

CSS

Università degli Studi di Milano

Storia degli standard web

- ▶ CSS (Cascading Style Sheet): fogli di stile a cascata
- ▶ Uno dei linguaggi fondamentali del W3C
- ▶ CSS parallelo a HTML
 - ▶ HTML specificato per definire il contenuto del sito...
 - ▶ ... non la sua formattazione
 - ▶ Quando con HTML 3.2 sono stati introdotti elementi tipo , lo sviluppo di pagine web è diventato un incubo per gli sviluppatori
 - ▶ Formattazione e colori aggiunti ad ogni singola pagina



Storia degli standard web

91-92	93-94	95-96	97-98	99-00	01-02	03-04	05-06	07-08	09-10	11-12	13-14
HTML 1	HTML 2	HTML 3	HTML 4	XHTML 1					HTML 5		
		CSS 1	CSS 2			Web 2.0			CSS3		
		JS	XML 1.0, DOM	DOM 2		XML 1.1	Ajax		DOM, APIs		

CSS

1996 – CSS 1	W3C Rec
1998 – CSS 2	W3C Rec
1999 – CSS 3	Proposed
2005 – CSS 2.1	W3C Candidate Rec
2001 – CSS 3	W3C Working Draft



Storia dei CSS

- ▶ Storia dei CSS
 - ▶ Prima specifica W3C (CSS1)
 - ▶ Dicembre 1996 diventa W3C Recommendation
 - ▶ Seconda specifica (CSS2)
 - ▶ Maggio 1998 diventa W3C Recommendation
 - ▶ Contiene funzionalità aggiuntive
 - ▶ Revisione della seconda specifica (CSS2.1)
 - ▶ Giugno 2011 diventa W3C Recommendation
 - ▶ Risolve errori, rimuove funzionalità poco supportate o non interoperabili, aggiunge estensioni dei browser
 - ▶ Terza specifica (CSS3)
 - ▶ Primo draft giugno 1999
 - ▶ Giugno 2012 ci sono oltre 50 moduli CSS



A cosa servono

- ▶ Oltrepassano i limiti dell'HTML
- ▶ Finalmente, si può associare uno stile al testo delle pagine (come con un word-processor)
 - ▶ Separazione netta tra struttura/contenuto e stile di visualizzazione/formattazione
 - ▶ Non solo colore o font
- ▶ Linguaggio che descrive la presentazione di un documento HTML
 - ▶ Descrive come gli elementi devono essere mostrati a video, quando stampati o su altri media
 - ▶ Ad esempio, imposta stili diversi per titoli e paragrafi, sfrutta i benefici dell'indentatura e della giustificazione



A cosa servono

- ▶ Gestione del sito facilitata
 - ▶ Riducono il carico di lavoro
 - ▶ Permette di controllare il layout di pagine web multiple in un colpo solo
 - ▶ Se avete impostato uno sfondo in 300 pagine e volete cambiarlo non dovrete più modificare una a una 300 pagine
- ▶ I CSS possono essere separati dal documento
- ▶ Aprite un foglio di stile, cambiate l'immagine
 - ▶ Tutte le pagine che riferiscono quel foglio di stile cambiano formattazione



A cosa servono

- ▶ Il risultato sono pagine più leggere e facili da modificare
- ▶ Milioni di byte di banda risparmiati per la gioia degli utenti
- ▶ Sono uno strumento portentoso per l'accessibilità, anche grazie al fatto di poter essere gestiti con linguaggi di scripting avanzati in grado di modificare con un solo click l'aspetto di una pagina



A cosa servono: alcuni esempi

- ▶ Migliorano gli sfondi
- ▶ Permettono di impostare gli sfondi al centro della pagina
- ▶ Gli sfondi non vengono più replicati
- ▶ Permettono di decidere come usare un'immagine di sfondo: la potete ripetere in una sola direzione, in due o per niente



A cosa servono: alcuni esempi

- ▶ Permettono di distanziare e posizionare gli elementi di una pagina in maniera semplice e intuitiva (gestione dei margini interni ed esterni)
- ▶ Permettono di aggiungere bordi non solo alle tavelle, ma a tutti gli elementi di una pagina
- ▶ ... e molto altro ancora!



Meccanismi, costrutti e selettori

Com'è fatta una regola

- ▶ Un foglio di stile è un insieme di regole
- ▶ h1: selettore
 - ▶ definisce la parte del documento cui verrà applicata la regola
- ▶ {...} blocco delle dichiarazioni
 - ▶ Copie proprietà, valore: definiscono un aspetto dell'elemento da modificare (margini, colore di sfondo, ...) secondo il valore espresso



Com'è fatta una regola

- ▶ Dichiarazione valida
 - ▶ `p {font: 12px Verdana, Arial;}`
- ▶ E' possibile fare più dichiarazioni separate da ;
 - ▶ `p {color: red; text-align: center;}`
- ▶ Esempio errato
 - ▶ `body {color background: black;}`



Proprietà singole e a sintassi abbreviata

- ▶ Per ogni elemento è possibile definire una sintassi singola
 - ▶

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

```
div {margin: 10px 5px 10px 5px;}
```
- ▶ L'esempio sopra imposta i margini di un elemento contenitore div



Selettori

- ▶ Selettore di elementi (type selector)
- ▶ È costituito da uno qualunque degli elementi di HTML
- ▶ Spesso riferito come tag selector invece che type selector
- ▶ Sintassi
 - ▶ `h1 {color: #000000;}`
 - `p {background: white; font: 12px Verdana, Arial, sans-serif;}`
 - `table {width: 200px;}`



Selettori (combinazioni)

- ▶ Possibili combinazioni di elementi
 - ▶ elemento1 elemento2: seleziona tutti gli elemento2 contenuti (discendenti) in elemento1
 - ▶

```
ul li {  
    background-color: yellow;  
}
```
 - ▶ elemento1 > elemento2: seleziona tutti gli elemento2 che hanno come padre un elemento1
 - ▶

```
div > p {  
    background-color: yellow;  
}
```
 - ▶ elemento1 + elemento2: seleziona tutti gli elemento2 che sono piazzati immediatamente dopo un elemento1
 - ▶

```
div + p {  
    background-color: yellow;  
}
```



Selettori (attributi)

▶ Selettori di attributi

- ▶ element[attribute]: seleziona tutti gli elementi con attributo attribute
 - ▶ a[target] {
 background-color: yellow;
}
- ▶ element[attribute=value]: seleziona tutti gli elementi con la coppia (attribute,value)
 - ▶ a[target=_blank] {
 background-color: yellow;
}
- ▶ element[attribute~=value]: seleziona tutti gli elementi che hanno un attributo il cui valore è una sequenza di parole separate da spazio e una delle parole è value
 - ▶ img[title~="xyz"] {
 background-color: yellow;
}
- ▶ element[attribute|=value]: uguale al precedente tranne per il fatto che il valore dell'attributo è una sequenza di parole separate da –



Raggruppare

- ▶ È possibile raggruppare diversi elementi
 - ▶ Elementi separati da una virgola
- ▶ Il raggruppamento è un'operazione molto conveniente
 - ▶ h1 {background: white;}
 - ▶ h2 {background: white;}
 - ▶ h3 {background: white;}
- ▶ Invece di scrivere tre regole separate
 - ▶ h1, h2, h3 {background: white;}
 - ▶ Selettore universale * per tutti gli elementi



Inserire i fogli di stile in un documento

- ▶ CSS esterni, interni, in linea
 - ▶ È esterno un foglio di stile definito in un file separato dal documento (editabili anche con il Blocco Note con estensione .css)
 - ▶ È interno un foglio di stile il cui codice è compreso in quello del documento HTML
 - ▶ È in linea quando lo stile è interno al tag HTML target
- ▶ Rispetto a queste diverse modalità si parla di fogli di stile collegati, incorporati o in linea



CSS esterni – Primo modo

- ▶ Utilizzo dell'elemento <link>
- ▶ Inserimento del tag <link> all'interno della testa (<head>)
- ▶ Sintassi:
 - ▶ <link rel="stylesheet" type="text/css" href="stile.css">



