

# Introduzione all'Informatica

**Riccardo Ortale**

**Pagine HTML**

Lezione 4

# Introduzione ad HTML (1)

- ◆ Il linguaggio HTML è un tipo di documenti SGML (*esiste un DTD di HTML*).
- ◆ HTML viene usato dai browser WWW per visualizzare documenti ipertestuali. Tramite HTML è possibile realizzare documenti con una semplice struttura, con aspetti grafici anche sofisticati, con testo, immagini, oggetti interattivi e connessioni ipertestuali ad altri documenti
- ◆ Fino ad oggi i browser si sono preoccupati poco della correttezza sintattica o strutturale dei documenti HTML. Questo significa che tra un documento HTML *visualizzabile da un browser WWW* ed un documento HTML *corretto* esistono differenze anche sensibili.

## Introduzione ad HTML (2)

- ◆ È normale associare un significato strutturale agli elementi definiti in un DTD. HTML associa anche significati grafici agli elementi che definisce. Cioè dà istruzioni più o meno precise su come rendere graficamente gli elementi che definisce.
- ◆ Questo porta anche ad abusi della resa grafica che a noi interessano poco. Per noi la resa grafica finale, in assenza di linguaggi di stile appropriato, è secondaria.
- ◆ HTML non forza strutture gerarchiche nei suoi documenti. Inoltre in HTML i vincoli di contenimento tra elementi sono pochi e piuttosto ovvi. I browser WWW sono ancora più lassisti, a questo proposito, del linguaggio stesso.

# Storia di HTML (1)

- ◆ HTML è esistito in varie versioni dal 1989 ad oggi:
  - 0.9 (?): il linguaggio di HTML disponibile sul browser WWW aveva strutture di base per intestazioni, paragrafi e stili base, più ovviamente il tag A che ne costituiva la novità.
  - 1.0 (1992): I primi browser shareware e freeware al mondo (il più importante di questo periodo fu Arena, ma esistevano anche MacWeb ed altri) implementavano alcune versioni di HTML leggermente diverse. Tra queste ebbe un certo successo la proposta HTML+. Viene introdotto il tag IMG e il supporto per il GIF.
  - 2.0 (1994): La prima versione veramente nota di HTML. Questa è quella implementata su Mosaic, da cui deriverà Netscape. E' la prima versione ad essere formalizzata su un DTD SGML, invece che ispirarsi vagamente ad SGML. Introduce i form.

## Storia di HTML (2)

- 3.0 (1995): Questa versione non è mai stata ufficialmente approvata. Durante la sua discussione vennero proposte molte aggiunte. Alcune di queste vennero implementate prima di raggiungere un consenso (tabelle), altre (ad es. supporto per la matematica) mai prese in considerazione.
- 3.2 (1997): Quando divenne chiaro che i browser non avrebbero supportato tutto il 3.0, si lavorò per generarne un sottoinsieme su cui ci fosse consenso, e che tenesse conto delle aggiunte proprietarie dei vari produttori. Questa versione include tabelle, applet, script e altre migliorie, ma non i frame, sebbene Netscape e Microsoft le avessero già implementate fin dal 1995.
- 4.0 (1997): supporto per l'internazionalizzazione, per gli style sheet, per i frame, tabelle molto più ricche, il tag OBJECT, ecc.
- 4.01 (dic. 1999): contiene alcune minime variazioni e correzioni.

## Storia di HTML (3)

- XHTML1.0: Nel 1998 parte l'iniziativa di riformulare HTML come applicazione di XML, piuttosto che di SGML.  
Il 26 gennaio 2000 esce la prima recommendation del W3C, XHTML 1.0, che è una semplice riformulazione di HTML 4 in termini di XML, senza nessuna introduzione di nuove forme. XHTML però identifica anche un percorso di evoluzione verso la creazione di una famiglia di tipi di documenti che estendano localmente o semplifichino XHTML per una vasta gamma di usi e device.

- ◆ Una prima recommendation è del 18 dicembre 1997, una revisione del 24 aprile 1998. Il 24 dicembre 1999 esce HTML 4.01, che contiene alcuni cambiamenti editoriali e precisazioni.
- ◆ La diffusione dei fogli di stile ha liberato i web designer dall'obbligo di forzare i tag HTML ad assumere uno scopo tipografico e decorativo.
- ◆ Però alcuni tag esistono solo allo scopo di fornire istruzioni tipografiche di base. Per compatibilità col passato, tuttavia, si è deciso di mantenere il supporto per alcuni elementi che non hanno più senso con l'uso dei fogli di stile.

## I DTD di HTML 4

◆ Per esigenze di compatibilità con il passato, HTML 4 è composto di tre strutture (DTD) alternative:

- Transitional DTD (detto anche loose): contiene l'intero linguaggio ammesso per HTML, inclusi quegli elementi "deprecati" che vengono mantenuti per compatibilità col passato.
- Strict DTD: contiene i soli elementi di HTML che non vengono influenzati dall'uso degli style sheet (ma sono escluse le tabelle, che non sono gestite da CSS).
- Frameset DTD: un semplicissimo DTD per quei documenti in cui al posto di BODY si usano i tag dei frame.

## Criteri di sviluppo (1)

- ◆ HTML 4.0 estende HTML 3.2 con meccanismi per i fogli di stile, gli script, i frame, oggetti embedded, criteri di internazionalizzazione, tabelle più ricche e miglioramenti ai form.
- ◆ Questi sono i criteri di sviluppo più importanti:
  - **Internazionalizzazione** (*Internationalization* o *I18N*): l'adozione dei meccanismi necessari per il supporto di linguaggi e notazioni di tutto il mondo, e per la creazione di documenti contenenti linguaggi misti.
  - **Accessibilità**: l'adozione dei meccanismi necessari per il supporto delle esigenze degli utenti con limitazioni fisiche (visive, uditive, etc.).

## Criteri di sviluppo (2)

- **Tabelle sofisticate:** l'adozione di meccanismi necessari per creare tabelle ancora più sofisticate delle precedenti.
- **Documenti composti:** l'adozione dei meccanismi necessari per inserire (embed) in maniera generalizzata oggetti di ogni possibile media all'interno di una pagina HTML.
- **Style sheet:** l'adozione di meccanismi per specificare in maniera precisa e sofisticata la resa tipografica di una pagina senza appesantire la gestione del contenuto.
- **Scripting:** l'adozione dei meccanismi necessari per realizzare sul client degli oggetti attivi, in grado di eseguire computazioni locali (ad esempio, per pre-verificare la correttezza delle informazioni inserite in un form).

## Usare HTML 4 (1)

### ◆ Separazione di struttura e presentazione

- Via via che HTML tende ad assomigliare al suo antenato SGML, molti dei suoi aspetti presentazionali vengono sostituiti o affiancati da altri meccanismi. Questo porta in particolare a "deprecare" gli aspetti più presentazionali di HTML (ad esempio, l'elemento FONT), e a proporre meccanismi alternativi, più indipendenti e sofisticati (ad esempio, gli style sheet).

### ◆ Considerare l'accessibilità universale al Web

- Via via che si considera il supporto di un numero maggiore di utenti, maggiori saranno le differenze tra di essi di cui tenere conto: esigenze linguistiche specifiche, minorazioni fisiche, architetture diverse, modalità diverse di fruizione delle pagina richiedono gli autori di considerare appropriatamente le specifiche esigenze di tutti.

## Usare HTML 4 (2)

### ◆ Aiutare i browser con il rendering incrementale

- Immagini, oggetti embedded, tabelle complesse rendono complicato (e quindi lento) il meccanismo di impaginazione dei documenti HTML. L'adozione di misure per permettere la visualizzazione incrementale delle caratteristiche del documento favorisce una velocità percepita di visualizzazione utile per il buon successo delle proprie pagine.

### ◆ Internazionalizzazione (I18N)

- Il supporto per l'internazionalizzazione passa attraverso 4 meccanismi:
  - Il set di caratteri (UCS -8, 16, 32).
  - L'attributo lang
  - L'attributo dir
  - Il tag BDO (di-directional override)

# Indice degli argomenti <sup>(1)</sup>

- ◆ La struttura generale
- ◆ I comandi fondamentali
- ◆ Formattazione del testo
- ◆ Creazione di liste
- ◆ Collegamenti

## Indice degli argomenti (2)

- ◆ Immagini
- ◆ Tabelle
- ◆ Moduli
- ◆ Fogli di stile

# Indice degli argomenti <sup>(1)</sup>

- ◆ La struttura generale
- ◆ I comandi fondamentali
- ◆ Formattazione del testo
- ◆ Creazione di liste
- ◆ Collegamenti

## La struttura generale <sup>(1)</sup>

- ◆ La struttura viene definita da un DTD, un documento che viene richiamato per consentire ai Browser Web la corretta interpretazione delle pagine HTML

Es: `<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">`

- ◆ Viene inserito all'inizio della pagina
- ◆ Anche se non compare nel documento HTML, il Browser lo da per scontato

## La struttura generale (2)

- ◆ La pagina web è composta da Tag, marcatori che identificano porzioni di testo e ne definiscono il significato
- ◆ Ogni tipo di tag è composto da due marcatori uno di “apertura” e uno di “chiusura”
- ◆ La sintassi è: `<nome tag>` tag di apertura  
`</nome tag>` tag di chiusura

