessa pagina, anche all'interno delle sezioni. **footer** identifica le informazioni su chi ha scritto i contenuti
  - **main**: contenuto principale
  - **nav**: contiene elementi di supporto alla navigazione. Può comparire anche dentro un header

Tecnologie Web - 96

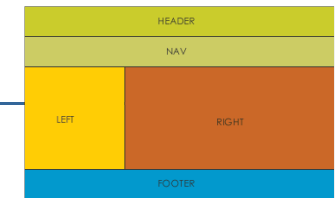


## Markup Strutturale - 2

- Nuovi tag introdotti:
  - **aside**: sidebar, contenuto a parte, a supporto, non necessariamente a destra o a sinistra. Identifica un parte di contenuto che può essere rimossa senza diminuire il significato della pagina (o della sezione), ma che è legata al contenuto del tag in cui è annidata
  - **section**: per raggruppare contenuti sullo stesso tema o logicamente collegati (ex. capitoli di un libro)
  - **article**: porzione di testo autocontenuto e indipendente dal resto del documento che possa essere distribuito in modo autonomo (ex. post di un blog, articolo di giornale)



## Struttura HTML



HTML 3	HTML 4	HTML 5
<pre>&lt;body&gt; &lt;table&gt;   &lt;tr&gt;     &lt;td&gt;header&lt;/td&gt;   &lt;/tr&gt;   &lt;tr&gt;     &lt;td&gt;nav&lt;/td&gt;   &lt;/tr&gt;   &lt;tr&gt;     &lt;td&gt;left &lt;/td&gt;     &lt;td&gt;right &lt;/td&gt;   &lt;/tr&gt;   &lt;tr&gt;     &lt;td&gt;footer&lt;/td&gt;   &lt;/tr&gt; &lt;/table&gt; &lt;/body&gt;</pre>	<pre>&lt;body&gt;   &lt;div id="header"&gt;...&lt;/div&gt;   &lt;div id="nav"&gt;...&lt;/div&gt;   &lt;div class="right"&gt;     ...   &lt;/div&gt;   &lt;div id="left"&gt;...&lt;/div&gt;   &lt;div id="footer"&gt;...&lt;/div&gt; &lt;/body&gt;</pre>	<pre>&lt;body&gt;   &lt;header&gt;...&lt;/header&gt;   &lt;nav&gt;...&lt;/nav&gt;   &lt;main&gt;     &lt;aside&gt; ... &lt;/aside&gt;     &lt;section&gt;       ...     &lt;/section&gt;   &lt;/main&gt;   &lt;footer&gt; &lt;/footer&gt; &lt;/body&gt;</pre>



## Regole del Markup strutturale

- È bene non ricorrere a **section** od **article** per soli motivi di stile o di scripting, in tal caso **div** è preferibile
- **article**, **nav**, **section** e **aside** sono *sectioning element*, ovvero possono contenere **header**, **nav** e **footer**
- Per controllare se il documento è stato strutturato bene una buona possibilità è verificare il sommario generato automaticamente
  - <http://gsnedders.html5.org/outliner/>
- Se il browser non supporta il markup strutturale (ex. IE9) esiste una libreria Javascript che la interpreta per lui
  - <http://html5shiv.googlecode.com/svn/trunk/html5.js>



## Esempio header - 1

```
<div id="header">
  
  <div id="headerText">
    <div id="headerUtility">
      <form> ..</form>
    </div>
    <h1>Corsi di laurea in informatica</h1>
    <h2>Sito dedicato ai corsi di laurea in
      informatica</h2>
  </div>
</div>
```



## Esempio header - 2

```
<header role="banner">
  
  <div id="headerText">
    <nav>
      <form> ..</form>
    </nav>
    <h1>Corsi di laurea in informatica</h1>
    <p>Sito dedicato ai corsi di laurea in informatica</p>
  </div>
</header>
```



## Altri tag semantici

- HTML5 introduce altri tag per caratterizzare altri dati contenuti in una pagina web
  - **figure**
  - **mark**
  - **time** (con l'attributo **datetime** che contiene la data - o l'ora - in formato XML)
  - **meter** per indicare una misura in una scala che ha un minimo (**min**) ed un massimo (**max**)
  - **progress** per indicare un valore che sta cambiando
  - **small** per indicare le note a piè pagina, i termini in piccolo dei contratti, etc