Attributi <link>

- ▶ rel: tipo di relazione tra documento e file collegato (obbligatorio)
 - ▶ due possibili valori: **stylesheet** e **alternate stylesheet**
 - ▶ alternate stylesheet: ulteriori stili usabili (ad es. Firefox: *Visualizza -> stile pagina* permette di scegliere fra multipli stili quando presenti)
- ▶ href: definisce l'URL assoluto o relativo del foglio di stile (obbligatorio)
- ▶ type: identifica il tipo di dati da collegare: **text/css** (obbligatorio)
- ▶ media: supporto (schermo, stampa, etc) cui applicare un particolare foglio di stile (opzionale) (ad es., screen)



Esempio

```
<HTML>
    <HEAD>
        <LINK href="stile.css" type="text/css"
              rel="stylesheet">
    </HEAD>
    <BODY>
        ...
    </BODY>
</HTML>
```



CSS esterni – Secondo modo

- ▶ direttiva @import all'interno dell'elemento <STYLE>
(sempre all'interno di <head>)

- ▶

```
<style>
@import url(stile.css);
</style>
```

- ▶ Funziona con tutti i browser



CSS interni – Primo modo

- ▶ inseriti direttamente l'elemento <STYLE> all'interno della testa <HEAD>
 - ▶ ...<head>

```
<style type="text/css">
    body {
        background: #FFFFCC;
    }
</style>
</head>...
```
- ▶ Attributi
 - ▶ type (obbligatorio)
 - ▶ media (opzionale)
- ▶ Seguono le regole del CSS e la chiusura di </STYLE>



CSS in linea

- ▶ Attributo style
- ▶ La dichiarazione avviene a livello dei singoli tag contenuti nella pagina (fogli di stile in linea)
- ▶ Sintassi: <elemento style="regole_di_stile">
- ▶ Esempio, titolo H1 con testo rosso e sfondo nero:
 - ▶ <h1 style="color: red; background: black;">...</h1>



CSS in linea

- ▶ Stile viene riferito all'interno del singolo tag HTML
- ▶ Sono molto potenti ma poco usati
- ▶ Sovrascrivono sia quelli interni che esterni
- ▶ Esempio
 - ▶ `<h1 style="color:purple">Testo</h1>`



Discussione

- ▶ CSS in linea devono essere definiti per ogni elemento target
 - ▶ Se ho 50 elementi h1 devo definire 50 volte lo stile per farli diventare omogenei
- ▶ CSS interni devono essere definiti per ogni pagina Web target
 - ▶ Se ho 50 pagine web che si devono comportare in maniera omogenea devo definire il CSS interno 50 volte
- ▶ CSS esterni sono definiti una volta, e riferiti in tutte le pagine web target
- ▶ CSS in linea sovrascrive CSS interno, CSS interno sovrascrive CSS esterno



Selettori speciali

- ▶ Senza i selettori speciali i CSS non sarebbero così potenti
- ▶ Class e Id
- ▶ I due selettori devono essere associati ad una stylesheet per non perdere significato



Selettore class

- ▶ Selettori che non mappano direttamente a uno specifico tag
 - ▶ Filtrano elementi sul valore del loro attributo **class**
 - ▶ Le classi applicano la regola a tutti gli elementi della pagina che presentano la proprietà `class="nome_classe"`
 - ▶ Ad esempio cambiare il colore di parola all'interno di un tag paragrafo, o cambiare il colore di alcuni paragrafi (non tutti)
- ▶ In questi casi si usano i selettori class
 - ▶ Permettono di scegliere liberamente il nome del selettore
 - ▶ Meglio se il nome descrive anche il significato (ad es. `caption`, `imageborder`)
 - ▶ Nome preceduto da `.`
 - ▶ Spesso usato per caratterizzare il tag vuoto `` (simile a `<div>` ma inline)



Esempio

- ▶

```
<style type="text/css">
.testorosso {
font: 12px Arial, Helvetica, sans-serif;
color: #FF0000;
}
</style>
```
- ▶ Definizione di un CSS incorporato con nome testorosso
- ▶ Esempio di utilizzo
 - ▶

```
<p class="testorosso">....</p>
```
 - ▶ Il testo all'interno del paragrafo è colorato di rosso



Seconda modalità

- ▶ Con **.nome_della_classe{specifica}** la specifica viene applicata a qualunque elemento con valore **nome_della_classe** per attributo **class**
- ▶ Possibile filtrare solo elementi specifici con **nome_della_classe** per attributo **class** :
 - ▶ `<elemento>.nome_della_classe`
 - ▶ Più restrittivo (si applica solo a contenuto in `<elemento> ... </elemento>`)
 - ▶ `p.testorosso {color: red;}` lo stile verrà applicato solo ai paragrafi che presentino l'attributo `class="testorosso"`



Terza modalità

- ▶ Classi multiple:
 - ▶ `p.testorosso.grassetto {color:red; font-weight:bold;}`
- ▶ Regola applicherà a tutti gli elementi in cui siano presenti (in qualunque ordine) i nomi delle classi definiti.
- ▶ Avranno dunque il testo rosso e in grassetto questi paragrafi:
 - ▶ `<p class="testorosso grassetto">...</p>`
- ▶ ma non questo:
 - ▶ `<p class="grassetto">...</p>`



Esempio

```
body {  
    background: white; font: 12px Verdana, Geneva, Arial,  
    Helvetica, sans-serif  
}  
  
P.classe1 {  
    color: red  
}  
  
.classe2 {  
    color: blue  
}  
  
P.classe3.classe4 {  
    color: green  
}
```



Pseudo classi

- ▶ Pseudo classi usate per definire lo stato di un certo elemento
- ▶ :active è usato per selezionare e impostare lo stile di elementi «attivi»
 - ▶ Lo stile cambia al click del mouse
 - ▶ Usato soprattutto con i link
 - ▶ Per i link si usano i) :link per pagine non visitate, ii) :visited per pagine visitate, iii) :hover quando il mouse passa sopra il link
 - ▶

```
a:active {  
    background-color: yellow;  
}  
  
▶ a:hover {  
    background-color: yellow;  
}
```



Selettore id

- ▶ Identificano lo stile di un singolo elemento all'interno della pagina HTML
 - ▶ Sorta di etichetta di un contenitore
- ▶ Come per i selettori class permettono di scegliere liberamente il nome del selettore
 - ▶ Meglio se il nome descrive anche il significato
- ▶ Nome preceduto da #



Esempio

- ▶

```
<style type="text/css">
#testorosso {
font: 12px Arial, Helvetica, sans-serif;
color: #FF0000;
}
</style>
```
- ▶ Definizione di un CSS incorporato con nome testorosso
- ▶ Esempio di utilizzo
 - ▶

```
<p id="testorosso">....</p>
```
 - ▶ Il testo all'interno del paragrafo è colorato di rosso



Pseudo elementi

- ▶ Usate per specificare parti di un elemento
- ▶ :after o ::after inserisce del contenuto con un certo stile dopo un elemento
 - ▶

```
p::after {  
    content: " - Remember this";  
}
```
- ▶ :before o ::before inserisce del contenuto con un certo stile prima di un elemento
 - ▶

```
p::before {  
    content: " Remember this - ";  
}
```



Pseudo elementi

- ▶ :first-child seleziona l'elemento solo se primo figlio
 - ▶ p:first-child {
background-color: yellow;
}
- ▶ ::first-letter seleziona la prima lettera dell'elemento
 - ▶ p::first-letter {
font-size: 200%;
color: #8A2BE2;
}
- ▶ ::first-line seleziona la prima linea dell'elemento
 - ▶ p::first-line {
background-color: yellow;
}



Combinazione di class e id

- ▶ Si possono combinare ricorsivamente
 - ▶ Anche se poi perdono di significato
- ▶ `#one.two { color: red; }`
 - ▶ Utilità limitata soprattutto nel caso degli id che sono già univoci per definizione
- ▶ `<h1 id="one" class="two">This Should Be Red</h1>`



Differenze

- ▶ Class può essere riutilizzato per qualunque elemento all'interno della pagina HTML
- ▶ Id una volta usato all'interno di un elemento non dovrebbe più essere riutilizzato per gli altri



