



HTML



1. World Wide Web

Il World Wide Web

- World Wide Web = sistema di accesso a Internet basato sul protocollo HTTP
 - insieme di protocolli e servizi (HTTP, FTP, ...)
 - insieme di tool per l'accesso (es. web browser)
- Basato sulla metafora dell'**ipertesto**
 - linguaggio HTML
- Distinzione tra Internet e WWW
 - Internet = rete
 - WWW = sistema di accesso alla rete

Architettura del web

Architettura client-server:

- Web client (es. web browser)
 - inoltra richieste di **risorse** (documenti, file, ecc.) ad una macchina (web server)
- Web server
 - risponde alle richieste dei web client
- Sia il web server che il web client sono programmi in esecuzione su macchine connesse a Internet
- Web server e web client comunicano in base al protocollo HTTP

Architetture client-server

- Basate sul concetto di richiesta di servizio (client) e di fornitura di servizio (server)
- Enormemente diffuse in informatica, in particolare nelle applicazioni di rete
 - HTTP
 - FTP
 - DNS
 - PPP
 - Proxy
 -

Protocolli

- Protocollo = insieme di convenzioni (o regole) per lo scambio di informazioni
- protocolli di “basso” livello per Internet:
 - determinano le modalità di comunicazione tra i nodi della rete
 - TCP/IP
- protocolli di “alto” livello:
 - determinano il formato dei messaggi e le modalità di scambio dei messaggi
 - HTTP, FTP, SMTP, TELNET...

Il protocollo HTTP

HTTP = HyperText Transfer Protocol

- Client-server
 - ogni interazione è una richiesta (messaggio ASCII) seguita da una risposta (messaggio tipo MIME)
- 7 metodi nativi:
 - GET
 - HEAD
 - PUT
 - POST
 - DELETE
 - LINK
 - UNLINK

Il protocollo HTTP

- Client-server
- HTTP è connectionless = **non** si instaura una connessione prima di richiedere un servizio
- FTP:
 - richiesta connessione
 - instaurazione connessione
 - richiesta servizio 1
 - fornitura servizio 1
 - richiesta servizio 2 ...
 - chiusura connessione
- HTTP:
 - richiesta servizio
 - fornitura servizio

URL

- In HTTP ogni interazione è relativa ad una URL (Uniform Resource Locator)
- La URL è un nome che identifica univocamente ogni risorsa disponibile sul web
- Ogni URL specifica:
 - il protocollo da utilizzare per il trasferimento della URL
 - il dominio, cioè il nome (simbolico) del computer su cui si trova il server (web server o altro server) che gestisce la risorsa
 - il nome, all'interno del dominio, della risorsa che si vuole accedere

Domain Name System (DNS)

- Sistema per introdurre nomi logici (o simbolici) ai computer su Internet
- IP (Internet Protocol):
 - l'indirizzo del computer è una sequenza di 4 numeri da 0 a 255
 - es. 151.100.16.20
- DNS:
 - l'indirizzo del computer è una stringa di caratteri
 - es. www.dis.uniroma1.it
- Il DNS è basato sul concetto di **dominio**

Domini

Domini radice o di primo livello:

- COM, ORG, NET, EDU, MIL, GOV, INT
- biletterale per ogni nazione (es. IT)

Per ogni dominio di primo livello:

- domini di secondo livello (es. IBM.COM, VIRGILIO.IT)
- ogni dominio di primo livello gestisce in modo autonomo i propri domini secondari
- domini di terzo livello, quarto, ecc.

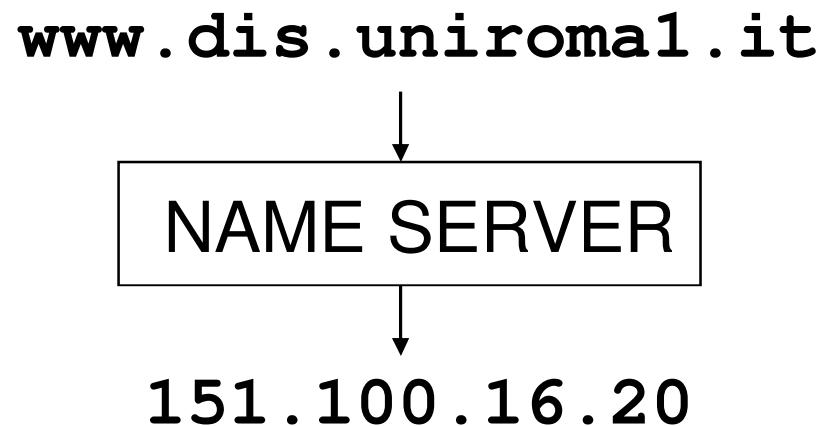
i nomi dei domini sono case-insensitive

Name Server

- Occorre tradurre il nome simbolico in indirizzo IP
 - NAME SERVER (o Domain Name Server)
 - Si deve ricorrere ad un name server ogniqualvolta si fa uso di un nome simbolico
 - Es. ogni web browser che richiede una URL, deve **prima** richiedere ad un name server l'indirizzo IP corrispondente al dominio nella URL:
- Dominio → Name Server → indirizzo IP

URL: esempio

URL: `http://www.dis.uniroma1.it/~rosati/index.htm`



- `~rosati/index.htm` è il nome (completo di percorso) del file corrispondente alla URL

Tipi di URL

Protocolli HTTP e FTP:

- Documenti ipertestuali (file HTML)
- Immagini
- Documenti multimediali (audio e video)
- Programmi
- File di altro genere

Altri protocolli: (es. mailto:)

- indirizzi di email
-

Domini, siti web e pagine web

Distinzione tra domini, web server, siti web e pagine web:

- dominio = computer dove gira un web server (“identifica” il web server)
- **sito web** = insieme di URL gestite da un’unica entità e organizzate in modo da essere accedute secondo un ordine logico
 - più siti web possono essere gestiti dallo stesso web server
- **pagina web** = singolo documento HTML
- **home page** = pagina iniziale di un sito web

2. Ipertesti e HTML

Ipertesti

Ipertesto = documento contenente:

- testo
- immagini
- audio
- video
- **collegamenti ipertestuali** = riferimenti ad altri documenti (o parti di documenti) ipertestuali

Collegamenti ipertestuali

- Sul World Wide Web, un collegamento ipertestuale ad un documento può essere specificato tramite la URL che corrisponde al documento
- Si possono usare le tecnologie informatiche per **accedere direttamente** ai documenti corrispondenti ai link mentre si legge un ipertesto
 - (esempio: web browser)
 - superamento dell'accesso “sequenziale” al testo

Linguaggio per ipertesti: HTML

- HTML = HyperText Markup Language
- Linguaggio standard per la specifica di documenti ipertestuali sul World Wide Web
- Linguaggio a marcatura, “figlio” di SGML (Standard Generalized Markup Language)
- Un documento HTML può contenere:
 - testo
 - link ipertestuali
 - immagini
 - link a risorse (URL) di ogni tipo

Breve storia di HTML

- Definito (insieme ad HTTP) da Tim Berners-Lee del CERN di Ginevra nel 1989
- Scopo: permettere lo scambio dei dati sperimentali tra i fisici
- 1993: diffusione di Mosaic (web browser sviluppato da NCSA)
- 1995: fondazione del World Wide Web Consortium (W3C)
- 1999: standardizzazione di HTML 4.0
- 2000: standardizzazione di XHTML 1.0 (ridefinizione di HTML basata su XML)
- 201?: standardizzazione di HTML5

Sintassi di HTML

- Documento HTML = testo ASCII
- contiene:
 - blocchi di testo
 - tag (marcature o “comandi”)
- tag = testo delimitato dai simboli “<” e “>”
- esempio:

<nome-tag>

Il concetto di tag

- TAG = “marcatura” (o marcatore)
- Un tag viene usato per **marcare** una parte di testo
- Tag:
 - di formattazione (per cambiare l’aspetto ad una parte di testo) (es.)
 - “semantici” (per dare un “significato” ad una parte di testo) (es. <a>)
- 2 tipi di tag:
 - tag di apertura (marcatore iniziale) <nome-tag>
 - tag di chiusura (marcatore finale) </nome-tag>

Attributi dei tag

- Ogni occorrenza di tag (di apertura) può contenere assegnazioni di **attributi**
- ogni tag ha un diverso insieme di possibili attributi
- assegnazione: nome-attributo = valore
 - es.
- struttura del tag con attributi:

<nome-tag attributo1 = valore1 attributo2 = valore2>

- alcuni attributi del tag sono **obbligatori** (vanno assegnati)

Elementi e tag

- Elemento = insieme formato da tag di apertura, testo e tag di chiusura corrispondente
- Esempio di elemento (font):
`ciao `
- In genere si usa il termine “tag” erroneamente, per indicare un elemento (composto da due tag)

Per HTML i tag sono case-insensitive (es.
`` e `` hanno lo stesso significato)

Semantica di HTML

Il “significato” di un documento HTML è dato da due componenti:

- aspetto del documento
- “contenuto” (rispetto ai tag) del documento

La semantica “immediata” di un documento HTML è la sua visualizzazione sul browser

- dipendente dal browser
- perdita (parziale) del significato legato al “contenuto”

Struttura di un documento HTML

```
<html>          <-- inizio del documento  
<head>  
...  
...          <-- intestazione del documento  
</head>  
<body>  
...          <-- corpo del documento  
...  
</body>  
</html>         <-- fine del documento
```