## La struttura generale <sup>(3)</sup>

- ◆ Dopo il DOCTYPE compare il tag HTML

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
```

```
<HTML>
```

```
Documento html
```

```
</HTML>
```

- ◆ All'interno dei due tag vengono inseriti gli altri marcatori e il resto del documento
- ◆ Non vanno inseriti altri marcatori, eccetto il DOCTYPE prima di ogni altro tag, al di fuori di HTML

## La struttura generale (4)

- ◆ I tag possono essere composti, oltre che dal nome, anche da attributi che ne determinano il “funzionamento”, in varie circostanze. Es:

```
<Font color="#FFFFFF">
```

- ◆ Fra i tag HTML troviamo i tag HEAD e BODY
- ◆ In alternativa a BODY c'è FRAMESET

## La struttura generale <sup>(5)</sup>

- ◆ Il tag HEAD viene usato per inviare i dati di intestazione della pagina
- ◆ Il tag BODY viene usato per inviare il corpo della pagina, ovvero il documento vero e proprio (testo, immagini, filmati, suoni, musica, ecc...)
- ◆ Il tag FRAMESET viene usato per gestire le aree della pagina

# La struttura generale (6)

## ◆ La pagina sarà così composta:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">  
<HTML>
```

```
<HEAD> Corpo dell'intestazione  
</HEAD>
```

```
<BODY> Corpo della pagina  
</BODY>
```

```
</HTML>
```

## La struttura generale (7)

- ◆ I tag di intestazione possono racchiudere i seguenti tag:
  - BASE
  - FRAMESET
  - ISINDEX
  - META
  - LINK
  - SCRIPT
  - STYLE
  - TITLE

## La struttura generale <sup>(8)</sup>

- ◆ L'elemento BASE ha soltanto l'attributo HREF ed è utilizzato per impostare l'URL di base del documento
- ◆ Gli URL relativi utilizzati all'interno del documento sono risolti utilizzando l'URL fornito in BASE
- ◆ Questo tag non è obbligatorio
- ◆ Esempio:  
<BASE HREF="http://www.pippo.com/pluto/paperino">

## La struttura generale <sup>(9)</sup>

- ◆ L'elemento FRAMESET è usato per dividere la finestra del browser in due o più finestre di documento
- ◆ In ogni finestra è possibile mostrare un file differente
- ◆ Ci si riferisce a queste differenti finestre come a FRAME

## La struttura generale <sup>(10)</sup>

- ◆ L'elemento ISINDEX aveva lo scopo di rappresentare un documento su cui poter effettuare ricerche
- ◆ Questo tag non ha avuto successo, perchè richiede un motore di ricerca separato
- ◆ È stato sostituito dall'utilizzo dei moduli e dai CGI (Common Gateway Interface)

## La struttura generale (11)

- ◆ L'elemento LINK specifica il riferimento a un determinato documento e possiede quattro attributi:
  - HREF
  - REL
  - REV
  - TITLE
- ◆ Esempio  
<LINK REV="style sheet" HREF="style.css" TYPE="text/css">

## La struttura generale (12)

- ◆ L'elemento META è utilizzato per passare informazioni aggiuntive su come dovrebbe essere gestito il documento
- ◆ È utilizzato per fornire informazioni supplementari sul documento che possono essere utilizzate dai motori di ricerca
- ◆ I suoi attributi sono: HTTP-EQUIV, NAME, CONTENT E SCHEME

## La struttura generale <sup>(13)</sup>

- ◆ L'elemento SCRIPT è stato incorporato in HTML per poter inserire piccole applicazioni (lato client), scritte con linguaggi di scripting
- ◆ Richiede un tag di apertura e uno di chiusura
- ◆ I suoi attributi sono: TYPE, LANGUAGE e SRC
- ◆ Per vecchi browser può essere necessario NOSCRIPT

## La struttura generale <sup>(14)</sup>

- ◆ L'elemento STYLE consente di richiamare un foglio di stile interno (contenuto nella stessa pagina html e racchiuso entro i tag)
- ◆ Ha un tag di apertura e uno di chiusura e possiede l'attributo TYPE
- ◆ LINK importa le informazioni di stile da un foglio di stile separato (contenuto in un file separato, con estensione “.css”); mentre STYLE consente la definizione degli stili nella pagina stessa

## La struttura generale (15)

- ◆ L'elemento TITLE consente di dare un titolo alla pagina web, che viene visualizzato dal browser sulla barra del titolo della finestra
- ◆ Non necessita di tag di chiusura
- ◆ Non ha attributi
- ◆ Dev'essere usato una sola volta all'interno del tag HEAD

# Indice degli argomenti <sup>(1)</sup>

- ◆ La struttura generale
- ◆ I comandi fondamentali
- ◆ Formattazione del testo
- ◆ Creazione di liste
- ◆ Collegamenti

## I comandi fondamentali (1)

Dopo aver visto la struttura di una pagina HTML e i possibili tag contenuti nell'intestazione, ci concentreremo sul tag BODY, i suoi attributi e sui principali insiemi di tag racchiusi nel corpo della pagina web

## I comandi fondamentali (2)

- ◆ BODY possiede tag di apertura e di chiusura.
- ◆ Contiene il testo e altre informazioni (anche multimediali) che verranno visualizzate al lettore
- ◆ Gli elementi contenuti all'interno di questo tag sono suddivisi in elementi di formattazione di blocco e di carattere

## I comandi fondamentali (3)

- ◆ Gli attributi di BODY sono:
  - BACKGROUND=definizione dell'immagine di sfondo
  - BGCOLOR=definizione del colore di sfondo
  - TEXT= imposta il colore del testo del documento
  - LINK=imposta il colore del collegamento ipertestuale

## I comandi fondamentali (4)

- ◆ Gli attributi di BODY sono:
  - ALINK=imposta il colore del collegamento ipertestuale attivo
  - VLINK=imposta il colore del collegamento ipertestuale visitato

## I comandi fondamentali (5)

Gli insiemi di comandi racchiusi all'interno del tag BODY, sono:

- Formattazione del testo
- Creazione di liste
- Collegamenti ipertestuali
- Immagini
- Tabelle
- Moduli
- Finestre e Frames
- Fogli di stile

## I comandi fondamentali <sup>(6)</sup>

Un comando utile per inserire testo non visualizzabile, è quello relativo ai commenti

- ◆ Sintassi : `<!-- commento -->`
- ◆ I commenti possono comprendere più di una riga di testo

# Indice degli argomenti <sup>(1)</sup>

- ◆ La struttura generale
- ◆ I comandi fondamentali
- ◆ Formattazione del testo
- ◆ Creazione di liste
- ◆ Collegamenti

# Formattazione del testo <sup>(1)</sup>

<H1> ... <H6>

- ◆ Rappresenta i 6 stili di titolo html

- ◆ Sintassi :

<Hn [align="left|center|right"]> testo </Hn>

Esempio:

<H1>text</H1>

Titolo con dimensione massima dei caratteri

<H6>text</H6>  
caratteri

Titolo con dimensione minima dei

## Formattazione del testo (2)

<P>

- ◆ Rappresenta un paragrafo
- ◆ Tra i paragrafi lascia una riga vuota
- ◆ Sintassi:

<P [align="left|center|right|justify"]>

testo

</P>

Esempio:

**<p>text</p>** Aggiunge un'interruzione di paragrafo dopo il testo (2 linebreaks).

**<p align="left">text</p>** Giustifica a sinistra il testo nel paragrafo.

**<p align="center">text</p>** Testo centrale nel paragrafo.

**<p align="right">text</p>** Giustifica a destra il testo nel paragrafo.

## Formattazione del testo <sup>(3)</sup>

<BR>

- ◆ È il tag di interruzione di riga, è simile a quello di paragrafo ma, a differenza di quest'ultimo, inizia una nuova riga nel paragrafo corrente.
- ◆ Sintassi:

<BR [clear="left|right|all"]> testo ...