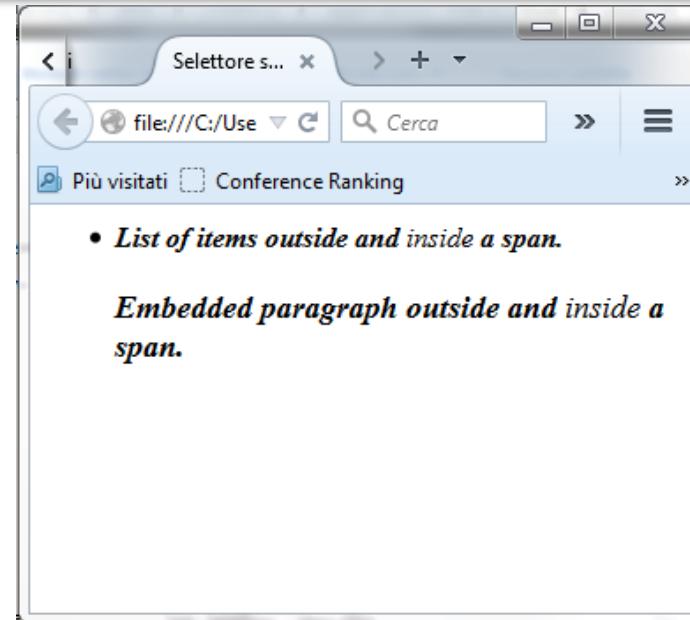
Ereditarietà

- ▶ In generale, le impostazioni di stile applicate ad un elemento vengono ereditate dai suoi discendenti
 - ▶ fino a quando, per un elemento discendente, non si imposti esplicitamente un valore diverso per quella proprietà
 - ▶ Non tutte le proprietà sono ereditate (ad es., formattazione del box model non lo è)

Ereditarietà

```
body {font-weight: bold;}  
li {font-style: italic;}  
p {font-size: larger;}  
span {font-weight: normal;}
```

```
<body>  
<ul>  
<li>  
    List of items outside and <span>inside</span> a span.  
    <p>Embedded paragraph outside and <span>inside</span> a  
    span.</p>  
    </li>  
</ul>  
</body>
```



Ereditarietà esplicita

- ▶ Ereditarietà esplicita
 - ▶ Associare a una proprietà il valore *inherit* per indicare che la proprietà ereditarà il valore dall'elemento padre
 - ▶ Sintassi “proprietà: inherit”
 - ▶ Può essere usato per ogni proprietà ed elemento HTML
- ▶ Esempio
 - ▶

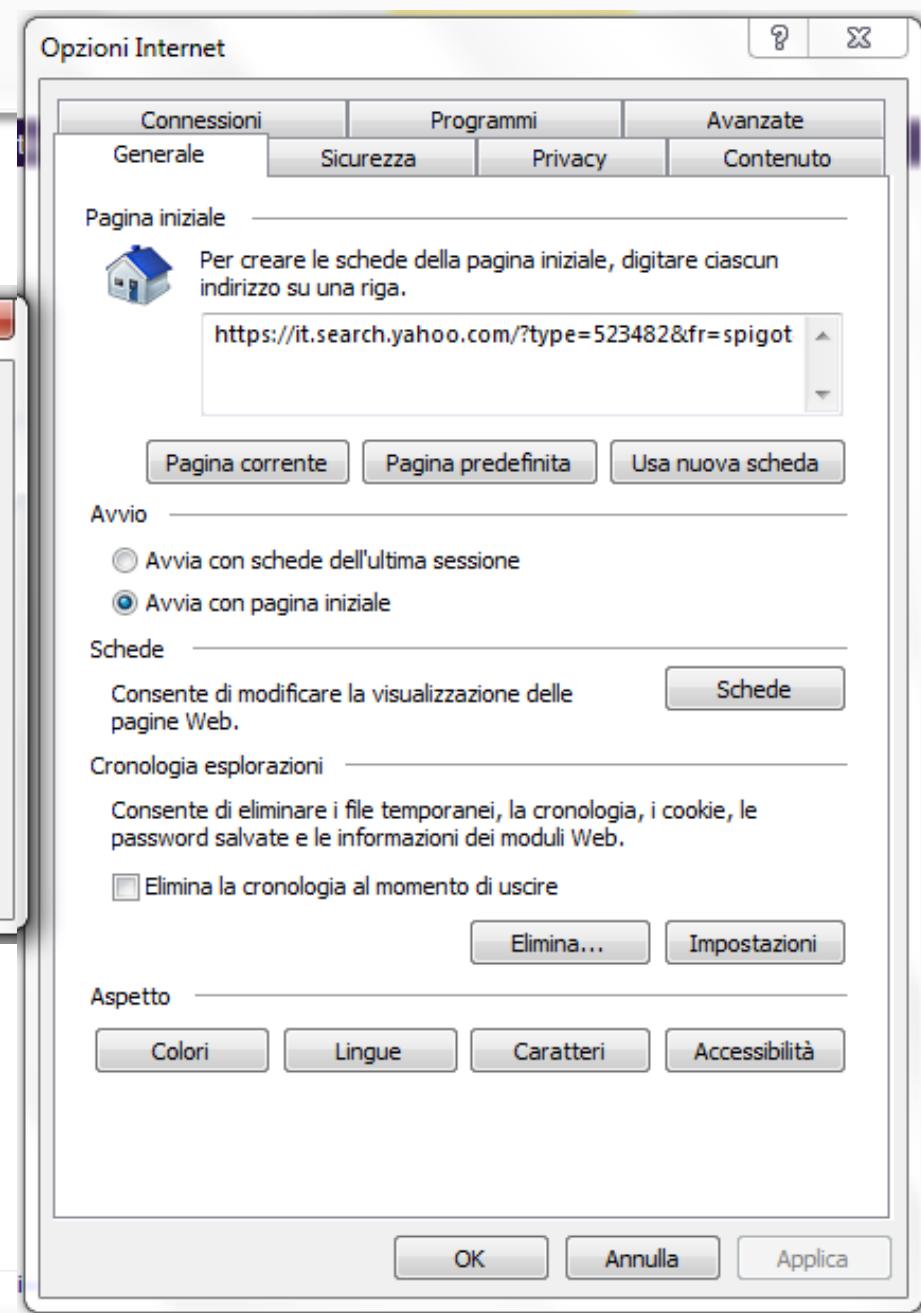
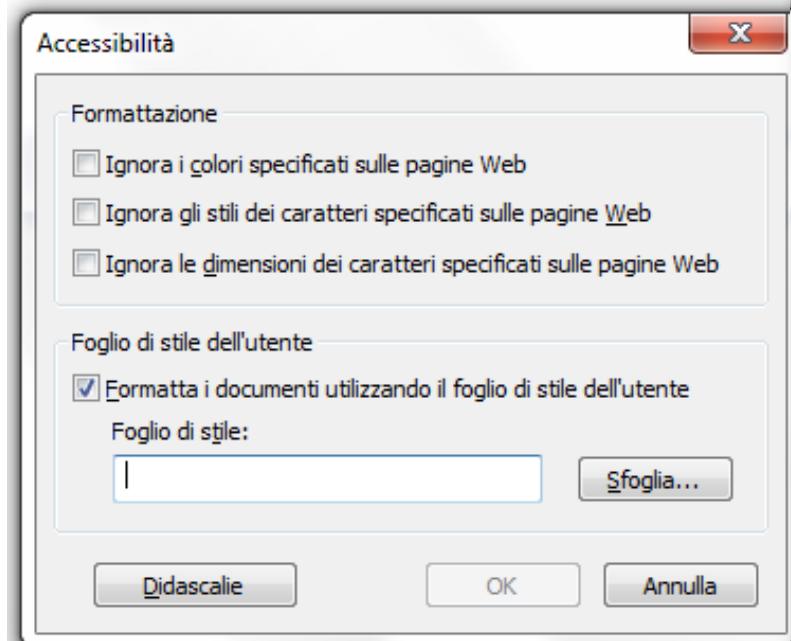
```
span {  
    color: blue;  
    border: 1px solid black;  
}  
  
.extra span {  
    color: inherit;  
}
```



Ereditarietà e risoluzione dei conflitti

- ▶ Conflitti fra stili e regole gestiti secondo diversi criteri
 - ▶ Importanza dei fogli: 1) foglio dell'autore; 2) foglio dell'utente; 3) foglio predefinito del browser
 - ▶ Specificità delle regole: 1) id; 2) classi; 3) singoli elementi
 - ▶ Origine delle regole: 1) stili in linea; 2) stili incorporati; 3) stili collegati
- ▶ I browser eseguono algoritmi che valutano i criteri sopra e stabiliscono il rendering della pagina