Informazione e meta-information

- corpo del documento = contiene l'informazione (il documento stesso)
- intestazione del documento = contiene **meta-information** (cioè informazioni **sul** documento)
 - esempi:
 - autore del documento
 - parole chiave
 - “titolo” del documento
 -

Contenuto e presentazione

- Problema: distinguere tra
 - **contenuto** del documento
 - **presentazione** (o aspetto) del documento
- E' molto importante poter individuare il contenuto di un documento indipendentemente dalla formattazione del documento
- Nelle intenzioni, HTML doveva mantenere separati i due aspetti. Nella realtà, non è così:
 - i marcatori sono usati anche per dare attributi di formattazione al testo
 - i vari browser “interpretano” il codice HTML in modo diverso

HTML e browser HTML

I principali web browser, specie in passato, hanno influito sull'evoluzione di HTML:

- imponendo nuovi elementi del linguaggio
- “rifiutando” (cioè non supportando) nuovi elementi del linguaggio
- Problemi principali:
 - differente interpretazione di HTML
 - presenza di vecchie versioni dei browser

Creare documenti HTML

Documento HTML = testo ASCII
(analogo ad un programma sorgente JAVA)

Modalità di creazione di un documento HTML:

- con un editor per testo ASCII (es. blocco note o Wordpad sotto Windows)
- con un “editor WYSIWYG” o “editor HTML” (es. FrontPage, Dreamweaver)
- con un editor di testi “normale” che permette di esportare i documenti in HTML (es. Word)

Editor HTML

- Permette di editare il documento vedendo direttamente come verrà visualizzato dal browser
- Facilità di utilizzo:
 - non è necessario conoscere il linguaggio HTML
- Limiti nella realizzazione:
 - non tutte le potenzialità di HTML possono essere utilizzate
- Editor HTML professionali possono essere utilizzati al massimo solo conoscendo il codice HTML

3. HTML di base

Sommario

- intestazione
- formattazione testo
- link
- oggetti, immagini e applet
- tavelle
- frame

Parte intestazione

- contiene una serie di informazioni necessarie al browser per una corretta interpretazione del documento, ma non visualizzate all'interno dello stesso:
- tipo di HTML supportato
- titolo della pagina
- parole chiave (per motori di ricerca)
- link base di riferimento
- stili (comandi di formattazione)

Parte intestazione

Elementi principali:

- DOCTYPE
- HTML
- HEAD
- TITLE
- META

Parte intestazione

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">

<HTML>

<HEAD>

<META name="keywords" Content= "HTML, parte
intestazione, meta-information">
<META name= "author" Content = "Riccardo Rosati">
<meta name="GENERATOR" content="Blocco note di
Windows 98">

<TITLE>Pagina web di prova </TITLE>

</HEAD>

.....
</HTML>
```

Parte intestazione

DOCTYPE:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
```

- fornisce informazioni sul tipo di documento visualizzato
- deve essere il primo elemento ad aprire il documento
- non è obbligatorio

Parte intestazione

META:

```
<META name="keywords" Content= "HTML, parte  
intestazione, meta-information">  
<META name= "author" Content = "Riccardo Rosati">  
<meta name="GENERATOR" content="Blocco note di  
Windows 98">
```

- fornisce meta-information sul contenuto del documento
- usate dai motori di ricerca per classificare il documento
- non è obbligatorio

Parte intestazione

TITLE:

<TITLE>Pagina web di prova </TITLE>

- Titolo della pagina
- Compare sulla barra del titolo della finestra del browser
- Usato dai motori di ricerca
- Usato nella visualizzazione di “bookmark” (siti preferiti)

HTML di base

- intestazione
- **formattazione testo**
- link
- oggetti, immagini e applet
- tavole
- frame

Corpo del documento

- Memorizza il contenuto del documento (la parte visualizzata all'utente del browser)

```
<body ..... >
```

```
.....
```

```
</body>
```

- gli attributi di `<body>` configurano alcuni parametri di visualizzazione del documento

Attributi di <body>

- BACKGROUND: setta lo sfondo
- BGCOLOR: setta il colore di sfondo
- TEXT: setta il colore del testo
- LINK, VLINK, ALINK: settano il colore del testo corrispondente ai link:
 - LINK: link non visitato
 - VLINK: link già visitato
 - ALINK: link “attivo”

Colori del corpo: formato RGB

Esempio:

```
<BODY BGCOLOR="red">
```

identico a

```
<BODY BGCOLOR="#ff0000">
```

- Colori in formato RGB = 3 numeri (componenti rossa, verde e blu)
- ogni colore va da 0 a 255 (FF in esadecimale)
 - es. #000000 = nero, #FFFFFF = bianco, #777777 = grigio, #00FF00 = verde, #0000FF = blu

Esempio:

```
<body background="sfondo.jpg" link="blue"  
vlink="red" alink="red">
```

- BACKGROUND fa riferimento ad una URL (nell'esempio un file locale) contenente una immagine
- l'immagine viene disegnata (eventualmente in modo ripetuto) per riempire lo sfondo del documento

Nota bene

Per separare forma e contenuto:

- gli attributi di formattazione NON dovrebbero comparire nel corpo del documento
- gli attributi di `<body>` NON dovrebbero essere usati
- nella pratica vengono usati, ma il W3C “depreca” ufficialmente tale utilizzo (come anche, ad esempio, l’uso di ``)
- alternativa: uso di XML = formattazione del tutto separata dal contenuto

Header

- <H1>,<H2>,...,<H6>
- Permettono di inserire titoli (o intestazioni) all'interno del documento
- il testo tra <Hn>...</Hn> viene evidenziato dal browser
- 6 livelli di titoli
- 6 differenti livelli di enfatizzazione del testo

Enfatizzare il testo

, <I>, <U>:

- (bold):
 - permette di visualizzare testo in neretto
- <I> (italic):
 - permette di visualizzare testo in corsivo
- <U> (underlined):
 - permette di sottolineare il testo
- definiscono attributi **fisici** di formattazione

Attributi logici e fisici

- tag fisico = ha il compito di formattare il documento
- tag logico = dà una struttura al documento, ed è indipendente dalla visualizzazione
- esempio: elemento `<address>`
 - permette di specificare che una parte di testo è un indirizzo
 - viene visualizzato come testo in corsivo
 - per il browser è come usare `<i>`
 - ma `<ADDRESS>` aggiunge “semantica” al documento

Selezione del font

- Tag fisico
- deprecato da W3C
- definisce il modo in cui deve essere visualizzata una parte di testo:
 - tipo di carattere
 - colore
 - grandezza

Attributi del tag

- FACE
 - determina il tipo di carattere (font) usato
 - font disponibili: courier, times, arial, verdana, ...
- SIZE
 - determina la grandezza dei caratteri
 - si esprime con numeri assoluti (da 1 a 7) o relativi
- COLOR determina il colore

esempio:

```
<FONT FACE="verdana" SIZE="+1" COLOR="green">  
testo in verde</FONT>
```

Apici e pedici

- <SUB> (subscript)
 - permette di scrivere testo come “pedici”
- <SUP> (superscript)
 - permette di scrivere “apici”
- esempio:

I₅ = 2⁴

viene visualizzato come

$$I_5 = 2^4$$

Testo preformatto

- Tag <XMP> e <PRE>
- permettono di visualizzare il testo esattamente come è scritto (“preformatto”) nel file sorgente HTML
- il testo preformatto non viene “interpretato”
- <XMP> non interpreta neanche le occorrenze dei tag HTML dentro al testo preformatto
- con <PRE> invece le occorrenze di tag HTML vengono interpretate
 - <PRE> è il tag di preformattazione di riferimento per HTML 4.0

Testo preformatto con <XMP>

Codice HTML:

```
<XMP>
begin
  if
    then
end;
      I<SUB>5</SUB>      =  2<SUP>4</SUP>
</XMP>
```

Visualizzazione:

```
begin
  if
    then
end;
      I<SUB>5</SUB>      =  2<SUP>4</SUP>
```

Testo preformatto con <PRE>

Codice HTML:

```
<PRE>
begin
  if
    then
end;
      I<SUB>5</SUB>      =  2<SUP>4</SUP>
</PRE>
```

Visualizzazione:

```
begin
  if
    then
end;
      I5      =  24
```

Stili logici

- <ADDRESS>
 - marcatura usata per indirizzi (mail, email, telefono,...)
- <BLOCKQUOTE>
 - usato per inserire nel testo citazioni da un altro testo o autore
- <CITE>
 - usato per la fonte della citazione
- e
 - “enfatizzano” il testo all’interno del tag
- <VAR> e <CODE>
 - utilizzati per righe di codice di programmazione

Separare e allineare il testo

- <P> (paragraph)
 - crea un paragrafo all'interno del testo
-
 (break)
 - interruzione di riga (“a capo”)
- <DIV>
 - usato per allineare il documento:
 - <DIV align = left> allinea a sinistra il testo
 - <DIV align = center> allinea al centro il testo
 - <DIV align = right> allinea a destra il testo
- <CENTER> tag non standard

Righe orizzontali

- <HR> (horizontal row)
 - aggiunge una riga orizzontale al testo
 - usato per separare parti di testo
- attributi di <HR>:
 - ALIGN (left|center|right|) allineamento rispetto alla pagina
 - WIDTH lunghezza orizzontale (in pixel o in percentuale)
 - SIZE altezza della riga in pixel
 - COLOR colore della riga
 - NOSHADE elimina l'effetto 3D