NB: Clear definisce come il materiale seguente debba fluire intorno alle immagini. Il tag di chiusura non è utilizzato

Esempio:

**text<br>** Aggiunge una singola interruzione di riga dove c'è il tag.

## Formattazione del testo (4)

<NOBR>

- ◆ Delimita un blocco di testo che non va mai separato (poco usato)

Esempio:

**<nobr>text</nobr>**

<WBR>

- ◆ Indica un punto preferenziale di rottura all'interno di una sequenza <NOBR>

Esempio:

**text<wbr>**

**Permette al browser di inserire un "a capo" esattamente in questo punto, anche se il testo è compreso tra tags <nobr>.**

## Formattazione del testo <sup>(5)</sup>

<PRE>

- ◆ Rappresenta un blocco di testo preformattato
- ◆ Sintassi:

<PRE [width="caratteri"]> testo </PRE>

NB: width indica il numero di caratteri da lasciare nel testo

## Formattazione del testo (6)

<DIV>

- ◆ È utilizzato per strutturare un documento html in una serie di suddivisioni
- ◆ Agisce in modo molto simile al tag <P>
- ◆ <DIV> non genera interruzioni di paragrafo prima o dopo il suo posizionamento
- ◆ Sintassi:

```
<DIV [align="left|center|right|justify"]> testo  
</DIV>
```

Esempio:

**<div align="center">text</div>** Testo centrale.

**<div align="left">text</div>** Testo giustificato a sinistra.

**<div align="right">text</div>** Testo giustificato a destra.

## Formattazione del testo (7)

<CENTER>

◆ È identico al tag <DIV align="center">

◆ Sintassi:

<CENTER> testo </CENTER>

## Formattazione del testo (8)

### <EM> & <STRONG>

- ◆ <EM> viene usato per enfatizzare il testo (rappresentarlo in corsivo)
- ◆ <STRONG> viene utilizzato per indicare un livello d'importanza più elevato del testo (rappresenta il testo in grassetto)
- ◆ Sintassi:
- ◆ <EM>testo</EM>   <STRONG>testo</STRONG>

## Formattazione del testo <sup>(9)</sup>

### <ADDRESS>

- ◆ Utilizzato per contrassegnare indirizzi nel documento
- ◆ Si comporta in modo simile al tag di paragrafo

NB: Il testo compare separato, tramite una riga, dagli altri paragrafi

- ◆ Sintassi:

```
<ADDRESS>info</ADDRESS>
```

## Formattazione del testo (10)

<BLOCKQUOTE> & <CITE> o <Q>

- ◆ Sono utilizzati per definire citazioni e fonti di annotazioni
- ◆ Sintassi:

<BLOCKQUOTE> testo </BLOCKQUOTE>

<CITE> testo </CITE>

## Formattazione del testo (11)

<DFN>

- ◆ Utilizzato per presentare termini e definizioni
- ◆ Sintassi:

<DFN> testo </DFN>

## Formattazione del testo (12)

<CODE> & <VAR>

- ◆ <CODE> Viene utilizzato per indicare che la porzione di testo racchiusa fra i tag rappresenta il codice di un programma
- ◆ <VAR> Viene utilizzato per indicare che la porzione di testo considerato rappresenta una variabile
- ◆ Sintassi:

<CODE>testo</CODE> <VAR>testo</VAR>

## Formattazione del testo (13)

<SAMP> & <KBD>

◆ Serviva per simulare l'interazione tra l'utente e la macchina

◆ Sintassi:

<SAMP> testo </SAMP>

<KBD> testo </KBD>

## Formattazione del testo (14)

<U> & <B> & <I>

◆ Rappresentano i tre stili di formattazione del testo fondamentali: **I** sta per corsivo, **B** per grassetto, **U** per sottolineato

◆ Sintassi:

<U> testo </U>    <B> testo </B>

<I> testo </I>

NB: la sottolineatura è da usare con parsimonia perchè può confondere, è utilizzata per contrassegnare i collegamenti ipertestuali

## Formattazione del testo (15)

<STRIKE> o <S>

- ◆ Serve per visualizzare il testo barrato (cancellato da una riga centrale rispetto al testo)
- ◆ Viene utilizzato per mostrare correzioni nel documento
- ◆ Sintassi:

<STRIKE> testo </STRIKE>

## Formattazione del testo (16)

<TT>

- ◆ Rappresenta il testo di una telescrivente, stampante o testo a spaziatura fissa (ciascun carattere occupa la stessa larghezza)
- ◆ Sintassi:

<TT> testo </TT>

## Formattazione del testo (17)

<SUP> & <SUB>

- ◆ Utilizzati per la rappresentazione, rispettivamente, di apici e pedici
- ◆ Sintassi:

<SUP> testo </SUP>

<SUB> testo </SUB>

# Formattazione del testo (18)

<FONT>

- ◆ Consente di modificare gli attributi dei caratteri
- ◆ Sintassi:

```
<FONT face="arial" size="3" color="nome|#0000FF"> testo  
</FONT>
```

Esempio:

```
<html>  
  <head>  
    <title>My Page</title>  
  </head>  
  
  <body>  
    <basefont color="#FFFFFF" face="arial" size="4">  
    Ciao! Questa è la mia pagina .<br><br>  
    <font color=#00FF00 face=arial size=2>  
    Questa parte di testo è diversa.  
    </font>  
    <br><br>  
    Questo testo è come la prima riga.  
  </body>  
</html>
```

## Formattazione del testo (19)

### <BASEFONT>

- ◆ Viene utilizzato per impostare la dimensione del testo
- ◆ Non ha tag di chiusura
- ◆ Sintassi:

<BASEFONT [size=numero]>

Esempio:

```
<html>
```

```
  <head>
```

```
    <title>my page</title>
```

```
  </head>
```

```
  <body>
```

```
    <basefont face="arial, verdana, courier" size="4" color="#FFFFFF">
```

```
    Ciao! Questa è la mia pagina.<br>
```

```
    Il testo è tutto uguale<br>
```

```
    perché ho specificato soltanto un basefont.<br>
```

```
  </body>
```

```
</html>
```

## Formattazione del testo (20)

### <BIG> & <SMALL>

- ◆ Consentono di rappresentare il testo con caratteri più grandi o più piccoli rispetto alla dimensione del carattere di base
- ◆ Sintassi:

<BIG> testo </BIG>

<SMALL> testo </SMALL>

## Formattazione del testo (21)

<HR>

- ◆ Visualizza una linea orizzontale nella pagina
- ◆ Non ha tag di chiusura
- ◆ Sintassi:

```
<HR [height="altezza"] [width="larghezza"] [align="left|center|right"]  
    [noshade]>
```

NB: noshade toglie l'ombreggiatura

# Indice degli argomenti <sup>(1)</sup>

- ◆ La struttura generale
- ◆ I comandi fondamentali
- ◆ Formattazione del testo
- ◆ Creazione di liste
- ◆ Collegamenti

## Creazione di liste <sup>(1)</sup>

### <OL> & <LI>

- ◆ Per creare elenchi ordinati bisogna inserire la lista di elementi all'interno dei tag di apertura e di chiusura di <OL>
- ◆ Ogni elemento di tale lista dev'essere contrassegnato dai tag di apertura e chiusura di <LI>
- ◆ OL significa "Ordered List", LI "List Item"

## Creazione di liste (2)

<OL> & <LI>

◆ Sintassi:

<OL>

<LI> elemento1 </LI>

<LI> elemento2 </LI>

...

<LI> elemento X

...

<LI> elemento N

</OL>

## Creazione di liste <sup>(3)</sup>

<OL> & <LI>

- ◆ Per ciascuna voce identificata con <LI>, il browser inizia una nuova riga, pone i rientri e aggiunge un numero
- ◆ Il tag di chiusura </LI> per ciascuna voce non è necessario
- ◆ Sintassi:

<OL [start="num di reinizio"] [type="1|A|a|I|i"]>

## Esempi di liste ordinate (1)

◆ **<ol>**

**<li>text</li>**

**<li>text</li>**

**<li>text</li>**

**</ol>**

**Crea una lista a numeri usando il valore di default:**

**1.text**

**2.text**

**3.text**

◆ **<ol start="5">**

**Inizia una lista a numeri, partendo dal n. 5.**

**5.Questa è una riga**

**6.Questa è un'altra riga**

**7.E questa è l'ultima riga**

## Esempi di liste ordinate (2)

◆ **<ol type="A">**

Inizia una lista a numeri, usando lettere maiuscole.

A. Questa è una riga

B. Questa è un'altra riga

C. E questa è l'ultima riga

◆ **<ol type="a">**

Inizia una lista a numeri, usando lettere minuscole.

a. Questa è una riga

b. Questa è un'altra riga

c. E questa è l'ultima riga

## Esempi di liste ordinate (3)

### ◆ `<ol type="I">`

Inizia una lista a numeri, usando numeri romani maiuscoli.

- I. Questa è una riga
- II. Questa è un'altra riga
- III. E questa è l'ultima riga

### ◆ `<ol type="i">`

Inizia una lista a numeri, usando numeri romani minuscoli.

- i. Questa è una riga
- ii. Questa è un'altra riga
- iii. E questa è l'ultima riga

## Esempi di liste ordinate (4)

◆ **<ol type="1">**

Inizia una lista a numeri, usando numeri normali.

1. Questa è una riga
2. Questa è un'altra riga
3. E questa è l'ultima riga

◆ **<ol type="I" start="7">**

Un esempio di come type (tipo) e start (inizio) possono essere combinati.

- VII. Questa è una riga
- VIII. Questa è un'altra riga
- IX. E questa è l'ultima riga

## Creazione di liste (4)

<UL> & <LI>

◆ Gli elenchi non ordinati sono utilizzati per rappresentare un insieme di voci, che non richiedono di seguire un ordine specifico

◆ Sintassi:

<UL [type="circle|square|disc"]>

◆ UL sta per “Unordered List”

NB: anche il tag <LI> può avere due attributi opzionali “type” e “start”

## Creazione di liste (5)

<UL> & <LI>

◆ Sintassi:

<UL>

<LI> elemento1

<LI> elemento2

...