User Style Sheet



Formattare testo con CSS

Font e web

- ▶ Nel design di pagine web si può formattare testo in maniera simile alle applicazioni di word processing
- ▶ Necessità di specificare un font che è disponibile nel browser lato utente
 - ▶ Altrimenti problemi di visualizzazione
 - ▶ Il browser sostituisce il font con un altro
- ▶ Font disponibili su (quasi) tutti i browser: arial, verdana, georgia, times new roman, courier, trebuchet, lucida, tahoma, impact



Font e web

- ▶ Soluzione alla mancanza di font comuni è l'utilizzo di alternative
- ▶ Font stack definisce un insieme di font che possono essere utilizzati per stampare il testo a video
 - ▶ Valutazione a cascata: se n non disponibile, prova n+1
 - ▶ Buona norma terminare con un font generico (e.g., serif)
- ▶ Esempio
 - ▶ font-family:"Helvetica Neue", Helvetica, Arial, serif;
 - ▶ Serif permette di usare qualunque font serif come times new roman, georgia



Font-family

- ▶ Definisce il font da utilizzare

- ▶ Esempio CSS (Body)

```
body {  
font-family:"Trebuchet MS", Tahoma, Arial, sans-serif;  
}
```

- ▶ Esempio CSS (paragraph)

```
p {  
font-family:Georgia, "Times New Roman", Times, serif;  
}
```

- ▶ Esempio CSS (heading)

```
h2 {  
font-family:Zapfino;  
}
```



Font-size

- ▶ Definisce la dimensione del testo
- ▶ Ci sono diverse opzioni di definizione
 - ▶ Absolute-size (utile quando si ha certezza sulla dimensione dell'output)
 - ▶ Length
 - ▶ Percentage
 - ▶ Relative size (permette di modificare lato user)
- ▶ Importante perché monitor hanno risoluzione diversa, la dimensione può essere modificata manualmente e visualizzazione fatta attraverso mobile device
 - ▶ Se non specificato diversamente, size di testo regolare è 16px (16px=1em)



Font-size: absolute-size

- ▶ Un insieme di keyword che definiscono dimensioni predefinite
- ▶ Scala di dimensione crescente in accordo alle preferenze utente
 - ▶ xx-small, x-small, small, medium, large, x-large, xx-large

```
body {  
font-family:"Trebuchet MS", Tahoma, Arial, sans-serif;  
font-size:small;  
}
```



Font-size: lenght

- ▶ Un numero seguito da una unità di misura assoluta (cm, mm, in, pt, pc)
 - ▶ Point pt non va bene per il web (è fatto soprattutto per stampanti e si applica male a monitor)
- ▶ Un numero seguito da una unità di misura relativa (em, ex, px)
 - ▶ Pixel px sembrano perfetti (unità di misura dei monitor e loro grafica)
 - ▶ Alcuni browser non ridimensionano caratteri in pixel se sovrascrivono i valori di default

```
body {  
font-family:"Trebuchet MS", Tahoma, Arial, sans-serif;  
font-size:10px;  
}
```



Unità di misura

TABLE 3.4: CSS length unit identifiers.

Identifier	Meaning
in	inches
cm	centimeters
mm	millimeters
pt	points: 1/72-inch
pc	picas: 12 points
px	pixel: typically 1/96-inch (see text).
em	1em is roughly the height of a capital letter in the reference font (see text).
ex	1ex is roughly the height of the lowercase 'x' character in the reference font (see text).



Font-size: percentage

- ▶ Un intero seguito da un %
- ▶ Il valore è la percentuale della dimensione del font dell'oggetto padre

```
body {  
font-family:"Trebuchet MS", Tahoma, Arial, sans-serif;  
font-size:100%;  
}
```



Proprietà aggiuntive

Property	Possible values
<code>font-style</code>	<code>normal</code> (initial value), <code>italic</code> (more cursive than normal), or <code>oblique</code> (more slanted than normal).
<code>font-weight</code>	<code>bold</code> or <code>normal</code> (initial value) are standard values, although other values can be used with font families having multiple gradations of boldness (see CSS2 [W3C-CSS-2.0] for details).
<code>font-variant</code>	<code>small-caps</code> , which displays lowercase characters using uppercase glyphs (small uppercase glyphs if possible), or <code>normal</code> (initial value)



CSS box model

CSS box model

- ▶ Tutti gli elementi HTML sono delle box
- ▶ CSS box model racchiude ogni elemento in un box e vi associa margin, border, padding, content



CSS box model

- ▶ Content: il contenuto della box, dove testo e immagini risiedono
- ▶ Padding: libera un'area attorno al content
 - ▶ È trasparente
- ▶ Border: un bordo che circonda padding e content
- ▶ Margin: libera un'area attorno al bordo
 - ▶ È trasparente



CSS box model: Border

- ▶ Border-style
 - ▶ None, dashed, dotted, solid, double, groove, ridge, inset, outset
 - ▶ Esiste anche border-{top,bottom,left,right}-style
- ▶ Border-width
 - ▶ Definita in pixel o con le keyword thin, medium o thick
- ▶ Border-color
 - ▶ Definita con nome (red, transparent), RBG (rgb(255,0,0)) o hex (#ff0000)



CSS box model: Border

- ▶ Border-* può avere da uno a 4 valori (4: da top in senso orario; 3: top, left&right, bottom; 2: top&bottom, left&right; 1: tutti i lati)
- ▶ **border-style: dotted solid double dashed;**
 - ▶ top border è dotted
 - ▶ right border è solid
 - ▶ bottom border è double
 - ▶ left border è dashed
- ▶ **border-style: dotted solid double;**
 - ▶ top border è dotted
 - ▶ right and left borders sono solid
 - ▶ bottom border è double
- ▶ **border-style: dotted solid;**
 - ▶ top and bottom borders sono dotted
 - ▶ right and left borders sono solid
- ▶ **border-style: dotted;**
 - ▶ Tutti e quattro borders sono dotted



CSS box model: Border

- ▶ Border: attributo scorciatoia
- ▶ Racchiude in una definizione
 - ▶ border-width
 - ▶ border-style (required)
 - ▶ border-color

```
p {  
    border: 5px solid red;  
}
```



CSS box model: Margin

- ▶ Definisce lo spazio tra gli elementi
 - ▶ Spazio vuoto fuori dai bordi
- ▶ Top, right, bottom, e left margin possono essere cambiati indipendentemente
- ▶ Una scorciatoia con l'attributo margin può essere usato per cambiare i margini in una volta sola
 - ▶ Usa le possibili combinazioni di valori dell'attributo border



CSS box model: Margin

► Esempi

```
p {  
    margin-top: 100px;  
    margin-bottom: 100px;  
    margin-right: 50px;  
    margin-left: 50px;  
}
```

```
p {  
    margin: 100px 50px;  
}
```



CSS box model: Padding

- ▶ Definisce lo spazio tra gli elementi border e content
 - ▶ Spazio vuoto all'interno dei bordi
- ▶ Top, right, bottom, e left margin possono essere cambiati indipendentemente
- ▶ Una scorciatoia con l'attributo padding può essere usato per cambiare i margini in una volta sola
 - ▶ Usa le possibili combinazioni di valori dell'attributo border



CSS box model: Padding

► Esempi

```
p {  
    padding-top: 25px;  
    padding-right: 50px;  
    padding-bottom: 25px;  
    padding-left: 50px;  
}
```

```
p {  
    padding: 25px 50px;  
}
```