Liste puntate

- (unordered list)
 - produce un elenco (lista) di elementi (parti di testo)
 - ogni elemento è evidenziato all'inizio da un simbolo grafico (di solito cerchietto o quadratino)
- (list item)
 - per identificare un elemento della lista
- esempio:

```
<UL>
  <LI>Primo elemento </LI>
  <LI>Secondo elemento </LI>
  <LI>Terzo elemento </LI>
</UL>
```

Liste numerate

- (ordered list)
 - produce un elenco (lista) di elementi (parti di testo)
 - ogni elemento è evidenziato all'inizio dal numero d'ordine all'interno della lista
- (list item) (come per liste puntate)
- esempio:

```
<OL>
<LI>Primo elemento </LI>
<LI>Secondo elemento </LI>
<LI>Terzo elemento </LI>
</OL>
```

Liste numerate

Oltre al numero progressivo, si possono usare altre indicizzazioni per le liste puntate

Uso dell'attributo TYPE di :

- <OL TYPE=A> indicizza con lettere alfabetiche maiuscole
- <OL TYPE=a> indicizza con lettere alfabetiche minuscole
- <OL TYPE=I> indicizza con numeri romani maiuscoli
- <OL TYPE=i> indicizza con numeri romani minuscoli

Esempio

Esempio di liste annidate:

```
<ol>
<li>gruppo di nomi:
<ul>
<li>nome a </li>
<li>nome b </li>
</ul></li>
<li>gruppo di nomi:
<ul>
<li>nome c </li>
<li>nome d </li>
<li>nome e </li>
</ul></li>
</ol>
```

HTML di base

- intestazione
- formattazione testo
- link
- oggetti, immagini e applet
- tavole
- frame

Link ipertestuali

Tag <A> (anchor)

- permette l'inserimento di link ipertestuali all'interno del documento
- attributo principale: **Href** (Hypertext reference)
 - specifica la URL che viene associata al link
- il testo tra <A> e viene associato a tale URL
 - cliccando su tale testo il browser accede alla URL
- **Esempio:**

```
<A HREF="http://www.virgilio.it">Visita  
virgilio.it</A>
```

Attributi del tag <a>

- 2 tipi di tag <a>:
- con attributo HREF:
 - aggiungono un link ipertestuale (esterno o interno al documento)
- con attributo NAME:
 - definiscono uno specifico punto del documento come un link interno
 - tale punto può essere direttamente acceduto attraverso un tag <A HREF...>
- altri attributi: TARGET, TITLE

Esempio

...

```
<a name="zona1"> zona 1 del documento  
raggiungibile direttamente </a>
```

...

```
<a href="#zona1">torna alla zona 1 del  
documento</a>
```

...

**Con l'anchor zona1 si può anche accedere direttamente
dall'esterno del documento:**

```
<a href="www.dis.uniroma1.it/index.html#zona1">  
vai alla zona 1 del documento index.html del sito  
dis.uniroma1.it </a>
```

Attributo target di <a>

Attributo TARGET di <A>:

- permette di specificare dove deve essere visualizzata la URL associata al link

valori principali di TARGET:

- _NEW: visualizza la URL collegata in una nuova finestra del browser
- _PARENT: visualizza nella stessa finestra, eliminando tutti i frame presenti (vedere sezione sui frame)

Attributo title di <a>

Attributo TITLE di <A>:

- permette di specificare una informazione associata al link
- es. commento riguardante il link
- i browser visualizzano tale informazione (pop-up) quando il puntatore del mouse passa sopra al link

HTML di base

- intestazione
- formattazione testo
- link
- **oggetti, immagini e applet**
- tavole
- frame

Immagini

- HTML permette di inserire immagini nel documento
- Tag (singolo)
- permette di inserire nel documento una immagine, memorizzata in un file (o URL) a parte
- i browser riconoscono i principali formati immagine (es. JPG, GIF, BMP)

Attributi del tag

- SRC
 - specifica il nome della URL contenente l'immagine
 - es.
- WIDTH larghezza (in pixel o percentuale)
- HEIGHT altezza (in pixel o percentuale)
 - se WIDTH e HEIGHT mancano, l'immagine viene visualizzata nelle sue dimensioni originali
- ALT
 - permette di aggiungere un commento testuale associato all'immagine

Attributi del tag

- BORDER
 - spessore cornice dell'immagine (in pixel)
- HSPACE e VSPACE
 - distanze minime orizzontali e verticali (in pixel) dell'immagine dagli oggetti più vicini
- ALIGN
 - determina l'allineamento del testo rispetto all'immagine

L'attributo ALT

- Permette di aggiungere una informazione testuale all'immagine
- Il testo viene visualizzato dai browser (popup)
- Necessario per i browser solo testuali
- Oppure per la navigazione con immagini disabilitate

es.

L'attributo ALIGN

- determina l'allineamento del testo rispetto all'immagine
 - **ALIGN=top**: allinea la prima riga di testo sulla sinistra al top dell'immagine
 - **ALIGN=middle**: allinea la prima riga di testo sulla sinistra al centro dell'immagine
 - **ALIGN=bottom**: allinea la prima riga di testo sulla sinistra nella parte più bassa dell'immagine
 - **ALIGN=left**: allinea il testo sulla destra dell'immagine partendo dal top
 - **ALIGN=right**: allinea il testo sulla sinistra dell'immagine partendo dal top

Mappe cliccabili

- Una immagine può essere associata ad un link
- es. ``
- spesso si vogliono associare due o più link alla stessa immagine
 - associare zone diverse dell'immagine a diversi link
- **mappe cliccabili**
- 2 tipi:
 - lato server (poco diffuse)
 - lato client (USEMAP)

Mappe cliccabili (lato client)

```
<IMG SRC="img1.gif" WIDTH=400 HEIGHT=100  
      BORDER=0 usemap="#immagine1">  
  
<map name="immagine1">  
  <area shape="rect" alt="parte 1 immagine"  
        coords="0,0,200,100" href="doc1.htm"  
        title="parte 1 immagine">  
  <area shape="rect" alt="parte 2 immagine"  
        coords="201,0,400,100" href="doc2.htm"  
        title="parte 2 immagine">  
  <area shape="default" nohref>  
</map>
```

Generare mappe cliccabili

- Tipi di aree definibili con usemap:
 - rect
 - circle
 - poly
- difficili da definire a mano
- uso di programmi (editor di mappe)
 - es. MAPEDIT

Programmi e documenti HTML

Tipi più diffusi di programmi associati a pagine HTML:

- applet JAVA
- script (es. scritti in JavaScript)

applet = file (estensione .class) esterni al documento HTML

script = righe di codice scritte all'interno del documento HTML

Applet e script

- Applet = codice compilato (bytecode JAVA)
- script = codice sorgente

Il browser deve essere in grado di interpretare sia bytecode JAVA che sorgente JavaScript:

- JAVA virtual machine
- interprete JavaScript

Differenze:

- Applet non modificabile (bytecode)
- script facilmente modificabile (sorgente)

Applet

Esempio:

```
<APPLET CODE= "applet1.class" WIDTH=500  
HEIGHT = 300>
```

- L'uso di `<applet>` è deprecato in HTML 4.0
- proposta: uso di `<object>`
 - inclusione di oggetti generici (tra cui applet) nel documento HTML
 - prepara HTML per future applicazioni

Simboli speciali

- Come rappresentare simboli non-ASCII e simboli utilizzati da HTML?
- insieme di definizioni di simboli (ogni simbolo è rappresentato da un nome)
- l'invocazione di un simbolo predefinito inizia con “&” e termina con “;”
- esempio: © per rappresentare ©

Simboli speciali

- esempi: lettere accentate:
 - à à
 - è è
 - ´ é
 - ì ì
 - ò ò
 - ù ù
- il simbolo “&” si rappresenta con &

Il simbolo “<”

- < è un simbolo particolarmente speciale in HTML (apertura dei tag)
- < si rappresenta con <
- > si rappresenta con >

HTML di base

- intestazione
- formattazione testo
- link
- oggetti, immagini e applet
- **tabelle**
- frame

Tabelle

- Rappresentano informazione in forma tabellare (righe e colonne) nei documenti HTML
- Molto utilizzate anche come strumento di formattazione di testi e/o immagini
- HTML permette una gestione piuttosto potente delle tabelle

Elementi relativi alle tavelle

- TABLE
 - definisce la tabella
- TD
 - definisce un campo “dati” all’interno della tabella
- TR
 - suddivide i campi in righe all’interno della tabella
- TH
 - definisce campi intestazione all’interno della tabella
- THEAD, TFOOT

L'elemento <table>

- Racchiude tutta l'informazione relativa ad una tabella
- Esempio:

```
<TABLE WIDTH=300 HEIGHT=200>
```

```
.....
```

```
</TABLE>
```

- gli attributi di <table> settano proprietà globali della tabella

Dimensioni della tabella

Espresso in:

- pixel (punti)
 - es. `<table width=300 height=200>`
 - indipendente dalle dimensioni della finestra di visualizzazione
- percentuale della dimensione della pagina
 - es. `<table width="60%">`
 - dipendente dalle dimensioni della finestra di visualizzazione

la scelta tra i due tipi di dimensioni dipende dalla applicazione

Attributi di <table>

- WIDTH larghezza
- HEIGHT altezza (non dovrebbe essere usato)
- BORDER spessore del bordo (in pixel)
- BGCOLOR colore sfondo tabella
- SUMMARY testo che spiega il contenuto della tabella (per media non visuali)
- CELLSPACING distanza tra i campi (celle) della tabella
- CELLPADDING distanza in pixel tra il contenuto del campo e i margini del campo