<LI> elemento N

</UL>

## Creazione di liste (6)

### <DL> & <DT> & <DD>

- ◆ Il tag <DL> viene utilizzato per creare elenchi di definizione e glossari
- ◆ DL sta per Definition List
- ◆ È utilizzato in modo analogo al tag di elenco non ordinato, tranne per il fatto che non utilizza il tag <LI>

## Esempi di liste non ordinate (1)

◆ `<ul>`  
`<li>text</li>`  
`<li>text</li>`  
`<li>text</li>`  
`</ul>`

Crea una lista a punti usando il tipo di punto del default:

- text
- text
- text

◆ `<ul type="disc">`

Inizia una lista a punti usando dischi come punti:

- Questa è una riga
- Questa è un'altra riga
- E questa è l'ultima riga

## Esempi di liste non ordinate (2)

### ◆ `<ul type="circle">`

Inizia una lista a punti usando cerchi come punti:

- Questa è una riga
- Questa è un'altra riga
- E questa è l'ultima riga

### ◆ `<ul type="square">`

Inizia una lista a punti usando quadrati come punti:

- Questa è una riga
- Questa è un'altra riga
- E questa è l'ultima riga

## Creazione di liste <sup>(7)</sup>

<DL> & <DT> & <DD>

◆ Un elenco di definizione richiede due voci per ogni lemma:  
un termine e la sua definizione

◆ Sintassi:

<DL>

<DT> termine 1 <DD> definizione 1

<DT> termine 2 <DD> definizione 2 .....

<DT> termine N <DD> definizione N </DL>

## Creazione di liste (8)

<DL> & <DT> & <DD>

- ◆ DL sta per “Definition List”
- ◆ DT sta per “Definition Term”
- ◆ DD sta per “Definition Defined”

## Creazione di liste <sup>(9)</sup>

<MENU> o <DIR> & <LI>

- ◆ I tag <MENU> e <DIR> vengono utilizzati per creare stili di elenco logici
- ◆ Non sono usati ampiamente come i tag <OL> e <UL>
- ◆ Sintassi:

<MENU> <!-- o DIR -->

<LI> elemento 1 <LI> elemento 2 ... <LI> elemento N

</MENU> <!-- o /DIR -->

# Indice degli argomenti <sup>(1)</sup>

- ◆ La struttura generale
- ◆ I comandi fondamentali
- ◆ Formattazione del testo
- ◆ Creazione di liste
- ◆ Collegamenti

# Collegamenti <sup>(1)</sup>

<A>

- ◆ Per creare collegamenti ipertestuali si utilizza questo tag che è l'acronimo di "Anchor", ovvero ancora
- ◆ Ha un tag di apertura e uno di chiusura
- ◆ Sintassi:

<A> [contenuto HTML (txt/img)] </A>

Esempio:

Clicca **<a href="http://www.yahoo.com">qui</a>** per andare su yahoo.

## Collegamenti (2)

### ◆ Sintassi degli attributi:

```
<A [[href="URL|[URL]#nome ancora"]][name="nome ancora"]  
  [target="nome frame"] [rel="tipo"] [rev="tipo"] [title="titolo  
  documento"]> contenuto HTML </A>
```

Href sta per Hypertext REFerence, ovvero riferimento d'ipertesto. Può contenere o l'indirizzo di una pagina esterna o un indirizzo interno alla pagina stessa

Name serve per definire la destinazione interna alla pagina

Target definisce in quale frame visualizzare la pagina richiamata

## Collegamenti (3)

### ◆ Sintassi degli attributi:

```
<A [[href="URL|[URL]#nome ancora"]][name="nome ancora"]  
  [target="nome frame"] [rel="tipo"] [rev="tipo"] [title="titolo  
  documento"]> contenuto HTML </A>
```

Rel e Rev sono utilizzati solo in abbinamento con "href", sono usati per definire il tipo di collegamento ipertestuale

Title è usato solo in abbinamento con "href" per fornire informazioni sul collegamento e dovrebbe essere uguale al testo contenuto nel tag <TITLE>

## Definizione del colore per i link testuali

- ◆ Definizione del colore per tutti i link su una pagina.
- ◆ Il colore generale dei link di testo è specificato nel tag **<body>**, come nell'esempio qui sotto:

```
<body link="#C0C0C0" vlink="#808080" alink="#FF0000">
```

- **link** - link normale - per una pagina dove il visitatore non è ancora stato. (il colore standard è blu - #0000FF).
- **vlink** - link visitato - per una pagina dove il visitatore è già stato. (il colore standard è viola - #800080).
- **alink** - link attivo - il colore del link quando è toccato dal mouse. (il colore standard è rosso - #FF0000).

## Link senza sottolineatura

- ◆ Per default, i link di testo vengono sottolineati dal browser.
- ◆ Si può disattivare la sottolineatura per un'intera pagina aggiungendo un tag "style" nella sezione "head" del documento.
- ◆ Esempio:

```
<html>
<head>
<title>Questa è la mia pagina</title>
<style type="text/css">
<!--
A{text-decoration:none}
-->
</style>
</head>

<body>
<a href="http://www.yahoo.com">Questo link per yahoo
non è sottolineato</a>
</body>

</html>
```

## Immagini come links

- ◆ Un'immagine può funzionare come un link. Il metodo è lo stesso che si usa per i testi.
- ◆ Si deve soltanto sistemare la coppia di tag `<a href>` e `</a>` su ogni lato dell'immagine.
- ◆ Esempio:

```
<a href="myfile.htm"></a>
```

- ◆ Le immagini che si comportano come link possono mostrare un testo che appare quando il mouse scorre su di esse. Questo si ottiene con la proprietà **alt** nel tag `<img>`.
- ◆ Esempio:
- ◆ `<a href="myfile.htm"></a>`

## Mappatura delle immagini (1)

- ◆ E' possibile fare sì che un'immagine serva da link per diverse pagine, a seconda del punto in cui l'immagine viene cliccata.
- ◆ Questa tecnica è chiamata imagemapping ("mappatura dell'immagine").
- ◆ Bisogna soltanto specificare i collegamenti tra le aree dell'immagine e le mete del link.
- ◆ Nell'esempio qui sotto, se posizioni il mouse sull'angolo in alto a sinistra ti colleghi con yahoo... e se vai sull'angolo in basso a destra... vai su hotbot.



```
  
<map name=example>  
<area shape=Rect Coords=0,0,29,29 Href="http://www.yahoo.com">  
<area shape=Rect Coords=30,30,59,59  
Href="http://www.hotbot.com">  
</map>
```

## Mappatura delle immagini (2)

- ◆ Nell'esempio precedente si sono usate soltanto mappature rettangolari. Le forme possibili sono:
  - `<area shape=rect coords= x1,y1, x2,y2 Href="http://www.domain.com">`
  - `<area shape=circle coords= x1,y1, x2,y2 Href="http://www.domain.com">`
  - `<area shape=polygon coords= x1,y1, x2,y2, ..., xn,yn Href="http://www.domain.com">`
- ◆ Di solito non si calcolano le coordinate a mano. Esistono degli strumenti eccellenti che automatizzano questa attività.

## Indice degli argomenti (2)

- ◆ Immagini
- ◆ Tabelle
- ◆ Moduli
- ◆ Fogli di stile

# Immagini <sup>(1)</sup>

## <IMG>

- ◆ Viene utilizzato per inserire immagini in linea
- ◆ Non c'è tag di chiusura
- ◆ Sintassi:

```
<IMG [src="[URL]nome file"] [alt="descrizione testuale"]  
    [align="top|middle|bottom|left|center|right"] [width="pixel"]  
    [height="pixel"] [border="pixel"] [hspace="pixel"] [vspace="pixel"]>
```

Esempio:

```

```

## Immagini (2)

<IMG>

### ◆ Sintassi:

```
<IMG [src="[URL]nome file"] [alt="descrizione testuale"]  
    [align="top|middle|bottom|left|center|right"] [width="pixel"]  
    [height="pixel"] [border="pixel"] [hspace="pixel"] [vspace="pixel"]>
```

Src indica dove si trova il file dell'immagine

Alt è usato per indicare una descrizione alternativa alla foto. È usato per vecchi browser o per quelli nuovi in cui l'utente ha disabilitato le immagini

Esempio:

```

```

## Immagini <sup>(3)</sup>

<IMG>

### ◆ Sintassi:

```
<IMG [src="[URL]nome file"] [alt="descrizione testuale"]  
    [align="top|middle|bottom|left|center|right"] [width="pixel"]  
    [height="pixel"] [border="pixel"] [hspace="pixel"] [vspace="pixel"]>
```

Align è usato per il posizionamento dell'immagine

Width e Height impostano lo spazio orizzontale e verticale per l'immagine

Border definisce la dimensione del bordo

Esempio:

```

```

```

```

# Immagini <sup>(4)</sup>

<IMG>

◆ Sintassi:

```
<IMG [src="[URL]nome file"] [alt="descrizione testuale"]  
    [align="top|middle|bottom|left|center|right"] [width="pixel"]  
    [height="pixel"] [border="pixel"] [hspace="pixel"] [vspace="pixel"]>
```

Hspace e Vspace vengono usati per impostare una zona “cuscinetto” attorno all’immagine

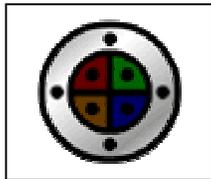
## Spazio intorno alle immagini (1)

- ◆ Si può facilmente aggiungere spazio sopra e sotto una specifica immagini con l'attributo **Vspace**.
- ◆ In modo simile si può aggiungere spazio a destra e a sinistra della medesima immagine per mezzo dell'attributo **Hspace**.
- ◆ L'esempio seguente sull'uso di questi attributi

```

```

genera il risultato riportato di seguito

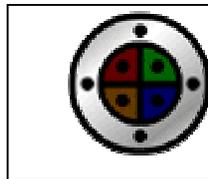


## Spazio intorno alle immagini (2)

- ◆ Gli attributi `Vspace` e `Hspace` obbligano ad aggiungere la stessa quantità di spazio a ogni lato dell'immagine (sopra e sotto - oppure a destra e sinistra).
- ◆ Per aggirare il problema, se si vuole aggiungere spazio solo ad un lato dell'immagine, si deve usare un'immagine gif trasparente di 1x1 pixel.
- ◆ Se, per esempio, si desidera uno spazio di 10 pixel sul lato destro dell'immagine, puoi usare l'immagine trasparente (`pixel.gif`) in questo modo:

```

```



## Spazio intorno alle immagini (3)

- ◆ L'immagine gif trasparente di 1x1 pixel viene semplicemente "tirata" fino a raggiungere la dimensione desiderata.
- ◆ Il "trucchetto" dell' 1x1 pixel è probabilmente uno degli stratagemmi più usati sull'intera rete per risolvere problemi.
- ◆ I motivi sono ovvi: funziona su tutti i browsers ed assicura una assoluta precisione di pixel nel progetto!