CSS box model: Esempio

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
    background-color: lightgrey;
    width: 300px;
    padding: 25px;
    border: 25px solid navy;
    margin: 25px;
}
</style>
</head>
<body>
<div>TESTO</div>
</body>
</html>
```

2-box.html



CSS box model: Esempio



Line-height

- ▶ Il testo è mostrato usando line box

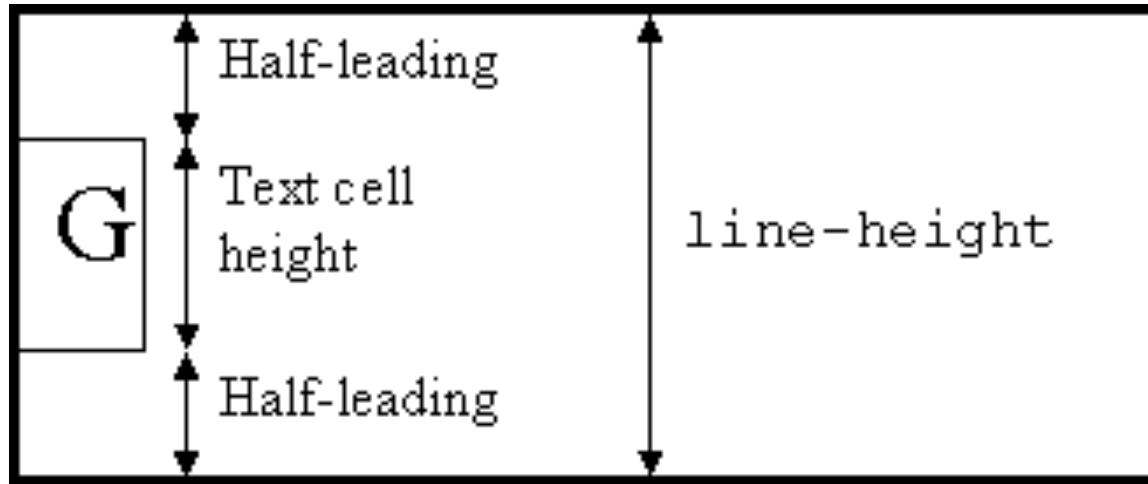


- ▶ L'altezza della line box è data da line-height
 - ▶ Definisce spaziatura tra linee di testo
 - ▶ Valore iniziale: normal (cioè cell height; relazione con altezza dell'unità di misura em è font-specific)
 - ▶ Altri valori
 - ▶ line-height:1.5em
 - ▶ line-height:150%
 - ▶ line-height:1.5



Line-height

- ▶ Quando line-height è più grande di cell height



- ▶ Ereditarietà di line-height
 - ▶ Considera il valore specificato se normal o numero senza unità di misura
 - ▶ Valore calcolato altrimenti



Font-weight, text-transform, letter-spacing

- ▶ Font-weight stabilisce lo spessore dei caratteri
 - ▶ font-weight:normal;
 - ▶ font-weight:lighter;
- ▶ Text-transform agisce e converte i caratteri
 - ▶ text-transform:uppercase;
- ▶ Letter-spacing definisce la spaziatura tra i caratteri del testo
 - ▶ letter-spacing:0.2em



Font

► Scorsciatoia attributo font

```
{font: italic bold 12pt "Helvetica",sans-serif}
```



```
{ font-style: italic;
  font-variant: normal;
  font-weight: bold;
  font-size: 12pt;
  font-height: normal;
  font-family: "Helvetica",sans-serif}
```



Sommario proprietà text

Property	Values
text-decoration	none (initial value), underline, overline, line-through, or space-separated list of values other than none.
letter-spacing	normal (initial value) or a length representing additional space to be included between adjacent letters in words. Negative value indicates space to be removed.
word-spacing	normal (initial value) or a length representing additional space to be included between adjacent words. Negative value indicates space to be removed.
text-transform	none (initial value), capitalize (capitalizes first letter of each word), uppercase (converts all text to uppercase), lowercase (converts all text to lowercase).
text-indent	length (initial value 0) or percentage of box width, possibly negative. Specify for block elements and table cells to indent text within first line box.
text-align	left (initial value for left-to-right contexts), right, center, or justified. Specify for block elements and table cells.
white-space	normal (initial value), pre. Use to indicate whether or not white space should be retained.



Liste HTML

- ▶ Ordered list

- ▶ ``
 - `list 1`
 - `list 2`
 - ...
 - ``

- ▶ Definition list

- ▶ `<dl>`
 - `<dt>list 1</dt>`
 - `<dd>list 2</dd>`
 - ...
 - `</dl>`

- ▶ Unordered list

- ▶ ``
 - `list 1`
 - `list 2`
 - ...
 - ``



CSS e Liste HTML

- ▶ Si possono modificare indentazione, spaziature usando margin e padding
- ▶ Si possono customizzare i bullet
- ▶ Si possono modificare i colori dello sfondo del carattere



CSS e Liste HTML

► Esempi

```
ul {  
    font-size:0.875em;  
    background-color:#E5DAB3;  
}
```

```
li {  
    background-color:#AA6C7E;  
}
```

```
ul li {  
    background-color:#ABC8A5;  
}
```



Flow processing

Normal Flow Layout

- ▶ Una pagina è in **normal flow** quando il posizionamento dei suoi elementi non è modificato esplicitamente
- ▶ In normal flow processing, ogni elemento ha una box corrispondente
 - ▶ L'elemento HTML si riferisce a una box chiamata **initial containing block**
 - ▶ Corrisponde all'intero documento
 - ▶ Box degli elementi figli sono contenuti in box degli elementi padri
 - ▶ Elementi block: fratelli sono messi uno sopra l'altro
 - ▶ Elementi inline: fratelli sono messi uno di fianco all'altro



Normal Flow Layout

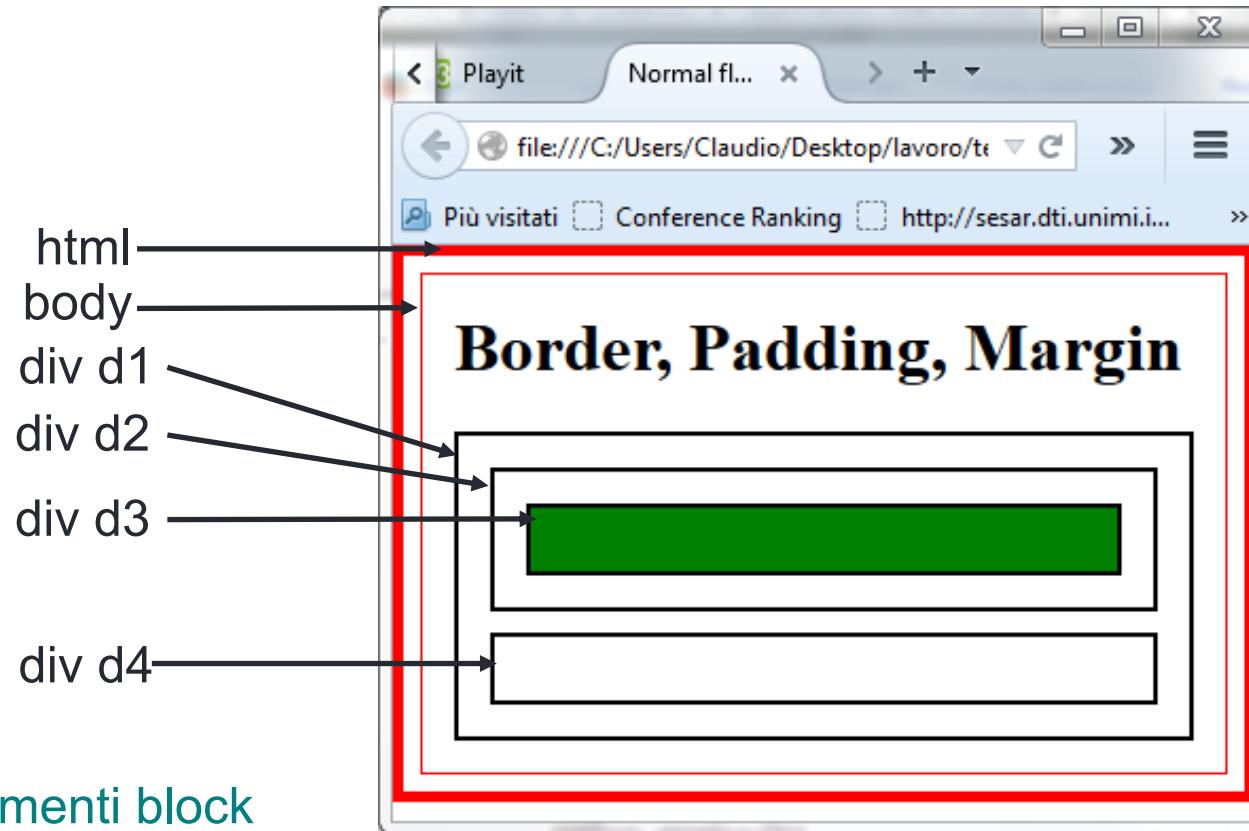
```
html, body {border: solid red thin}
html {border-width: thick}
body {padding: 15px}
div {margin: 0px; padding: 15px; border: solid black 2px}
.shade {background-color: green}
.topMargin {margin-top: 10px}
```