L'elemento <TD>

<TD> permette la definizione di un singolo campo (cella)

- va usato per campi di tipo “dati”
- non va usato per campi di tipo intestazione di colonne
- esempio:

```
<TD width=100>prova</TD>
```

definisce una cella con contenuto prova

Attributi di <TD>

- WIDTH, HEIGHT non dovrebbero essere usati
- VALIGN (top|bottom|middle) allineamento verticale
- ALIGN (left|center|right) allineamento orizzontale
- BGCOLOR colore di sfondo della cella
- BACKGROUND motivo di sfondo della cella
- ROWSPAN, COLSPAN

L'elemento <TR>

- Divide le celle in righe
- Esempio:

```
<TABLE border=1 cellpadding=2>  
<TR>  
<TD>cella 1</TD>  
<TD>cella 2</TD>  
<TD>cella 3</TD>  
<TR>  
<TD>cella 1 riga 2</TD>  
<TD>cella 2</TD>  
<TD>cella 3</TD>  
</TABLE>
```

cella 1	cella 2	cella 3
cella 1 riga 2	cella 2	cella 3

Attributi di <TR>

- ALIGN (left|center|right) allineamento orizzontale delle celle che seguono <TR>
- VALIGN (top|middle|bottom) allineamento verticale
- BGCOLOR

Esempio

```
<TABLE WIDTH=300 HEIGHT=200>  
<TD width=100 VALIGN=TOP>  
Prova1</TD>  
<TD WIDTH=100 VALIGN=BOTTOM>  
Prova2</TD>  
<TD WIDTH=100 VALIGN=MIDDLE>  
Prova3</TD>  
</TABLE>
```

Prova1		
	Prova2	Prova3

Esempio

```
<TABLE WIDTH=300 HEIGHT=200 border=1>  
<TD width=100 ALIGN=RIGHT>  
prova1</TD>  
<TD WIDTH=100 ALIGN=CENTER>  
Prova2</TD>  
<TD WIDTH=100 ALIGN=LEFT>  
Prova3</TD>  
</TABLE>
```

prova1	Prova2	Prova3
--------	--------	--------

L'elemento <TH>

- Come <TD> ma va usato per specificare campi **intestazione**
- permette di dare più “semantica” alla tabella
- da un punto di vista di presentazione, per i browser non c’è differenza
- stessi attributi di <TD>

Esempio

```
<TABLE border=1 cellpadding=2>
<TR>
<TH>nome</TH>
<TH>cognome</TH>
<TH>età</TH>
<TR>
<TD>Mario</TD>
<TD>Rossi</TD>
<TD>36</TD>
<TR><TD>Paola</TD>
<TD>Bianchi</TD>
<TD>32</TD>
</TABLE>
```

nome	cognome	età
Mario	Rossi	36
Paola	Bianchi	32

L'elemento <CAPTION>

- Permette di associare una didascalia alla tabella
- esempio:

```
<TABLE border=1 cellpadding=2>
<CAPTION>Elenco degli impiegati</CAPTION>
<TR>
<TH>nome</TH>
<TH>cognome</TH>
<TH>età</TH>
<TR>
<TD>Mario</TD>
<TD>Rossi</TD>
<TD>36</TD>
</TABLE>
```

Elenco degli impiegati

nome	cognome	età
Mario	Rossi	36
Paola	Bianchi	32

Elementi di raggruppamento righe

- <THEAD>, <TBODY>, <TFOOT>
- Permettono di suddividere l'informazione contenuta in una tabella per righe
- Permettono la gestione separata di intestazione e contenuto
- Permettono di raggruppare il contenuto della tabella in più gruppi (più occorrenze di <TBODY>...</TBODY>)
- Struttura non visualizzata dai browser (occorrono fogli di stile)

Esempio

```
<TABLE border=1 cellpadding=2>
<THEAD>
<TR><TH>nome</TH>
<TH>cognome</TH>
<TH>et&agrave;;</TH>
</THEAD>
<TBODY>
<TR><TD>Mario</TD>
<TD>Rossi</TD>
<TD>36</TD>
<TR><TD>Paola</TD>
<TD>Bianchi</TD>
<TD>32</TD>
</TBODY>
</TABLE>
```

Raggruppamento delle colonne

- <COLGROUP>, <COL>
- Permettono di suddividere l'informazione contenuta in una tabella per colonne
- Struttura non visualizzata dai browser (occorrono fogli di stile)

Celle variabili

- Una cella può occupare più righe o più colonne di una tabella
- Attributi ROWSPAN, COLSPAN di <TD>
- ROWSPAN = numero righe occupate dalla cella
- COLSPAN = numero colonne occupate dalla cella

Esempio

```
<TABLE border=1 cellpadding=2>  
<TR>  
<TH>nome</TH>  
<TH>cognome</TH>  
<TH>età</TH>  
<TR>  
<TD colspan=2>Rossi</TD>  
<TD>36</TD>  
<TR><TD>Paola</TD>  
<TD rowspan=2>Bianchi</TD>  
<TD>32</TD>  
<TR><TD>Maria</TD>  
<TD>36</TD>  
</TABLE>
```

nome	cognome	età
Rossi		36
Paola		32
Maria	Bianchi	36

HTML di base

- intestazione
- formattazione testo
- link
- oggetti, immagini e applet
- tavole
- **frame**

Frame

- Frame = riquadro (zona rettangolare) della finestra di visualizzazione del browser
- ogni frame è gestito in modo indipendente dal browser (come una finestra a sé stante)
- In HTML è possibile “organizzare” più documenti in un insieme di frame
 - i documenti sono presentati in un’unica schermata (finestra) divisa in frame
 - ogni documento è visualizzato in un diverso frame
 - la presentazione sfrutta la divisione in “sottofinestre”

Ha senso utilizzare i frame?

- I frame aiutano a migliorare la fruizione di un sito
- ma: molti navigatori su web “odiano” i frame
- E’ possibile avere “annidamenti” di frame, che rendono difficile la fruizione delle pagine
- I frame rendono impossibile la navigazione per alcuni media (per esempio per i non vedenti)

Nel caso si usino i frame, è buona norma prevedere **sempre** una versione senza frame dei documenti

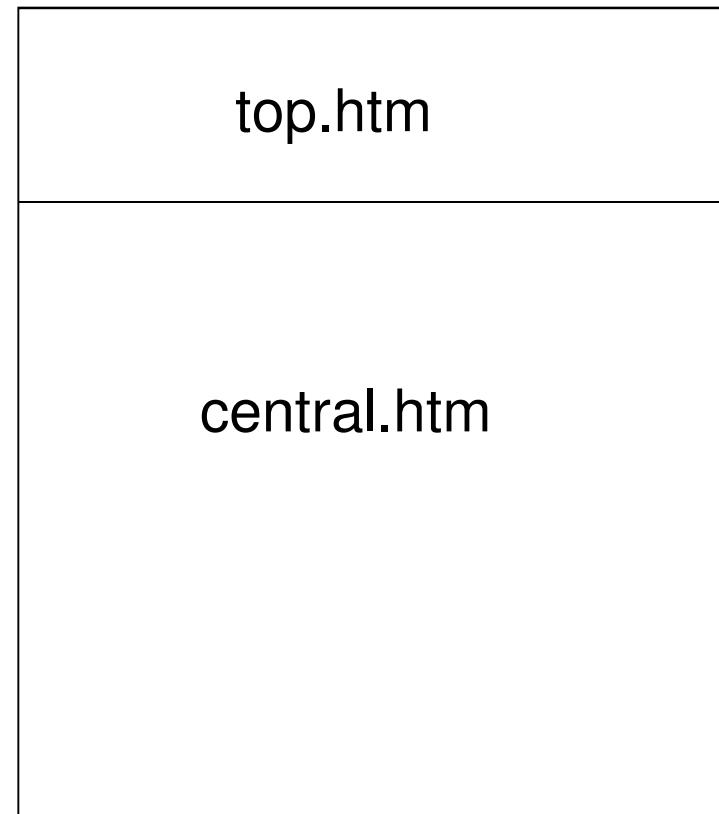
Frame principale

- Per creare una pagina divisa in frame è necessario creare più files HTML richiamati da un file principale
- Il documento principale contiene l'elemento <frameset>

```
<FRAMESET rows="80, *">  
    <frame name="alto" src="top.htm">  
    <frame name="centrale"  
        src="central.htm">  
</FRAMESET>
```