## Form

- HTML5 aggiunge alcuni widget che possono essere utilizzati nelle form
  - **Datalist**

```
<input list="browsers">

<datalist id="browsers">
  <option value="Internet Explorer">
  <option value="Firefox">
  <option value="Chrome">
  <option value="Opera">
  <option value="Safari">
</datalist>
```
  - **placeholder**



## Innovazioni per le form - 1

- Al tag **<form>** vengono aggiunti i seguenti attributi
  - **target**: indica dove visualizzare la risposta (**\_blank**, **\_self**, **\_parent**, **\_top**, **\_iframename**)
  - **autocomplete**
  - **novalidate**
- Al tag **<input>** vengono aggiunti i seguenti valori per l'attributo **type**
  - **number** (inserisce due frecce per aumentare o diminuire il valore, ma rimane editabile), **range**
  - **color**
  - **email**, **url**, **tel**
  - **search**
  - **datetime**, **datetime-local**, **date**, **month**, **time**, **week**



## Innovazioni per le form - 2

- Vengono inoltre aggiunti i tag:
  - **datalist** per definire liste di suggerimenti
  - **keygen** per generare le chiavi per un sistema crittografico
  - **menu** per i menù contestuali
  - **output** che funge da segnaposto per i risultati di un calcolo



## Nuovi attributi per i controlli

- **required**
- **formnovalidate**
- **pattern**: contiene un'espressione regolare per la validazione dell'input
- **placeholder**: contiene un suggerimento
- **autocomplete**, **autofocus**
- **spellcheck**
- **min**, **max**, **step**
- **multiple**



## Un esempio

Dettagli:

Nome:	<input type="text" value="Nome"/>
Cognome:	<input type="text" value="Cognome"/>
Password:	<input type="password" value="....."/>
Data di nascita:	<input type="text" value="31/02/2014"/>
Sesso:	<input type="radio"/> Masc <span>! Inserisci un valore valido. Il campo è incompleto o presenta una data non valida.</span>
Email:	<input type="text" value="Ad esem"/>
Indirizzo Web:	<input type="text" value="Ad esem"/>
Materie preferite:	<input type="checkbox"/> Storia: <input type="checkbox"/> Matematica: <input type="checkbox"/> Geografia: <input type="checkbox"/> Fisica:
Quanti libri leggi all'anno?	<input type="text" value="1"/>
Ogni quanti mesi compri un libro?	<input type="text" value="1"/>
Il tuo colore preferito?	<input type="text" value=""/>
Seleziona un sistema operativo:	<input type="text" value="FreeBSD"/>
Scegli un linguaggio di programmazione:	<input type="text" value=""/>
Note:	<div></div>



## Innovazioni per le liste ordinate

- HTML5 aggiunge al tag **<ol>** i seguenti attributi
  - **reversed**
  - **start**: indica il numero con cui parte la lista
  - **type**: specifica il tipo di marcatore
- Inoltre aggiunge al tag **<li>** l'attributo **value** che consente di impostare un numero arbitrario



## figure e figcaption

- ❑ `figure` e `figcaption` permettono di definire figure e didascalie
- ❑ Una figura non deve necessariamente contenere un'immagine

`<figure>`

``

`<figcaption>Didascalia della figura</figcaption>`

`</figure>`

- ❑ Un'immagine non ha bisogno dell'attributo `alt` se ha associata una didascalia



## Canvas

- ❑ È un elemento bit-map che permette di disegnare degli elementi, quindi di creare immagini animate
- ❑ Deve essere usato solo quando appropriato (ad esempio non per disegnare un header)
- ❑ È necessario impostare dei fallback

`<canvas id="canvasID" width="300" height="200">`

`</canvas>`

Tutto ciò che è qui viene visualizzato solo dai browser che non supportano HTML5



## Cosa si può fare con un canvas

- ❑ Disegnare forme, testo, linee e curve
- ❑ Colorare forme, testo, linee e curve
- ❑ Creare gradienti e pattern
- ❑ Copiare immagini, immagini di un video e altri canvas
- ❑ Manipolare i pixel
- ❑ Esportare il contenuto di un canvas in un file statico



## Canvas & JavaScript

- ❑ Tutto ciò che viene fatto nelle canvas viene realizzato tramite JavaScript
  - Canvas 2D API
  - <http://html5doctor.com/an-introduction-to-the-canvas-2d-api/>

```
var canvas = document.getElementById('canvasID');  
var context = canvas.getContext('2d');  
context.strokeStyle = '#990000';  
context.strokeRect(20,30,100,50);
```



```
strokeRect(left, top, width, height);  
fillStyle, fillRect, lineWidth, shadowColor, ...
```