```
<body>
  <div id="d1">
    <div id="d2">
      <div id="d3" class="shade"></div>
    </div>
    <div id="d4" class="topMargin"></div>
  </div>
</body>
```

3-box-div.html



Normal Flow Layout



Gli elementi block
compaiono in ordine
rispetto all'ordinamento
nel document HTML



Normal Flow Layout: proprietà Display

- ▶ Proprietà display
 - ▶ Controlla il layout e dice se e come un elemento deve essere mostrato
 - ▶ Valore di default **block** o **inline**



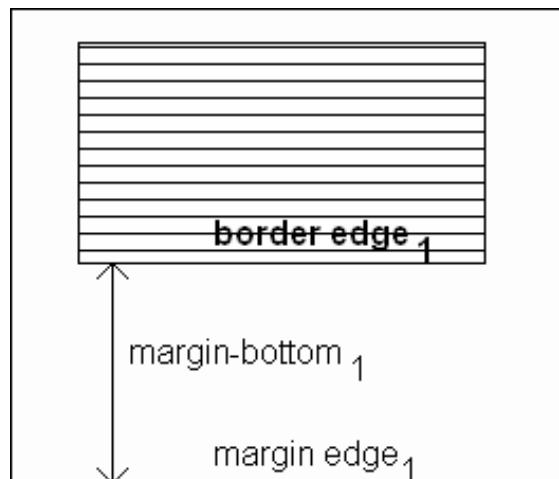
Normal Flow Layout (Display: block)

- ▶ Cosa è un elemento blocco?
 - ▶ Un elemento con proprietà **display** e valore **block** inizia sempre su una riga nuova
 - ▶ Lo style sheet dello user agent (non è CSS) specifica valori di default
 - ▶ Elementi block tradizionali includono *html*, *body*, *p*, *pre*, *div*, *form*, *ol*, *ul*, *dl*, *hr*, *h1-h6*
 - ▶ Molti degli altri elementi eccetto *i* e quelli relativi alle tavelle hanno la proprietà display con valore inline

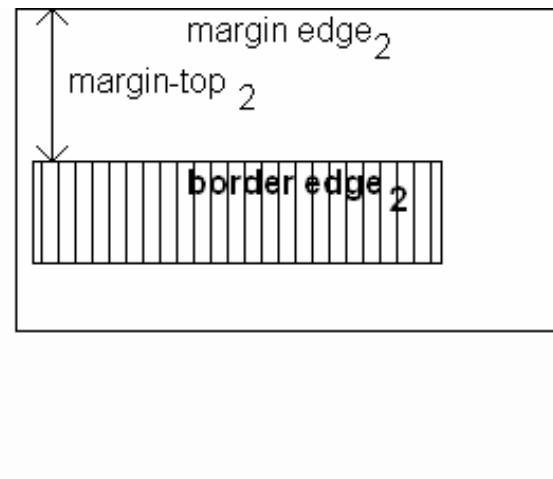


Normal Flow Layout (Display: block)

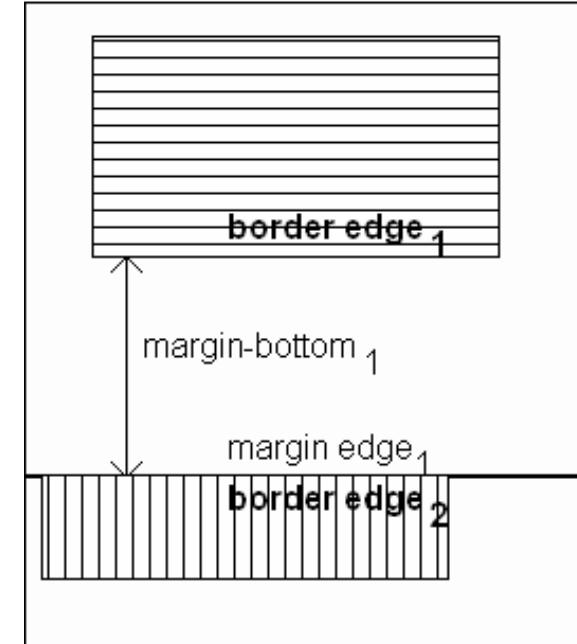
- ▶ Quando elementi block sono impilati, i margini adiacenti sono collassati nel margine più grande
 - ▶ (a) e (b) mostrano i margini dei due elementi
 - ▶ (c) il margine selezionato quando vengono impilati



(a)



(b)

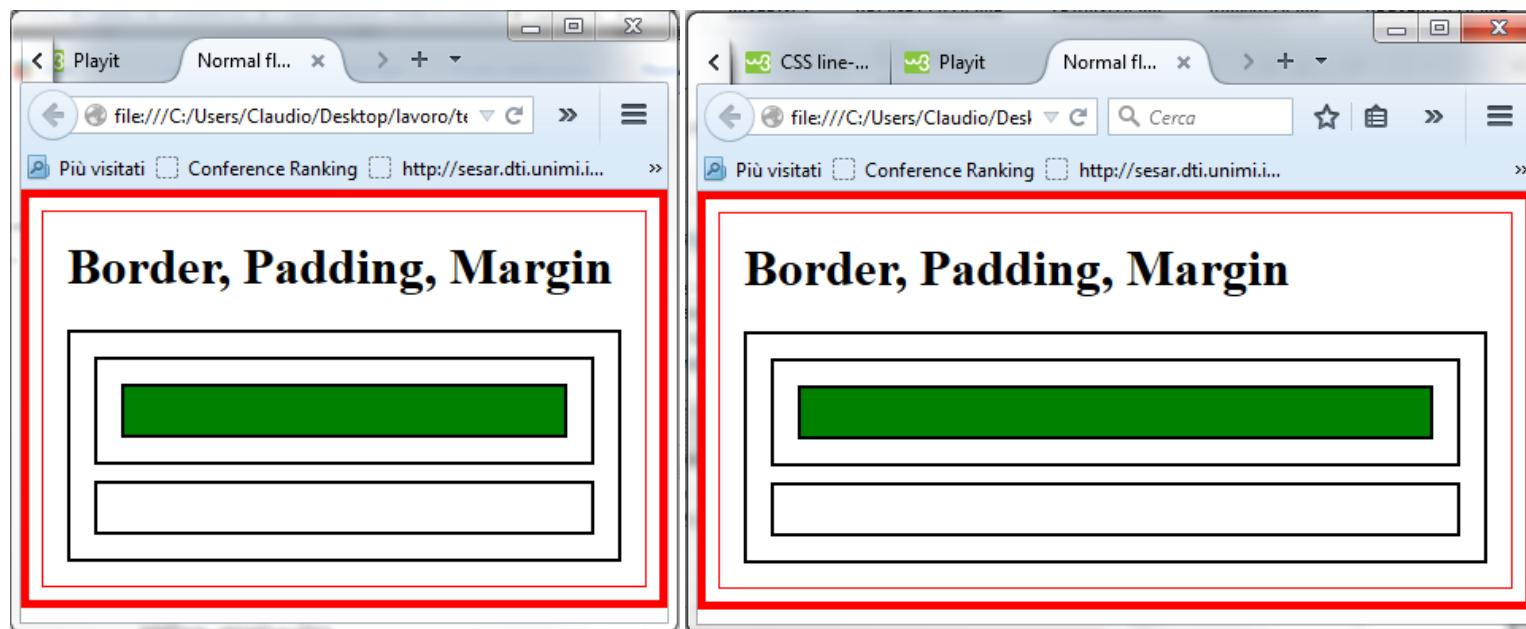


(c)



Normal Flow Layout (Display: block)

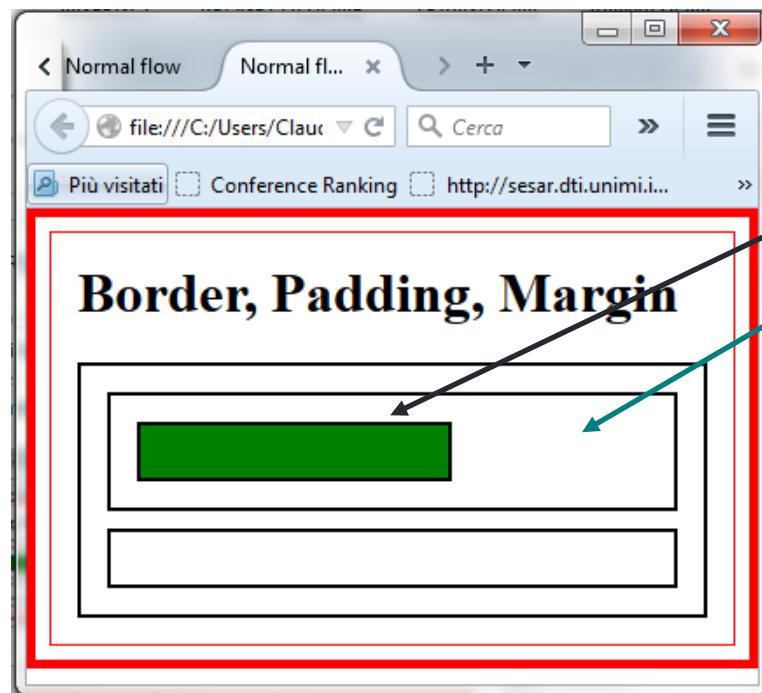
- ▶ Valore iniziale della proprietà width è auto
- ▶ Elementi block occupano l'area di contenuto più larga possibile
 - ▶ Tutto fatto rispettando margini e padding



La larghezza delle box block aumenta all'aumentare dell'area del browser

Normal Flow Layout (Display: block)

- ▶ Possibile specificare la larghezza usando CSS width
 - ▶ Possibile specificare la larghezza come percentuale del contenuto dell'elemento padre

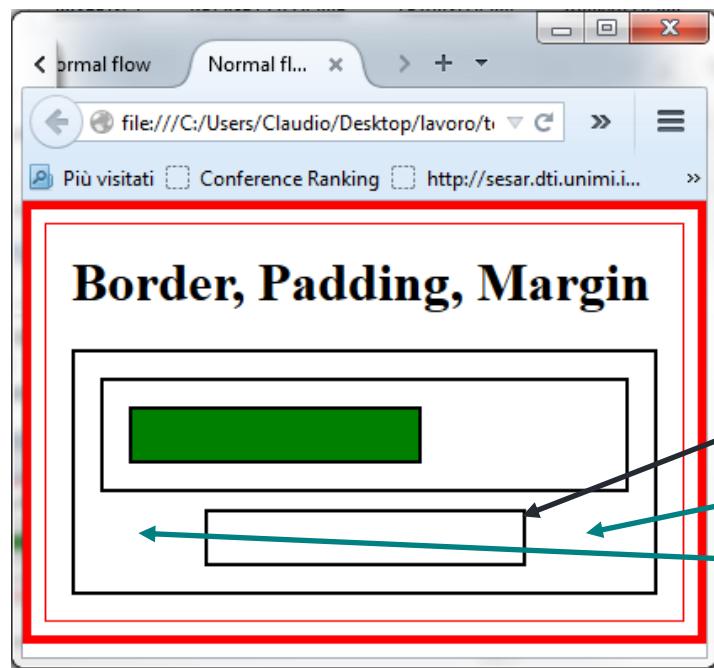


#d3 {width: 50%}

Default: il margine destro viene adattato per gestire il cambiamento della larghezza

Normal Flow Layout (Display: block)

- ▶ Possibile specificare la larghezza usando CSS width
 - ▶ Possibile specificare la lunghezza come percentuale del contenuto dell'elemento padre