Frame

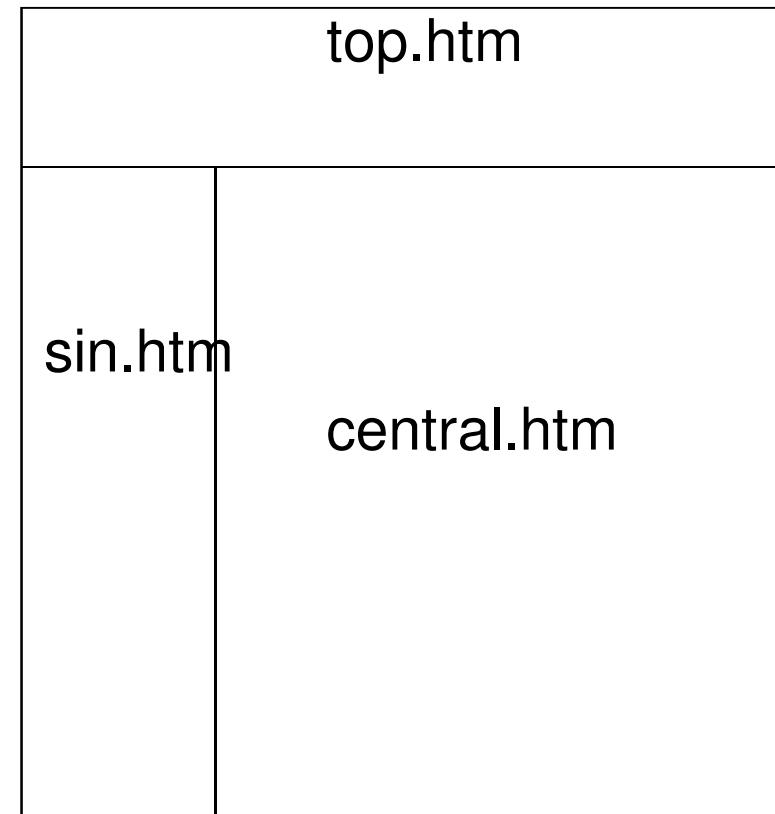
```
<FRAMESET rows="80, *">  
  <frame name="alto"  
        src="top.htm">  
  <frame name="centrale"  
        src="central.htm">  
</FRAMESET>
```



Dimensioni dei frame

- righe (rows)
- colonne (cols)

```
<frameset rows="100, *">  
<frame name="alto"  
       src="top.htm">  
<frameset cols="150, *">  
<frame name="sx"  
       src="sin.htm">  
<frame name="centrale"  
       src="central.htm">  
</frameset>  
</frameset>
```

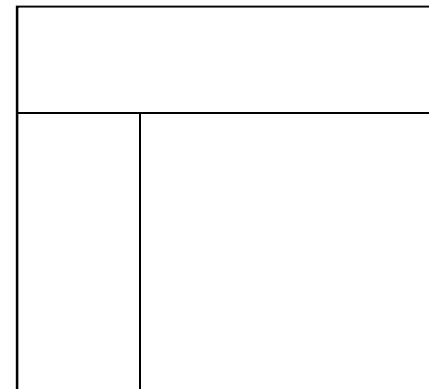
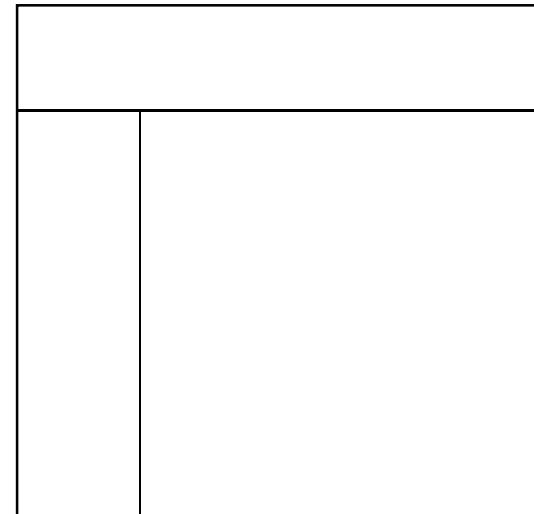


Dimensioni dei frame

- Dimensioni assolute:
 - espresse come numero di punti (pixel)
 - non cambiano se cambiano le dimensioni della finestra del browser
 - es. <FRAMESET rows="80, *">
- Dimensioni relative:
 - espresse come percentuale della dimensione corrente della finestra
 - cambiano al variare della dimensione della finestra
 - es: <FRAMESET rows="20%, *">

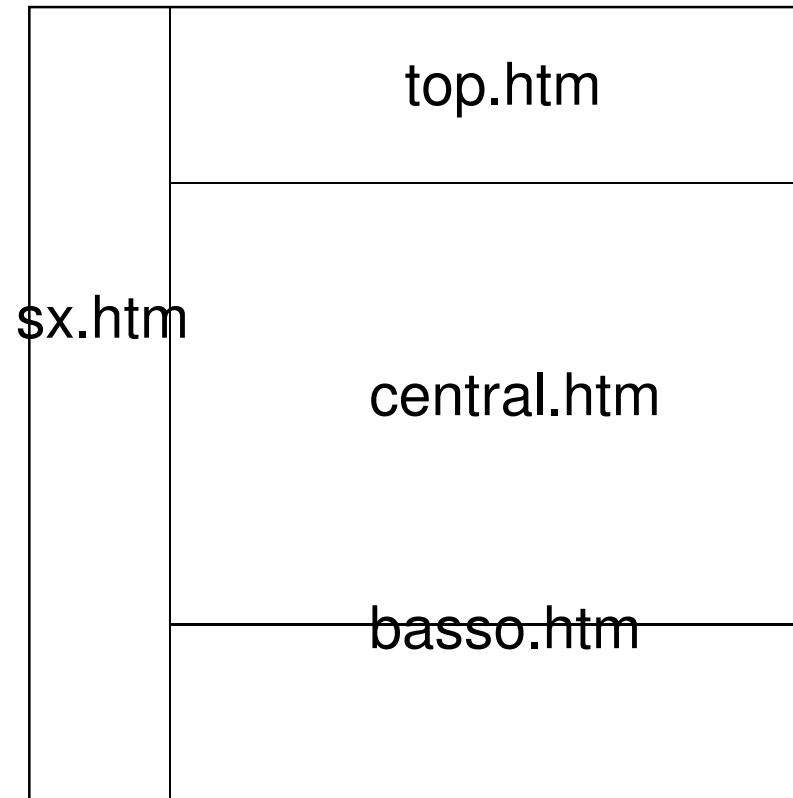
Esempio

```
<frameset rows="100, *">  
  <frame name="alto"  
        src="top.htm">  
  <frameset cols="150, *">  
    <frame name="sx"  
          src="sin.htm">  
    <frame name="centrale"  
          src="central.htm">  
  </frameset>  
</frameset>
```



Esempio

```
<frameset cols="120, *">  
  <frame name="sx"  
        src="sx.htm">  
  <frameset  
        rows="20%, 60%, 20%, *">  
    <frame name="alto"  
          src="top.htm">  
    <frame name="centrale"  
          src="central.htm">  
    <frame name="basso"  
          src="basso.htm">  
  </frameset>  
</frameset>
```



Elementi relativi ai frame

- <FRAMESET>
 - sostituisce l'elemento <BODY> nel frame principale
- <FRAME>
 - serve ad importare i frame secondari dal frame principale
- <NOFRAMES>
 - serve a specificare un documento **alternativo** al frame

Attributi di <frameset>

- **ROWS** altezza
- **COLS** larghezza
- attributi della cornice:
 - **FRAMEBORDER** (= yes | no) presenza della cornice
 - **BORDER** spessore della cornice
 - BORDER = 0 elimina la cornice
 - **BORDERCOLOR** colore della cornice
- esempio:

```
<frameset cols="120, *" rows="120, *"  
bordercolor="#FF0000" border="5">
```

Attributi di <frame>

- **SRC** (search) URL da caricare nel riquadro
- **NAME** denominazione del frame
- **SCROLLING** (yes|no|auto) tipo di barra di scorrimento nel riquadro
- **MARGINWIDTH, MARGINHEIGHT**
larghezza e altezza dei margini (risp. distanza dal margine alto e dal margine sinistro)
- **NORESIZE** dimensione non modificabile
- **BORDER, BORDERCOLOR** spessore e colore del margine

Nomi dei frame

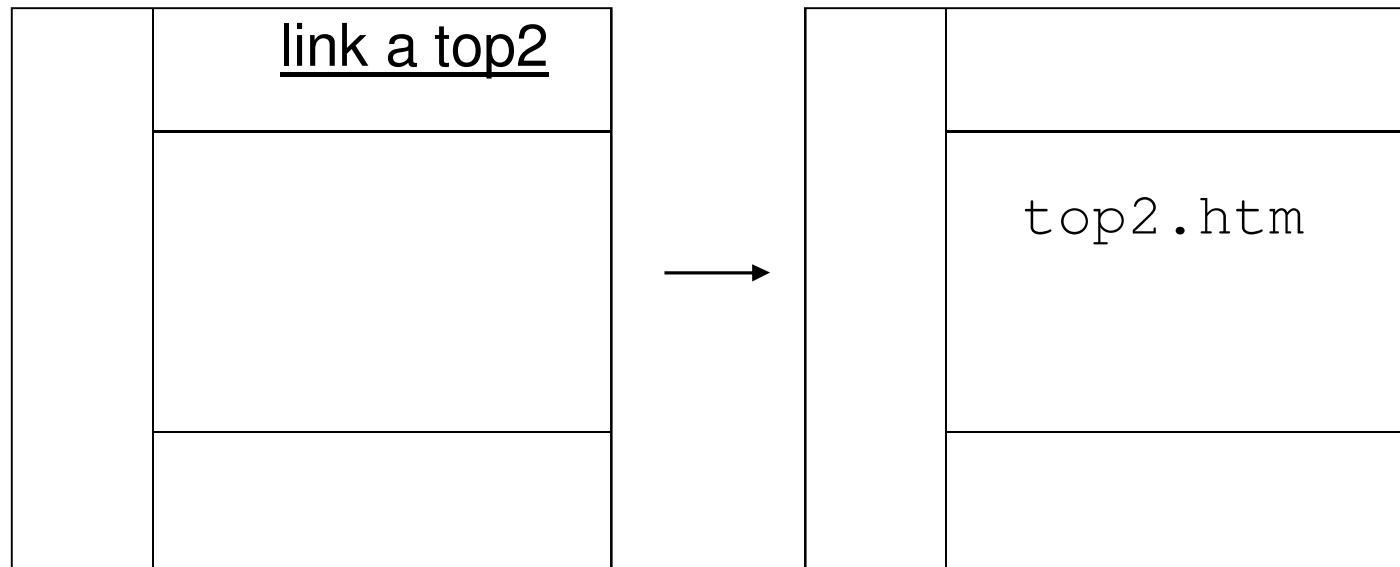
- L'attributo NAME dà un nome al riquadro
- Tale nome è usato per **indirizzare** il caricamento di una URL in un particolare riquadro
- Per fare questo si assegna il nome del frame all'attributo TARGET dell'elemento <A>
- es:

```
<A href="top2.htm" target = "centrale">  
    carica il documento top2.htm nel frame di nome  
    centrale (se esiste)
```