## Formati per le immagini (1)

- ◆ I computer immagazzinano le immagini in diverse maniere.
- ◆ Alcuni metodi si basano sulla massima compressione possibile dell'immagine.
- ◆ Uno dei maggiori problemi circa l'uso delle immagini nei siti web consiste nel fatto che scaricare le immagini richiede molto più tempo rispetto al testo.
- ◆ Per ridurre il più possibile il tempo necessario a scaricare, vengono usati dei formati di compressione di immagini. Due formati largamente utilizzati sul web sono GIF e JPEG.

## Formati per le immagini (2)

GIF	JPEG
Può gestire aree trasparenti	Non può gestire aree trasparenti
Questo formato non va bene per comprimere fotografie	Eccellente nella compressione di fotografie e immagini complesse
In generale, è eccellente per titoli, bottoni e clipart	In generale, non è adatto per titoli, bottoni e clipart.

- ◆ Dalla tabella si deduce che:
  - Titoli, bottoni, divisori, clipart e altre semplici immagini funzionano meglio in **GIF**.
  - Fotografie e immagini complesse normalmente funzionano meglio in **JPG**.

## Indice degli argomenti (2)

- ◆ Immagini
- ◆ Tabelle
- ◆ Moduli
- ◆ Fogli di stile

# Tabelle <sup>(1)</sup>

## <TABLE>

- ◆ Questo tag consente la creazione di tabelle
- ◆ Possiede tag di apertura e di chiusura
- ◆ Sintassi:

```
<TABLE [align="left|center|right"] [width="pixels|percent"] [border="dimens"]  
[cellspacing="dimens"] [cellpadding="dimens"]>
```

## Tabelle (2)

### <TABLE>

#### ◆ Sintassi:

```
<TABLE [align="left|center|right"] [width="pixels|percent"]  
      [border="dimens"] [cellspacing="dimens"] [cellpadding="dimens"]>
```

- ◆ Align allinea la tabella
- ◆ Width definisce l'ampiezza della tabella
- ◆ Border definisce l'ampiezza del bordo esterno
- ◆ Cellspacing definisce l'ampiezza del bordo tra le celle
- ◆ Cellpadding definisce lo spazio tra i contenuti di cella e i bordi

## Tabelle <sup>(3)</sup>

<TABLE>

◆ Sintassi per la creazione di tabelle:

<TABLE>

<CAPTION> didascalia </CAPTION>

<TR><TH>titolo cella1 <TH> ... <TH>titolo cellaN<TH></TR>

<TR><TD>titolo cella1 <TD> ... <TD>titolo cellaN<TD></TR>

....

<TR><TD>titolo cella1 <TD> ... <TD>titolo cellaN<TD></TR>

</TABLE>

## Tabelle (4)

### <CAPTION>

- ◆ Tramite questo tag è possibile aggiungere una breve descrizione alla tabella immediatamente dopo il tag di apertura di <TABLE>
- ◆ Sintassi:  
<CAPTION [align="top|bottom"]> didascalia <CAPTION>

## Tabelle (5)

<TR>

- ◆ L'inizio di una riga è contrassegnato da questo tag e si chiude con quello di chiusura
- ◆ Sintassi:

<TR [align="left|center|right"] [valign="top|middle|bottom"]>

## Tabelle (6)

<TH>

◆ Rappresenta una cella della tabella contenente un titolo (o per la riga o per la colonna)

◆ Sintassi:

```
<TH [width="pixel|percentuale"] [height="pixel"] [rowspan="numero righe"]  
[colspan="numero colonne"] [align="left|center|right"]  
[valign="top|middle|bottom"] [nowrap]>
```

## Tabelle (7)

<TH>

### ◆ Sintassi:

```
<TH [width="pixel|percentuale"] [height="pixel"] [rowspan="numero  
  righe"] [colspan="numero colonne"] [align="left|center|right"]  
  [valign="top|middle|bottom"] [nowrap]>
```

- ◆ Width è la larghezza della tabella
- ◆ Height è l'altezza della tabella
- ◆ Rowspan è il numero di righe unite insieme
- ◆ Colspan è il numero di colonne unite insieme
- ◆ Nowrap disabilita l' "a capo" automatico

## Tabelle (8)

<TD>

- ◆ Rappresenta una cella della tabella contenente il testo
- ◆ Sintassi:

```
<TD [width="pixel|percentuale"] [height="pixel"] [rowspan="numero righe"]  
[colspan="numero colonne"] [align="left|center|right"]  
[valign="top|middle|bottom"] [nowrap]>
```

## Tabelle (9)

<TD>

### ◆ Sintassi:

```
<TD [width="pixel|percentuale"] [height="pixel"] [rowspan="numero  
  righe"] [colspan="numero colonne"] [align="left|center|right"]  
  [valign="top|middle|bottom"] [nowrap]>
```

- ◆ Width è la larghezza della tabella
- ◆ Height è l'altezza della tabella
- ◆ Rowspan è il numero di righe unite insieme
- ◆ Colspan è il numero di colonne unite insieme
- ◆ Nowrap disabilita l' "a capo" automatico

# Esempi di tabelle (1)

```
<table>
```

```
  <tr>Questa è la prima riga.</tr>
```

```
  <tr>Questa è la seconda riga.</tr>
```

```
</table>
```

**Genera**

Questa è la prima riga
Questa è la seconda riga

## Esempi di tabelle (2)

**<table>**

**<tr> <td>Prima riga, sinistra.</td> <td>Prima riga, destra.</td> </tr>**

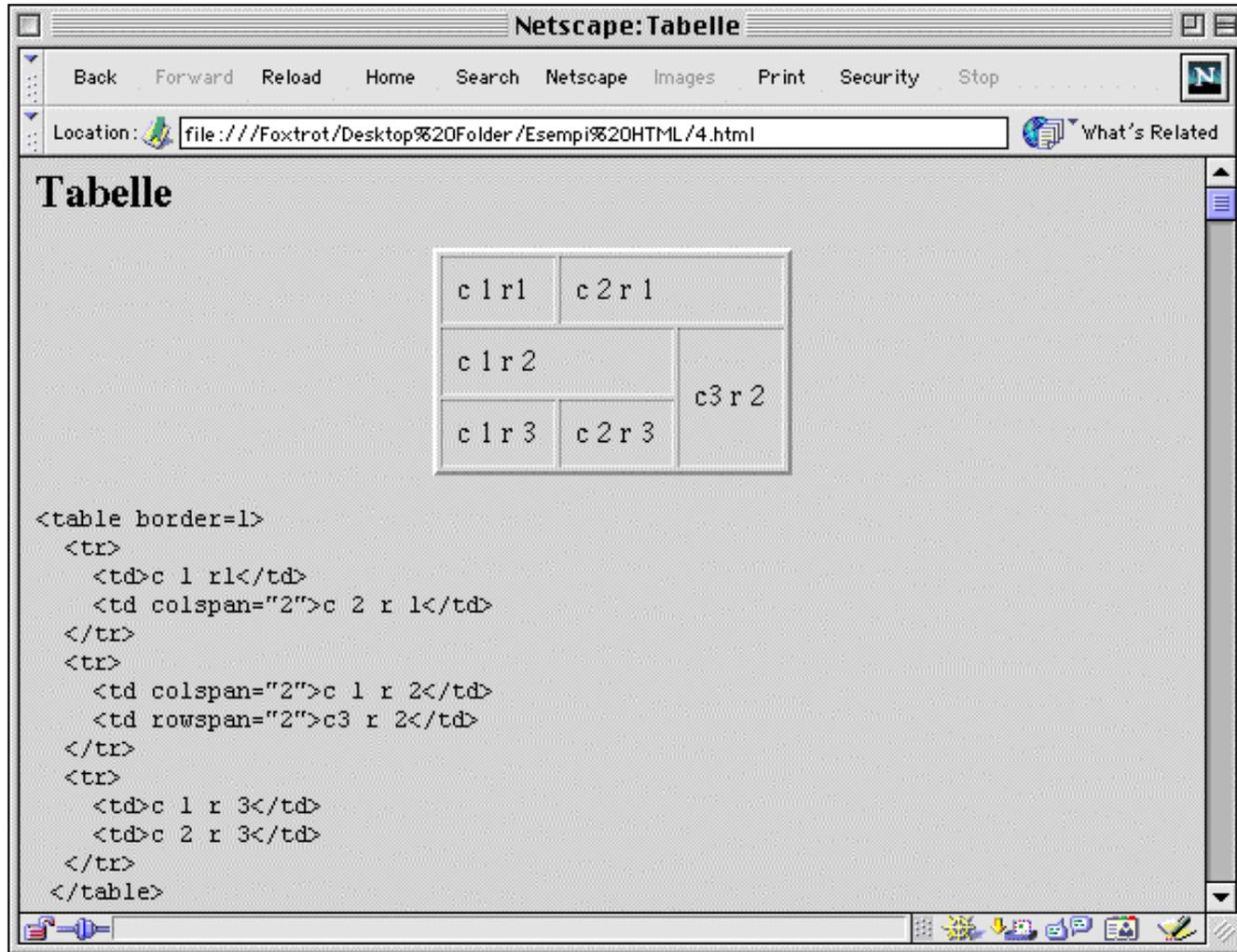
**<tr> <td>Seconda riga, sinistra.</td> <td>Seconda riga, destra.</td> </tr>**