#d4 {width: 50%; margin-left: auto; margin-right: auto}

Possibile centrare l'elemento mettendo entrambi i margini ad auto

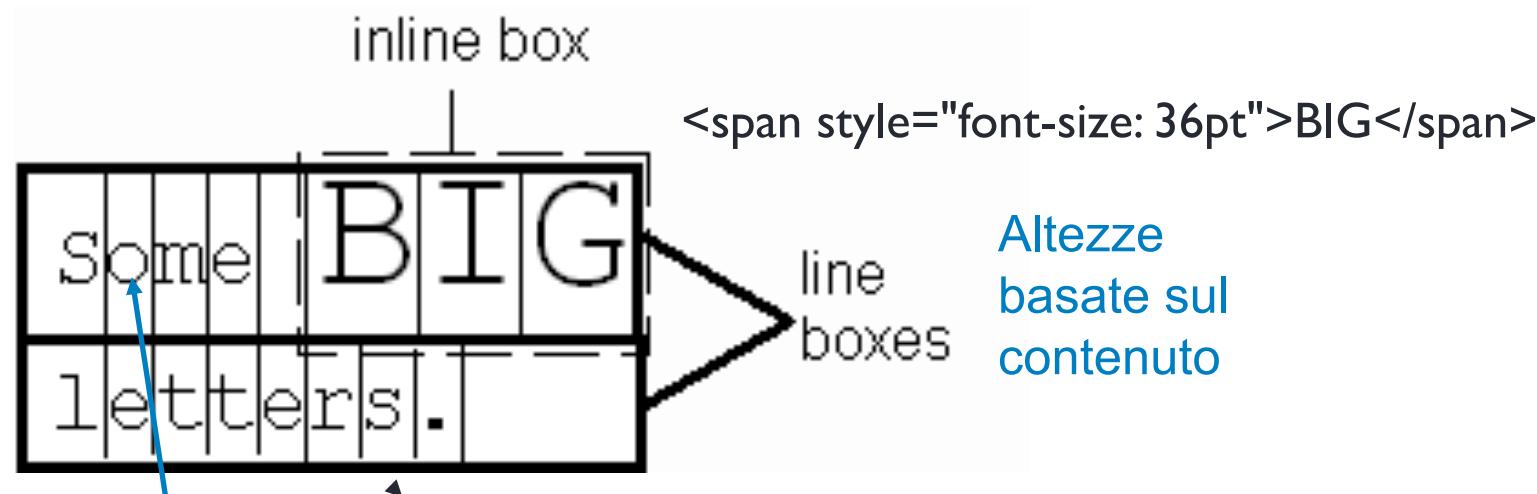
Normal Flow Layout (Display: inline)

- ▶ Elementi inline vengono mostrati uno dopo l'altro nella direzione di scrittura del testo (ad es., da dx a dx)
 - ▶ Anch'essi hanno box, uno dopo l'altro
 - ▶ Se non c'è abbastanza spazio nel blocco contenitore, si spezzano su una nuova linea: queste linee sono **line boxes**
- ▶ Esempio: consideriamo un paragrafo (creato da un elemento **blocco <p>**) contenente testo ed elementi **** e ****
 - ▶ <P>Several emphasized words appear in this sentence, dear.</P>
- ▶ L'elemento **<p>** genera **un box blocco** contenente **5 box inline**
 - ▶ Anonymous: "Several"
 - ▶ EM: "emphasized words"
 - ▶ Anonymous: "appear"
 - ▶ STRONG: "in this"
 - ▶ Anonymous: "sentence, dear."
- ▶ Le **5 box inline** vengono considerate **line box**, contenute nel box dell'elemento **<p>**
 - ▶ Se il box contenitore è sufficientemente largo, tutte le box inline stanno su **una sola line box**
 - ▶ Altrimenti, le box inline vengono spezzate in **più line box**



Normal Flow Layout (Display: inline)

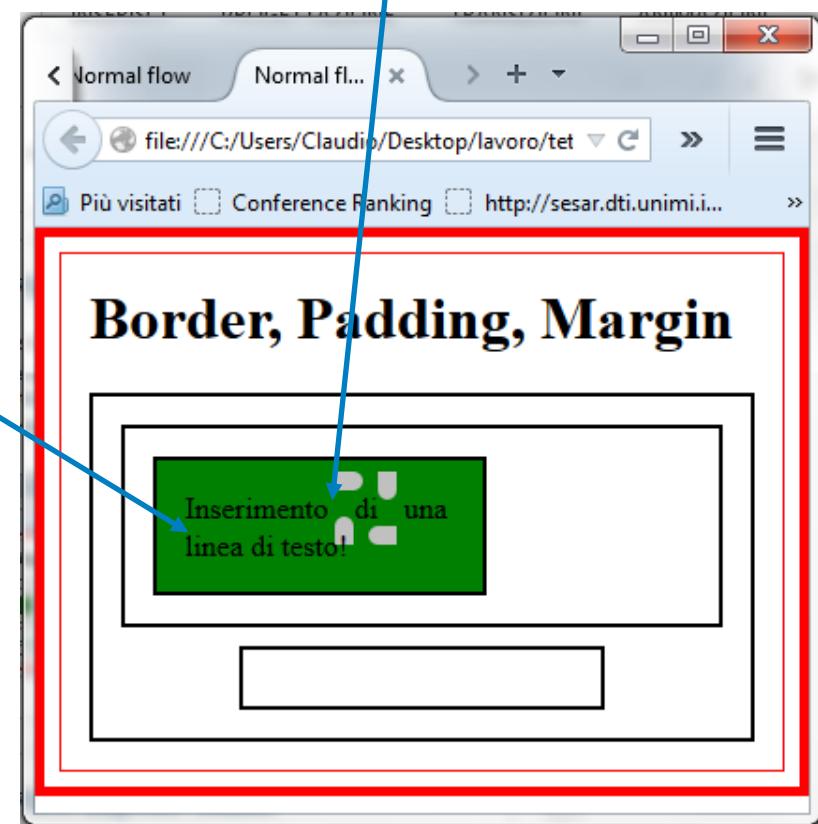
- ▶ Elementi inline vengono mostrati uno dopo l'altro nella direzione di scrittura del testo (ad es., da dx a dx)
 - ▶ Anch'essi hanno box, uno dopo l'altro
 - ▶ Se non c'è abbastanza spazio nel blocco contenitore, si spezzano su una nuova linea: queste linee sono **line boxes**



Normal Flow Layout (Display: inline)

- ▶ Padding/border/margin influenzano la larghezza ma non l'altezza delle box inline

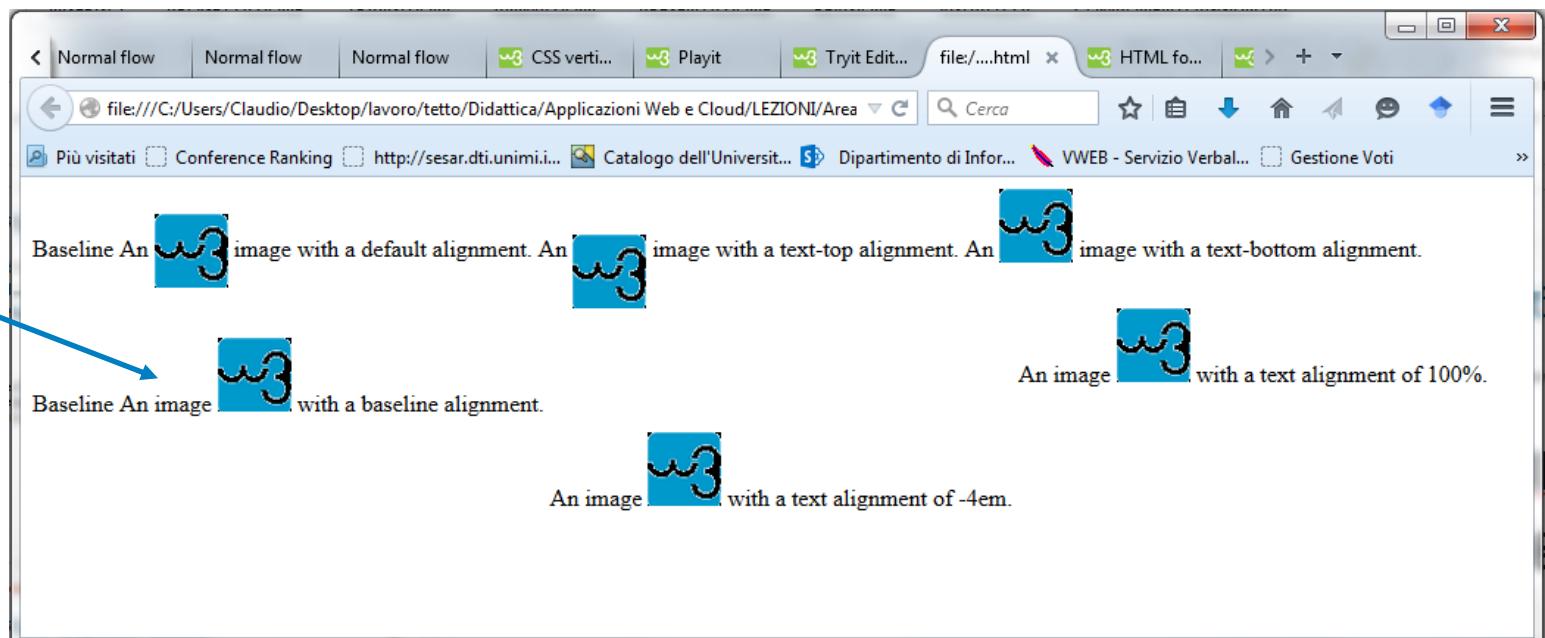
```
<div id="d3" class="shade">  
  Inserimento <span style="border: dotted  
  silver 10px">di</span> una linea di testo!  
</div>
```



Normal Flow Layout (Display: inline)

- ▶ Per posizionare elementi inline all'interno delle line box si usa la proprietà vertical-align

Valore
iniziale del
vertical-
align



Normal Flow Layout (Display: block Vs. inline)

- ▶ Il valore di default di un elemento (i.e., blocco o inline) può essere sovrascritto
- ▶ Ad esempio definito come inline per ottenere un menù orizzontale
- ▶