Esempio

- Contenuto di top.htm:

```
<html>  
<head> </head>  
<body>  
  <a href="top2.htm" target="centrale">link a  
    top2</a>  
</body>  
</html>
```

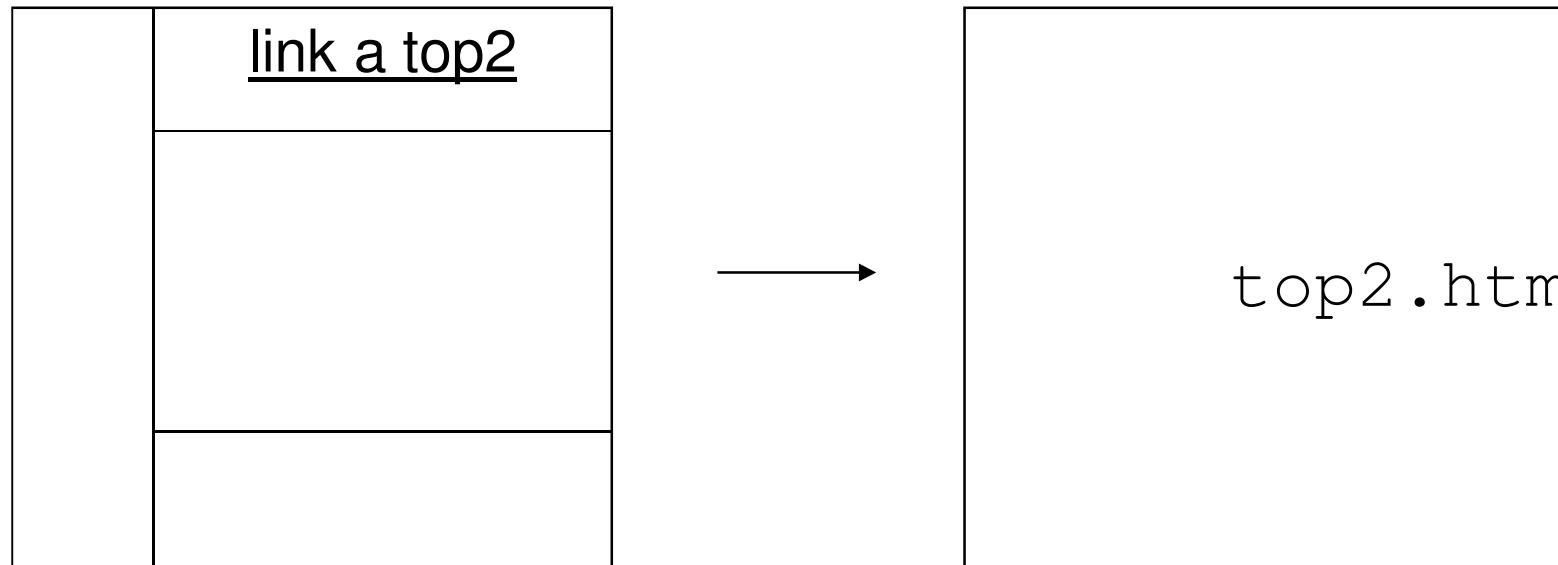


Esempio (cont.)

- Uso del valore `_parent`:

```
<a href="top2.htm" target="_parent">link a  
top2</a>
```

l'attivazione di questo link elimina tutti i frame dalla finestra



Caricare più frame

- E' possibile con una sola operazione effettuare il caricamento simultaneo di più pagine in due o più frame
- Per tale operazione è necessario uno **script** (per esempio in JavaScript)

Esempio

```
<HEAD>
<script language="JavaScript">
<!-- Hiding
function loadtwo(page1, page2) {
parent.alto.location.href=page1;
parent.centrale.location.href=page2; }
// -->
</script>
</HEAD>

<BODY>
<FORM NAME="buttons">
<INPUT TYPE="button" VALUE="Clicca"
onClick="loadtwo ('nuovo1.htm', 'nuovo2.htm') ">
</FORM>
</BODY>
```

L'elemento <noframes>

- Necessario per permettere l'accesso al documento a tutti i media per cui **non** ha senso la nozione di frame
 - vecchie versioni dei browser
 - sistemi di navigazione web per non vedenti
 - wap browser, ecc.
- struttura:

```
<noframes>
```

codice HTML alternativo ai frame

```
</noframes>
```

Esempio

```
<frameset rows="100,*">
<frame name="alto" src="top.htm">
<frameset cols="150,*">
<frame name="sx" src="sin.htm">
<frame name="centrale" src="central.htm">
</frameset>
<noframes>
<html><body>
Attenzione! il tuo browser non supporta l'opzione
frame. Per visualizzare queste pagine &egrave;
necessario un browser recente
</body></html>
</noframes>
</frameset>
```

L'elemento <iframe>

- Questo elemento («inline frame») permette di definire un frame in qualsiasi punto di un documento HTML
- struttura:

```
<iframe src="http://www.uniroma1.it"  
width="500" height="300">  
codice HTML alternativo (per i browser  
che non supportano iframe  
</iframe>
```

4. HTML avanzato

Sommario

- **forms**
- fogli di stile (CSS)
- HTML “dinamico”
- cenni su HTML5

Form (modulo)

- Usati per inviare informazioni via WWW
- Il modulo viene compilato dall'utente (sul browser)
- quindi viene inviato al server
- un programma apposito sul server (es. un CGI) elabora il modulo
- in genere il CGI invia una “risposta” all'utente (pagina web, email, ecc.)

Form e CGI

- CGI: metodo più usato per elaborare form
- in teoria è possibile evitare l'uso di CGI e di qualunque programma lato server
 - invio diretto di email dal browser
- in pratica, ciò è possibile solo per form estremamente semplici
- si possono anche usare programmi lato server alternativi al CGI

L'elemento <form>

Apre e chiude il modulo e raccoglie il contenuto dello stesso

Esempio:

```
<FORM method="get|post"  
      action="http://www.dis.uniroma1.it/cgi-  
      bin/nome_script.cgi">
```

- **attributo ACTION:** specifica il CGI che elabora la form

L'attributo method

method=get

- i dati vengono spediti al server e separati in due variabili
- per questo metodo il numero massimo di caratteri contenuti nel form è di 255

method=post

- i dati vengono ricevuti direttamente dallo script CGI senza un preventivo processo di decodifica
- questa caratteristica fa sì che lo script possa leggere una quantità illimitata di caratteri

Campi editabili del modulo

Vengono definiti tramite gli elementi:

- INPUT (campi editabili, checkbox, radio, ecc.)
- TEXTAREA (area di testo)
- SELECT (creazione di menu di opzioni)

L'elemento INPUT

- Permette l'inserimento di campi editabili e/o modificabili nel modulo
- Attributi principali:
 - TYPE: determina il tipo di campo
 - NAME
 - VALUE
 - MAXLENGTH

Attributo TYPE di <INPUT>

- Attributo TYPE: determina il tipo di campo
- Possibili valori:
 - HIDDEN
 - TEXT
 - PASSWORD
 - CHECKBOX
 - RADIO
 - SUBMIT
 - RESET
 - IMAGE

TYPE=“HIDDEN”

- Usato per dare informazioni “nascoste” al CGI (cioè al server)
- Il suo uso dipende dal tipo di CGI associato al modulo

TYPE=“HIDDEN”

Esempi:

```
<INPUT type="HIDDEN" name=MAILFORM SUBJECT  
value="titolo del form">
```

- Questo codice determina il titolo (subject) del messaggio che verrà ricevuto via e-mail, e che conterrà il contenuto del modulo

```
<INPUT TYPE="HIDDEN" NAME=MAILFORM_URL  
VALUE="http://www.tuosito.it">
```

- dopo aver compilato e spedito correttamente il form, dà in risposta una pagina HTML successiva (es. pagina di conferma di invio modulo avvenuta)

TYPE=“TEXT”

```
<INPUT type="TEXT" name="nome"  
maxlength="40" size="33"  
value="inserisci nome">
```

- crea i tipici campi di testo
- usato soprattutto per informazioni non predefinite che variano di volta in volta
- attributi di <INPUT> nel caso TYPE=TEXT:
 - MAXLENGTH (num. max caratteri inseribili)
 - SIZE (larghezza campo all'interno della pagina)
 - VALUE (valore di default che compare nel campo)

TYPE=“PASSWORD”

```
<INPUT type="PASSWORD" name="nome"  
maxlength="40" size="33">
```

- crea i campi di tipo password
- vengono visualizzati asterischi al posto dei caratteri
- i dati NON vengono codificati (problema per la sicurezza)