**</table>**

**Genera**

Prima riga, sinistra	Prima riga, destra
Seconda riga, sinistra	Seconda riga, destra

# Una tabella più complessa (1)



The screenshot shows a Netscape browser window titled "Netscape: Tabelle". The address bar displays the file path: `file:///Foxtrot/Desktop%20Folder/Esempi%20HTML/4.html`. The main content area shows a table with the following structure:

c 1 r 1	c 2 r 1	
c 1 r 2		c 3 r 2
c 1 r 3	c 2 r 3	

Below the table, the HTML code is displayed:

```
<table border=1>
  <tr>
    <td>c 1 r 1</td>
    <td colspan="2">c 2 r 1</td>
  </tr>
  <tr>
    <td colspan="2">c 1 r 2</td>
    <td rowspan="2">c 3 r 2</td>
  </tr>
  <tr>
    <td>c 1 r 3</td>
    <td>c 2 r 3</td>
  </tr>
</table>
```

## Una tabella più complessa (2)

```
<table border=2 align=center>
  <tr>
    <td>c 1 r1</td>
    <td colspan="2">c 2 r 1</td>
  </tr>
  <tr>
    <td colspan="2">c 1 r 2</td>
    <td rowspan="2">c3 r 2</td>
  </tr>
  <tr>
    <td>c 1 r 3</td>
    <td>c 2 r 3</td>
  </tr>
</table>
```

## Indice degli argomenti (2)

- ◆ Immagini
- ◆ Tabelle
- ◆ Moduli
- ◆ Fogli di stile

## Moduli <sup>(1)</sup>

### <FORM>

- ◆ Per creare un modulo bisogna utilizzare questo tag
- ◆ Possiede un tag di apertura e uno di chiusura
- ◆ Fra questi tag bisogna inserire gli elementi del form mediante il tag <INPUT> <TEXTAREA> <SELECT> (che vedremo più avanti)

## Moduli (2)

### <FORM>

#### ◆ Sintassi:

```
<FORM [method="get|post"] [action="URL script|URL server page"]  
      [enctype="tipo di documento MIME"]>
```

Method definisce il metodo di trasferimento dei dati contenuti nella form

Action specifica l'indirizzo a cui spedire le informazioni: può essere l'indirizzo di un CGI, JSP, ASP, PHP, ecc...

Enctype stabilisce il tipo di documento che si sta trasferendo

## Moduli (3)

### <INPUT>

- ◆ Questo tag viene utilizzato per inserire elementi nella form, ad esempio: pulsanti, immagini, caselle di controllo, password, campi di testo, ecc.
- ◆ Non ha tag di chiusura
- ◆ Sintassi:

<INPUT type="tipo elemento" name="nome campo" [value="valore del campo"]>

### <INPUT>

#### ◆ Sintassi:

<INPUT

```
type="text|password|hidden|checkbox|radio|submit|reset|button|file"  
name="nome campo" [value="valore del campo"]>
```

Type="text" campi usati per inserire testo.

Attributi aggiuntivi: maxlength (lunghezza massima del campo, caratteri inseribili), size (ampiezza, in caratteri, della casella nella pagina, caratteri visualizzabili)

## Moduli (5)

### <INPUT>

#### ◆ Sintassi:

```
<INPUT type="text|password|hidden|checkbox|radio|submit|reset|button|file"  
       name="nome campo" [value="valore del campo"]>
```

Type="password" campi usati per inserire testo (non in chiaro, input nascosto da asterischi)

Ha gli stessi attributi del campo "text"

## Moduli (6)

### <INPUT>

#### ◆ Sintassi:

```
<INPUT type="text|password|hidden|checkbox|radio|submit|reset|button|file"  
       name="nome campo" [value="valore del campo"]>
```

Type="hidden" campi usati per inserire testo che non viene visualizzato: è testo nascosto

Usato per inviare informazioni aggiuntive ad uno script

### <INPUT>

#### ◆ Sintassi:

<INPUT

```
type="text|password|hidden|checkbox|radio|submit|reset|button|file"  
name="nome campo" [value="valore del campo"]>
```

Type="checkbox" campi usati per inserire risposte tipo "si/no"  
"vero/falso" a scelta multipla

Usato per inviare informazioni aggiuntive ad uno script

Attributi aggiuntivi: checked (controlla lo stato iniziale della casella)

## Moduli (8)

### <INPUT>

#### ◆ Sintassi:

```
<INPUT type="text|password|hidden|checkbox|radio|submit|reset|button|file"  
       name="nome campo" [value="valore del campo"]>
```

Type="radio" sono simili alle caselle di controllo "checkbox", ma le scelte sono esclusive: si può selezionare una sola scelta per volta

Ha gli stessi attributi dei "checkbox"

## Moduli (9)

### <INPUT>

#### ◆ Sintassi:

```
<INPUT type="text|password|hidden|checkbox|radio|submit|reset|button|file"  
       name="nome campo" [value="valore del campo"]>
```

Type="submit/reset/button" questi sono i pulsanti utilizzati per la gestione dei dati, ovvero: invio, cancellazione e operazioni definibili mediante script lato client

# Moduli (10)

## <INPUT>

### ◆ Sintassi:

```
<INPUT type="text|password|hidden|checkbox|radio|submit|reset|button|file"  
       name="nome campo" [value="valore del campo"]>
```

Type="file" questa opzione consente la selezione di un file (nel file system del client) per l'operazione di "upload"

### <TEXTAREA>

- ◆ È simile al campo “text” del tag <INPUT>, ma consente di inserire un brano o testo molto lungo
- ◆ Se si utilizza questo tag bisogna scegliere il metodo “post” per l’invio dei dati del modulo
- ◆ Ha un tag di apertura e uno di chiusura

### <TEXTAREA>

- ◆ Se necessita di essere preriempito con del testo, questo deve essere inserito fra il tag di apertura e quello di chiusura
- ◆ Sintassi:  
<TEXTAREA name="nome del campo" [rows="numero di righe"] [cols="numero di colonne"]> testo da visualizzare </TEXTAREA>

### <SELECT>

- ◆ Con questo tag è possibile mostrare un elenco scorrevole di possibili opzioni selezionabili (anche a scelta multipla)
- ◆ Ha un tag di apertura e uno di chiusura
- ◆ Le singole voci della lista sono identificate dal tag <OPTION> (di apertura e chiusura)

## <SELECT>

### ◆ Sintassi:

```
<SELECT name="nome campo" [size="elementi visibili"] [multiple]>
```

```
<OPTION [selected]> testo opzione 1 </OPTION>
```

```
<OPTION [selected]> testo opzione 2 </OPTION>
```