## Pro e contro delle canvas

### Contro:

- Se usate troppo posso appesantire notevolmente il caricamento della pagina
- Non utilizzano il DOM
- In genere, non sono accessibili perché gli screen reader si basano su DOM
  - <http://www.w3.org/WAI/PF/html-task-force>

### Pro:

- Sono l'unico modo per generare immagini dinamicamente
- Sono una buona alternativa a SVG



## Audio

- HTML5 permette di supportare la riproduzione di file audio in modo nativo

```
<audio src="song.mp3" autoplay loop controls />
```

```
<audio controls="controls">  
  <source src="song.ogg" type="audio/ogg" />  
  <source src="song.mp3" type="audio/mp3" />  
  <p>Testo sostitutivo dell'audio.</p>  
</audio>
```



## Video

- HTML5 permette di supportare la riproduzione di file audio/video in modo nativo
- Il funzionamento è simile al tag audio con gli stessi attributi.

```
<video width="320" height="240" controls="controls"  
  poster="img.jpg">  
  <source src="movie.mp4" type="video/mp4" />  
  <source src="movie.ogg" type="video/ogg" />  
  <p>Testo alternativo al video.</p>  
</video>
```



## Video + stile





## Lavorare in locale

- ❑ Per salvare i dati di un client prima si potevano utilizzare i cookies, ma questi erano molto limitati.
- ❑ HTML5 offre tre diverse alternative: *Web Storage*, *Web SQL Database* e *IndexedDB*
- ❑ *Web Storage* offre due oggetti sessionStorage e localStorage che memorizzano i dati sotto forma di coppie <nome, valore>
- ❑ *Web SQL Database* è un database relazionale
- ❑ *IndexedDB* si basa su una memorizzazione basata su oggetti indicizzati molto veloce ed efficiente



## Lavorare Offline: cache manifest

- ❑ La crescente diffusione dei dispositivi mobili richiede la necessità di sviluppare applicazioni che possono lavorare offline, ovvero senza un costante collegamento alla rete
  - Gmail, Calendar
- ❑ Il download delle risorse che saranno disponibili anche in assenza di rete avviene in modo trasparente all'utente
- ❑ Il file *.manifest*, chiamato anche *cache manifest*, elenca le risorse disponibili anche in assenza di connessione alla rete
- ❑ La prima riga deve contenere la stringa **CACHE MANIFEST**
- ❑ I commenti si esprimono con # e devono apparire su una riga a parte
- ❑ Il file è organizzato in sezioni, quella predefinita si chiama **CACHE:** poi ci sono **NETWORK:** e **FALLBACK:**



## Esempio file .manifest

CACHE MANIFEST

# versione 0.1

CACHE:

Risorsa1.html

Risorsa2.html

NETWORK:

Aggiorna.cgi

FALLBACK:

Online.html Offline.html

/news/\* avviso.html



## Esempio Risorsa1.html

```
<!DOCTYPE html>
```

```
<html manifest="risorsa.manifest" >
```

```
...
```

```
</html>
```

- ❑ Il file che contiene il riferimento al file .manifest viene comunque conservato in locale anche se non presente nel file .manifest
- ❑ Il file .manifest deve essere servito con il tipo MIME corretto, ovvero text/cache-manifest



## Bibliografia - XHTML

- Specifiche W3C
  - <http://www.w3.org/TR/xhtml1/>
- Validatore
  - <http://validator.w3.org/>
- Esempi utilizzati
  - <http://www.htmldog.com/examples/>
- Tutorial
  - <http://www.htmldog.com/>
  - <http://www.w3schools.com/xhtml/default.asp>
  - <http://xhtml.html.it/guide/leggi/52/guida-xhtml/>  
(italiano)



## Bibliografia - HTML5

- Specifiche W3C
  - <http://www.w3.org/TR/html5/>
- Tutorial
  - <http://www.w3schools.com/html5/default.asp>
  - [https://developer.mozilla.org/en/Canvas\\_Tutorial](https://developer.mozilla.org/en/Canvas_Tutorial)  
(Tutorial sui Canvas)
  - <http://www.html5rocks.com/en/> Tutorial e esempi
  - <http://www.html5rocks.com/en/mobile/mobifying/>
- Altri siti di riferimento
  - <http://html5doctor.com/>
  - <http://diveintohtml5.info/> "*Dive into HTML5*" Mark Pilgrim