TYPE=“CHECKBOX”

```
<INPUT type="checkbox" name="eta"  
size="3" VALUE="YES" CHECKED>
```

- crea campi booleani (si/no)
- crea delle piccole caselle quadrate da spuntare o da lasciare in bianco
- VALUE impostato su YES significa che di default la casella è spuntata
- CHECKED controlla lo stato iniziale della casella, all'atto del caricamento della pagina

TYPE=“RADIO”

```
<INPUT type="RADIO" name="giudizio"  
      value="sufficiente">  
  
<INPUT type="RADIO" name="giudizio"  
      value="buono">  
  
<INPUT type="RADIO" name="giudizio"  
      value="ottimo">
```

- usato per selezionare una tra alcune scelte
- tutte le scelte con lo stesso “name” (altrimenti una scelta non esclude le altre)

TYPE=“SUBMIT”

```
<INPUT type="SUBMIT" value="spedisci">
```

- crea un bottone che invia il modulo al server
- la lunghezza del bottone dipende dalla lunghezza del valore di VALUE

TYPE=“RESET”

```
<INPUT type="RESET" value="reimposta">
```

- crea un bottone che resetta i campi del modulo
- i dati inseriti vengono eliminati
- la lunghezza del bottone dipende dalla lunghezza del valore di VALUE

TYPE=“IMAGE”

```
<INPUT type="IMAGE" SRC="bottone.gif">
```

- come SUBMIT, ma crea un bottone tramite l'immagine (URL) specificata in SRC

L'elemento TEXTAREA

```
<TEXTAREA cols=40 rows=5 WRAP="physical"  
name="commento"> </textarea>
```

- utilizzato per commenti o campi che prevedono l'inserimento di molto testo
- WRAP="physical" stabilisce che qualora il testo inserito superi la larghezza della finestra, venga automaticamente riportato a capo

L'elemento SELECT

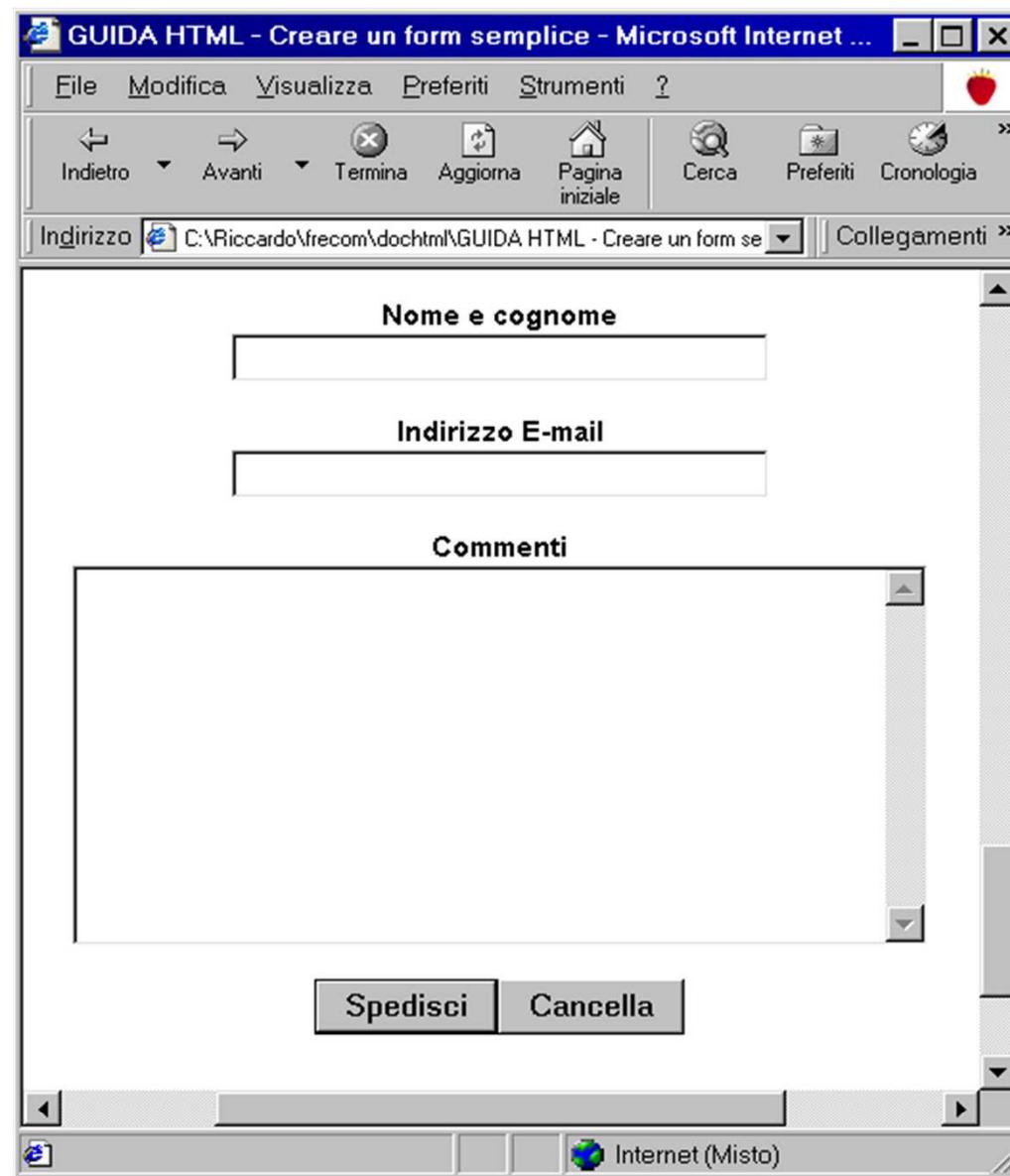
```
<SELECT size=1 cols=4 NAME="giudizio">
  <OPTION selected Value=nessuna>
  <OPTION value=buono> Buono
  <OPTION value=sufficiente> Sufficiente
  <OPTION Value=ottimo> Ottimo
</select>
```

- Permette la creazione di menu a tendina con scelte multiple

Esempio

```
<FORM ACTION=http://www.coder.com/code/mailform/mailform.pl.cgi METHOD=POST>
<INPUT TYPE=HIDDEN NAME=MAILFORM_ID VALUE="Val_7743">
<INPUT TYPE=HIDDEN NAME=MAILFORM SUBJECT VALUE="Il mio
primo FORM">
<INPUT TYPE=HIDDEN NAME=MAILFORM_URL
      VALUE="http://www.html.it/risposta.htm">
<B>Nome e cognome</B><BR>
<input type=text NAME=MAILFORM_NAME size=33><BR><BR>
<B>Indirizzo E-mail</B><BR>
<input type=text NAME=MAILFORM_FROM size=33><BR><BR>
<B>Commenti</B><BR>
<TEXTAREA NAME=MAILFORM_TEXT ROWS=10 COLS=42
          WRAP></TEXTAREA><BR><BR>
<INPUT TYPE=SUBMIT VALUE="Spedisci"><INPUT TYPE=RESET
      VALUE="Cancella">
</FORM>
```

Esempio (cont.)



Sommario

- forms
- fogli di stile (CSS)
- HTML “dinamico”
- cenni su HTML5

Fogli di stile

- In HTML non c'è separazione tra **forma e contenuto** del documento
- Fogli di stile o Cascading Style Sheets (CSS) = estensione di HTML introdotta da Internet Explorer 3
- Esempio: tag
- I CSS si occupano di gestire la formattazione del documento esternamente al documento stesso
- Standard W3C (CSS1 e CSS2)

Tipi di fogli di stile

- HTML permette di utilizzare diversi linguaggi per fogli di stile
- Linguaggio standard W3C per i fogli di stile: CSS (Cascading Style Sheets)
- text/css

Tipi di CSS

- CSS in linea
 - agiscono su singole istanze di testo all'interno della pagina
- CSS incorporati
 - agiscono in modo globale su un singolo documento
- CSS esterni
 - agiscono in modo globale su insiemi di documenti

CSS in linea

Agiscono su singole istanze di testo all'interno della pagina

Esempio:

```
<DIV style="font-size:18px;  
font-family:arial; color:red">  
    Questa parte di testo ha colore rosso  
</DIV>
```

Marcature usate:

- senza “semantica” (<DIV> o)
- marcature con “semantica” (es. <p>)

Limiti dei CSS in linea

- Semplici da usare (si possono considerare una “estensione” di HTML)
- Modificano il contenuto del testo
- Non rispondono all'esigenza di separazione tra forma e contenuto
- È opportuno usarli in modo molto circoscritto

CSS incorporati

- Agiscono in modo globale su un singolo documento HTML
- Vanno scritti nella parte intestazione del documento (tra <HEAD> e </HEAD>)
- Permettono di dare opzioni di formattazione **globale** ai tag (e quindi al documento)

CSS incorporati: esempio

```
<HTML>
<HEAD>
<style type="text/css">
H1 {font-size:17px; color:green}
H2 {font-family:arial; color:red}
</style>
</HEAD>
<BODY>
<H1>Tutti gli H1 sono di colore verde</H1>
<H2>Tutti gli H2 sono di colore rosso</H2>
</BODY>
</HTML>
```

CSS incorporati: sintassi

```
H1 { font-size:17px; color:green }  
H2 { font-family:arial; color:red }
```