...

```
<OPTION [selected]> testo opzione N </OPTION>
```

```
</SELECT>
```

## <SELECT>

### ◆ Sintassi:

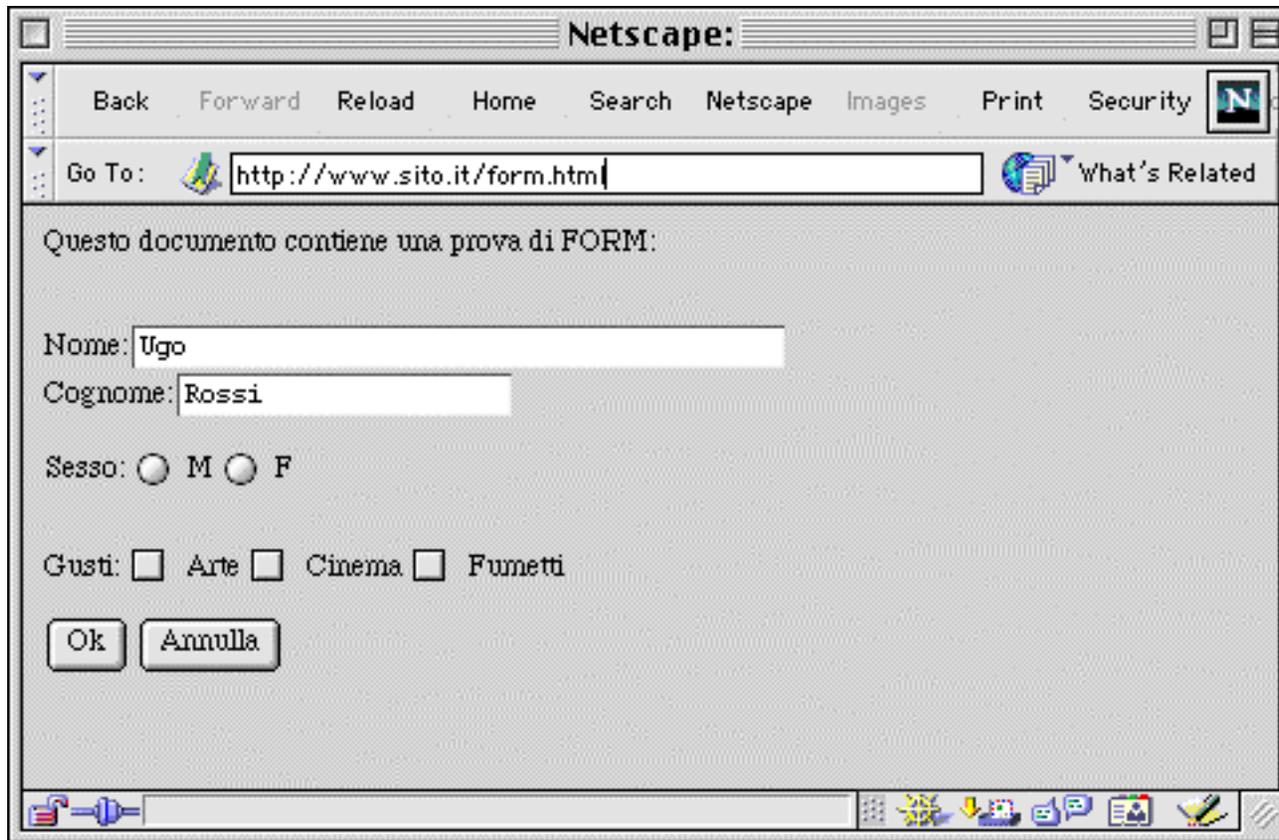
<SELECT name="nome campo" [size="elementi visibili"] [multiple]>

<OPTION [selected]>

Multiple consente di selezionare più elementi della lista

Selected permette di definire gli elementi che sono selezionati per default

# Esempio di form



# Il codice della form

- ◆ `<P>Questo documento contiene una prova di FORM:</P>`
- ◆ `<FORM METHOD="GET" ACTION="http://www.sito.it/cgi-bin/a.pl">`
- ◆ `<BR>Nome:<INPUT TYPE="text" NAME="Nome" VALUE="Ugo" SIZE=40>`
- ◆ `<BR>Cognome:<INPUT TYPE="text" NAME="Cognome" VALUE="Rossi">`
- ◆ `<P>Sesso: <INPUT TYPE="radio" NAME="Sesso" VALUE="M"> M`
- ◆ `<INPUT TYPE="radio" NAME="Sesso" VALUE="F"> F </P>`
- ◆ `<BR>Gusti:`
- ◆ `<INPUT TYPE="checkbox" NAME="Gusti" VALUE="Arte"> Arte`
- ◆ `<INPUT TYPE="checkbox" NAME="Gusti" VALUE="Cinema"> Cinema`
- ◆ `<INPUT TYPE="checkbox" NAME="Gusti" VALUE="Fumetti"> Fumetti`
- ◆ `<P><INPUT TYPE=submit NAME="Submit" VALUE="Ok">`
- ◆ `<INPUT TYPE=reset NAME="Cancel" VALUE="Annulla">`
- ◆ `</FORM>`

# I tag del form (1)

## ◆ FORM

Raggruppa tutti i controlli che debbono raccogliere data per un destinatario. Attributi:

- ACTION: l'URL dell'applicazione server-side che riceverà i dati
- METHOD: il metodo HTTP che deve essere usato per i dati
- NAME: un nome univoco per il form
- ENCTYPE: il metodo di codifica dei dati da usare. Per default si usa `'application/x-www-form-urlencoded'`.

# La codifica application/x-www-form-urlencoded

- ◆ E' un estensione della codifica degli URI prevista nel RFC 2396 (Sintassi degli URI)
  - i codici non alfanumerici sono sostituiti da '%HH' (HH: codice esadecimale del carattere),
  - gli spazi sono sostituiti da '+',
  - i nomi dei controlli sono separati da '&',
  - il valore è separato dal nome da '='

## I tag del form (2)

### ◆ INPUT

Raccoglie la maggior parte dei tipi di controllo disponibili in un form. Attributi:

- TYPE: rappresenta il tipo di controllo da istanziare: text, password, checkbox, radio, submit, reset, file, hidden, image, button
- NAME: il nome che verrà passato all'applicazione server-side insieme al valore inserito
- VALUE: il valore di default (per i controlli di testo) o selezionato (per i controlli di tipo bottone) che viene presentato all'utente e passato all'applicazione server-side
- SIZE e MAXLENGTH: dimensione prevista e lunghezza massima (per i testi)
- DISABLED, CHECKED: per i bottoni, radio e checkbox.

## I tag del form (3)

### ◆ TEXTAREA

Area di inserimento testo. Attributi (oltre a NAME):

- ROWS: il numero di righe previste
- COLS: il numero di colonne previste

### ◆ SELECT, OPTGROUP

Una lista di opzioni e sotto-opzioni. Attributi (oltre a NAME):

- SIZE: il numero di opzioni visibili
- MULTIPLE: Vero se la lista permette selezioni multiple.

### ◆ OPTION

Una opzione in un SELECT o in un OPTGROUP.

- SELECTED: vero se l'elemento è selezionato all'avvio
- LABEL: scritta visibile all'utente
- VALUE: valore passato all'applicazione server-side

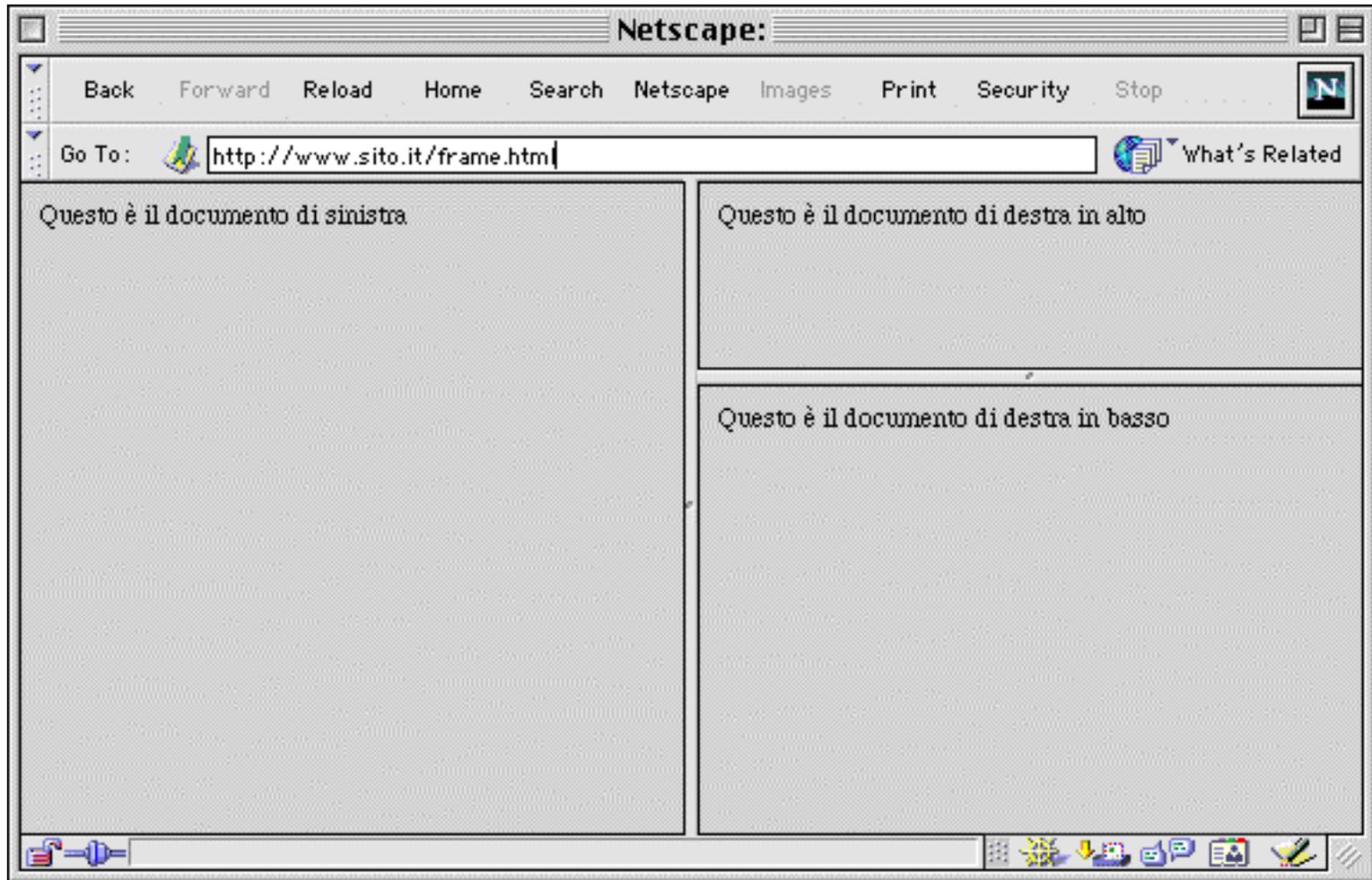
# Indice degli argomenti <sup>(2)</sup>

- ◆ Frame

## I frame

- ◆ I frame servono per dividere la finestra in più zone indipendenti ed associare a ciascuna di loro un documento diverso.
- ◆ Ogni zona (o frame) ha un nome, che viene usata per specificare in quale zona viene visualizzato la destinazione di un link.
- ◆ Il tag FRAMESET introduce una serie di frame o verticali o orizzontali. FRAMESET può contenere annidati altri FRAMESET (per cambiare l'orientamento).
- ◆ La definizione dei link A cambia per rendere possibile precisare il nome di una zona o di una finestra come destinazione del documento.

## Un esempio di frame



# Il codice del frame

◆ <HTML>

```
<FRAMESET COLS="50%,*">
```

```
<FRAME SRC="f2.html" NAME="sinistra">
```

```
<FRAMESET ROWS="30%,*">
```

```
<FRAME SRC="f3.html" NAME="destra alto">
```

```
<FRAME SRC="f4.html" NAME="destra basso">
```

```
</FRAMESET>
```

```
</FRAMESET>
```

```
<NOFRAMES>
```

```
<BODY>
```

```
<P>Testo per browser che non supportano i frame</P>
```

```
</BODY>
```

◆ </HTML>

## I tag dei frame (1)

### ◆FRAMESET

Introduce un gruppo di frame. Attributi:

- ROWS, COLS: la disposizione delle righe o delle colonne del frameset. Il valore è una lista di numeri separati da virgole. Ogni numero identifica la dimensione di un frame. Espresso o in pixel, o in percentuale, o con un asterisco (divisione equa del rimanente spazio)

**<FRAMESET ROWS="30, 30%, \*, \*">: quattro righe, una alta 30 pixel, una alta il 30 per cento dello spazio rimanente, la terza e la quarta si dividono lo spazio rimanente.**

### ◆NOFRAMES

Blocco di dati da visualizzare nel caso non si sappiano visualizzare i frame.

## I tag dei frame (2)

### ◆ FRAME

Introduce un frame. Attributi:

- SRC: l'URL del documento da visualizzare nel frame.
- NAME: il nome del frame, da usare nel target dei link <A>
- FRAMEBORDER, MARGINWIDTH, MARGINHEIGHT: misure di visualizzazione dei margini dei frame
- NORESIZE, SCROLLING: controllano il ridimensionamento e la possibilità di scrolling del frame.

### ◆ IFRAME

Inserisce un frame all'interno di un documento HTML normale (non diviso in frame). Attributi:

- Gli stessi di frame
- ALIGN e WIDTH: controllano la posizione e la dimensione del frame rispetto alla pagina.

## Indice degli argomenti <sup>(2)</sup>

- ◆ Fogli di stile

## Fogli di stile <sup>(1)</sup>

I fogli di stile a cascata o CSS (Cascading Style Sheets), sono un meccanismo per controllare, in maniera più raffinata, l'aspetto grafico degli elementi presenti in una pagina web. Esistono tre modi fondamentali di utilizzare i fogli di stile: in linea, con pagina singola o incorporati e con collegamento a un foglio di stile principale o esterno

## Fogli di stile (2)

**Fogli di stile in linea:** Questo approccio utilizza i tag HTML esistenti in un documento (consente il controllo dei singoli elementi di una pagina), ad esempio:

```
<TABLE style="color: #FF0000; font-style: oblique; font-weight: bold">
```

## Fogli di stile <sup>(3)</sup>

**Fogli di stile incorporati:** Questo modello consente il controllo di singole pagine. Gli attributi di stile sono inseriti tra i tag `<STYLE>` e `</STYLE>` posti all'interno dei tag `<HEAD>` e `</HEAD>` (come spiegato nella slide 11). Ad esempio:

```
<STYLE>  
    H3 {color : #0000FF}  
</STILE>
```

## Fogli di stile <sup>(4)</sup>

**Fogli di stile esterni o collegati:** si deve creare un file di foglio di stile, con estensione .CSS in cui sono definiti gli stili generali con la stessa sintassi utilizzata per gli stili incorporati. Poi occorre collegare al foglio di stile, mediante il tag <LINK>, tutti i documenti che richiedono questi controlli

## Fogli di stile <sup>(5)</sup>

### <STYLE>

- ◆ Come spiegato prima consente di definire i fogli di stile nei tre modi, aparendo come attributo o come tag a seconda del metodo impiegato

## Fogli di stile (6)

### <LINK>

- ◆ Viene utilizzato per richiamare fogli di stile esterni
- ◆ Sintassi:

```
<LINK href="URL del documento collegato"  
rel="Precede|Annotation|Present|History|Made"  
rev="Precede|Annotation|Present|History|Made" title="nome del  
documento" name="nome del collegamento">
```

## Fogli di stile (7)

### <LINK>

#### ◆ Sintassi:

```
<LINK href="URL del documento collegato"  
  rel="Precede|Annotation|Present|History|Made"  
  rev="Precede|Annotation|Present|History|Made" title="nome del  
  documento" name="nome del collegamento">
```

Rel indica la relazione del documento corrente con quello a cui si fa riferimento in "href"

Rev è simile a Rel, indica una relazione inversa tra il documento e l'URL

# Indice degli argomenti

- ◆ I Tag di HEAD

# I tag di HEAD

- ◆ HEAD contiene delle informazioni che sono rilevanti per tutto il documento. Esse sono:
  - TITLE: il titolo del documento
  - ISINDEX: definisce la risorsa come indicizzata (deprecato)
  - BASE: l'URL da usare come base per gli URL relativi
  - LINK: link di documenti a tutto il documento
  - SCRIPT: librerie di script
  - STYLE: librerie di stili
  - META: meta-informazioni sul documento

## I tag di HEAD: BASE: URL relativi ed assoluti (1)

- ◆ Ogni documento HTML visualizzato in un browser ha associato un URL. Questo può appartenere allo schema di naming http://, ftp://, o anche file://. Tipicamente sono schemi gerarchici.
- ◆ Spesso accade che esistano degli oggetti dipendenti dalla pagina (immagini, stili, script, applet, link a pagine secondarie, ecc.), che appartengono allo “stesso dominio” della pagina di partenza.
- ◆ E' data allora possibilità, nello specificare l'URL della risorsa secondaria, di affidarsi ad un URL relativo, che si basa sull'URL del documento di partenza.

## I tag di HEAD: LINK, SCRIPT, STYLE: documenti esterni

- ◆ Con i tag SCRIPT e STYLE si possono definire, rispettivamente, blocchi di funzioni di un linguaggio di script e blocchi di stili di un linguaggio di stylesheet.
- ◆ A volte può essere utile mettere esternamente queste specifiche, e riferirvi esplicitamente.
- ◆ In questo caso si usa il tag LINK, che permette di creare un link esplicito al documento esterno di script e/o di stili.
- ◆

```
<LINK REL="STYLESHEET"
      TYPE="text/css"
      HREF="lib/style1.css">
```
- ◆ N.B.: Netscape lo permette solo per stylesheet. Per gli script richiede qualcosa del tipo:
- ◆

```
<SCRIPT LANGUAGE="JavaScript"
          SRC="lib/script.js">
```
- ◆

```
</SCRIPT>
```

## I tag di HEAD: META: meta-informazioni (1)

- ◆ Le meta-informazioni sono informazioni sul documento, piuttosto che informazioni del documento.
- ◆ Il tag META è un meccanismo generale per specificare meta-informazioni su un documento HTML.
- ◆ Ci sono due tipi di meta-informazioni definibili con il tag META:
  - **Intestazioni HTTP:** la comunicazione HTTP fornisce informazioni sul documento trasmesso, ma il suo controllo richiede accesso al server HTTP. Con il tag META si può invece fornire informazione “stile-HTTP” senza modifiche al server.
  - **Altre meta-informazioni:** i motori di ricerca usano le meta-informazioni (ad esempio “Keyword”) per catalogare ed organizzare al meglio i documenti indicizzati.

## I tag di HEAD: META: meta-informazioni (2)

### ◆ Ad esempio:

- `<META HTTP-EQUIV="Expires" CONTENT="10">`

Il documento viene ricaricato tra dieci secondi (ad esempio con un'immagine pubblicitaria diversa)

- `<META NAME="keyword" CONTENT="sesso, sesso, sesso, soldi, borsa, mp3, musica, sesso, sesso">`

Al documento vengono associate le parole chiave qui specificate, che permettono ai motori di ricerca di classificare il documento secondo le categorie qui precisate. Più accurata e popolare la scelta delle keyword, più volte il documento viene presentato nelle ricerche in una posizione alta.

# Bibliografia

- ◆ HTML 4 Flash  
Tiziano Daniotti – Apogeo
- ◆ HTML 4 Tutto&Oltre  
Rick Darnell – Sams Publishing - Apogeo