- gli attributi sono inseriti tra parentesi graffe
- al posto del segno = vengono usati i due punti
- gli attributi composti da più termini sono separati da un trattino
- quando un attributo è considerato proprietà di un oggetto i trattini si eliminano e le iniziali dei termini diventano maiuscole
- esempio: font-style diventa FontStyle

Il tag <style>

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

- L'attributo TYPE del tag <STYLE> definisce il linguaggio in formato MIME del foglio di stile
- TYPE indica al browser il tipo di foglio di stile supportato
- Internet Explorer supporta i CSS solo in formato MIME

Il tag <style>

- Esistono anche i CSS in formato text/jass, cioè accessibili tramite JavaScript
- Se l'attributo TYPE viene omesso, il browser lo identifica di default con text/css

Attenzione: non confondere il **tag** <style> con
l'attributo style usato nei CSS in linea

CSS incorporati vs. CSS in linea

- I CSS incorporati permettono di configurare globalmente la formattazione del testo (cioè l'aspetto dei tag)
- I CSS incorporati sono definiti al di fuori del corpo del documento, e permettono così la separazione tra contenuto e forma
- Tuttavia, i CSS incorporati sono inadatti a trattare in modo uniforme insiemi di documenti (ad esempio un sito web)

CSS esterni

agiscono in modo **globale** su insiemi di documenti:

- gli stili dei singoli marcatori vengono raggruppati in un documento separato (file distinto dai documenti)
- ogni documento “richiama” gli stili (con un opportuno comando)
- una modifica sul file di stili genera automaticamente la stessa modifica su tutti i documenti che lo richiamano

CSS esterni

Esempio: file stile.css:

```
H1 {font-size:17px; color:green}  
H2 {font-family:arial; color:red}
```

contiene gli stili che si vogliono definire

Collegamento ad un CSS esterno

Collegamento ad un CSS esterno in un documento:

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

- Il tag `<link>` identifica un file esterno al documento HTML
- L'attributo `rel` indica il tipo di file collegato (stylesheet)
- L'attributo `href` richiama il percorso e il nome del file esterno

Vantaggi dei CSS esterni

- Permettono la separazione tra contenuto e forma
- Permettono di gestire insiemi di documenti
- Massima flessibilità di utilizzo
- Riusabilità degli stili
- Possibilità di **fondere** insieme più stili (cascading)

Attributi di stile per il testo

- font-family (serif / sans serif / cursive)
- font-size
 - punti (pt) / pixel (px) / centimetri (cm) / pollici (in) / percentuale (%)
- font-style
 - extra-light / demi-light / light / medium /bold/ demi-bold / extra-bold
- font-variant (normal / small-caps)
- font-weight
- text-decoration (none / underline / italic / line-height)

Esempio

Per i CSS in linea:

```
<A style="text-decoration:none"  
    href="#a1">ciao </a>
```

Per i CSS incorporati:

```
<style>  
A {text-decoration: none}  
</style>
```

Classi

- A tutti gli elementi è possibile assegnare l'attributo CLASS
- Questo attributo permette ai fogli di stile di trattare singole occorrenze o sottoinsiemi di occorrenze dello stesso elemento o di elementi diversi

Classi: esempio

```
<STYLE>
    H2.top {font-family:verdana; font-
            size:15px; font-weight:bold; font-
            style:normal}
    H2.bottom {font-family:arial; font-
               size:10px; font-weight:bold; font-
               style:italic}
</STYLE>

<H2 class=bottom> Testo della pagina</H2>
...
<H2 class=top> Testo della pagina</H2>
```

Pseudoclassi

- esempio: elementi anchor:
 - quando visitati costituiscono una pseudoclasse `visited`, quando attivi `active` e quando non visitati `link`
- viene specificata all'interno dello stile seguita dai due punti
- esempio:

```
<STYLE>
  A:link { text-decoration: none }
  visited { text-decoration: none }
</STYLE>
```

Sommario

- forms
- fogli di stile (CSS)
- HTML “dinamico”
- cenni su HTML5

Aspetti dinamici di HTML

- documento HTML “dinamico” = dipendente dal momento dell’esecuzione
- due tipi di dinamicità:
 - dinamicità lato client (HTML dinamico)
 - linguaggi di scripting lato client (es. Javascript)
 - dinamicità lato server (HTML generato dinamicamente)
 - linguaggi di scripting lato server (es. PHP, ASP)

HTML dinamico

HTML “dinamico”:

- documento HTML contenente script lato client
- il documento HTML è generato staticamente
- la sua esecuzione da parte del web client (browser) è dinamica:
 - lo script è eseguito al momento della richiesta del web client
 - la visualizzazione del documento HTML può variare da richiesta a richiesta

HTML generato dinamicamente

HTML generato dinamicamente:

- le pagine sono generate (sintetizzate) al momento della richiesta dell'utente
- generate da programmi in esecuzione sul web server
- i programmi possono essere
 - compilati
 - Java servlet, CGI,...
 - interpretati (linguaggi di scripting lato server)
 - PHP, JSP, ASP...

Script lato client

- Codice immerso nel documento HTML
- eseguiti dal web client (browser)
- principale linguaggio di scripting lato client: JavaScript
- Cosa fanno gli script lato client:
 - posizionamento dinamico degli oggetti
 - validazione campi dei moduli
 - effetti grafici
 -

Script lato server

- Codice immerso nel documento HTML
- eseguiti dal web server
- principali linguaggi di scripting lato server:
 - PHP, JSP, ASP
- Cosa fanno gli script lato server:
 - gestione di contenuti dinamici
 - es.: generazione in tempo reale del documento HTML in base al contenuto di sorgenti informative (basi di dati) collegate al web server
 - elaborazione informazione spedita dal client (form)

Script PHP: esempio

script PHP:

```
<html>
<head>
<title>Test PHP</title>
</head>
<body>
<?php echo "Hello World!<p>"; ?>
</body>
</html>
```

output HTML
generato:

```
<html>
<head>
<title>Test PHP</title>
</head>
<body>
Hello World!<p>
</body>
</html>
```

Gestione form in PHP: esempio

script PHP:

```
<form action="action.php" method="POST">
    Il tuo Nome:
    <input type="text" name="name" value="" />
    La tua et&grave;;
    <input type="text" name="age" value ="" />
    <input type="submit">
</form>
```

file
action.php:

```
<html><head></head>
<body>
Ciao <?php echo $_POST["name"]; ?>.<br>
La tua et&grave;; &egrave; di
<?php echo $_POST["age"]; ?> anni.
</body></html>
```

Gestione form in PHP: esempio

form:

A screenshot of a Mozilla Firefox browser window. The title bar says "Mozilla Firefox". The menu bar includes "File", "Modifica", "Visualizza", "Cronologia", "Segnalibri", "Strumenti", and a question mark icon. The toolbar includes standard icons for back, forward, search, and refresh. The address bar shows "file:///C:/Documents and S...". The main content area contains a form with two input fields: "Il tuo Nome:" and "La tua età:". Below the fields is a blue "Invia richiesta" button. At the bottom of the page, a status bar displays "Completato".

compilazione
del form:

A screenshot of a Mozilla Firefox browser window, identical to the one above but with data entered into the form fields. The "Il tuo Nome:" field now contains "mario" and the "La tua età:" field contains "25". The "Invia richiesta" button remains blue. The status bar at the bottom still says "Completato".

Gestione form in PHP: esempio

output HTML
generato:

```
<html><head></head>
<body>
Ciao Mario.<br>
La tua età è di 25 anni.
</body></html>
```

Riferimenti

- Raccomandazioni ufficiali W3C:
 - www.w3.org
- Guida di riferimento HTML:
 - es.: Campbell-Darnell, *HTML dinamico - guida completa*, APOGEO editore
- Guida “pratica” per HTML:
 - es.: Tittel, *HTML for dummies*, APOGEO editore
- Sito italiano per gli sviluppatori HTML:
 - www.html.it

Riferimenti

- Tutorial su PHP:
 - <http://it2.php.net/tut.php>

Sommario

- forms
- fogli di stile (CSS)
- HTML “dinamico”
- cenni su HTML5

HTML5: breve storia

- Proposta (nata nel 2007) dal working group WHATWG (formato da Apple, Mozilla, Opera)
- HTML5 si poneva come alternativa all'XHTML 2 allora proposto dal W3C
- In reazione a questo, il W3C ha accettato di abbandonare la standardizzazione di XHTML 2 e di adottare HTML5
- WHATWG ha collaborato col W3C nella definizione di HTML5 fino al 2012 (ora propone un «living standard» per HTML)
- Attualmente (novembre 2013) HTML5 è una «W3C candidate recommendation», dovrebbe diventare uno standard ufficiale nel 2014

HTML5: principali caratteristiche

- Supporto nativo di risorse audio e video:
 - Elementi `<audio>`, `<video>`
 - `<video id="sampleMovie" width="640" height="360" preload controls></video>`
- Geolocalizzazione
 - Introduzione di API per la geolocalizzazione
 - `getCurrentPosition`, `watchPosition`, ...
- Canvas:
 - Introduzione nel documento di una regione disegnabile dinamicamente
 - `<canvas id="esempio" width="196" height="96"></canvas>`

HTML5: principali caratteristiche

- Nuove API (DOM per HTML e XML)

Canvas

- Il Canvas consiste in una regione disegnabile, definita in codice HTML con gli attributi *height* and *width*
- Il codice JavaScript può accedere all'area con un set completo di funzioni per il disegno, permettendo così la generazione dinamica di disegni
- Canvas permette la generazione di grafici, l'animazione e la composizione di immagini