```
li {  
    display: inline;  
}
```

Normal Flow Layout (Display: none)

- ▶ Valore display none indica che l'elemento non deve essere mostrato
- ▶ L'elemento è nascosto e la pagina visualizzata come se non ci fosse
 - ▶ L'elemento e i suoi discendenti non vengono mostrati
 - ▶ Non influenza il normal flow
- ▶ **display: none** diverso da **visibility: hidden** (gli elementi non vengono visualizzati ma ci sono)
 - ▶ Influenza il normal flow

Proprietà position

- ▶ Specifica il tipo di posizionamento
- ▶ In base al valore della proprietà position, un elemento resta posizionato nel normal flow o ne esce
- ▶ Valori entro il normal flow:
 - ▶ Static
 - ▶ Relative
- ▶ Valori fuori dal normal flow:
 - ▶ Fixed
 - ▶ Absolute



Position: static

- ▶ Position: static
 - ▶ Valore di default
 - ▶ Elemento posizionato in base al normal flow
- ▶

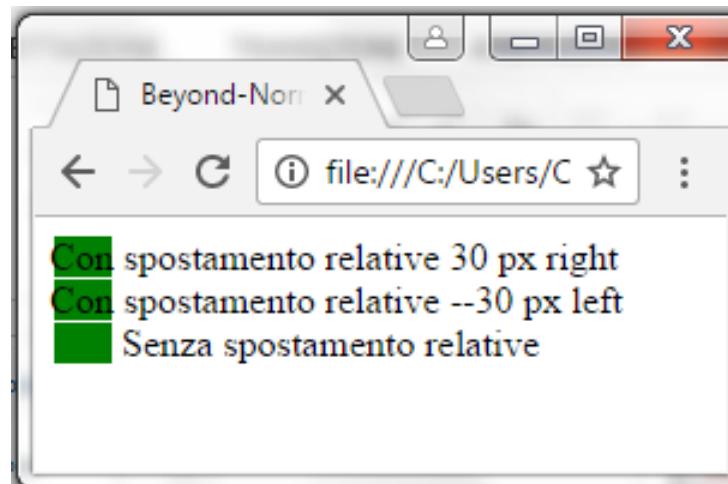
```
div.static {  
    position: static;  
    border: 3px solid #8AC007;  
}
```



Position: relative

- ▶ Position: relative
 - ▶ Elemento posizionato nel suo punto “naturale” del normal flow, spostato di un offset (top, right, bottom, left)
 - ▶ top (bottom): spostamento del top edge (bottom edge) sotto (sopra) il normale
 - ▶ right (left): spostamento dell'edge destro (sinistro) a sinistra (destra) del normale
 - ▶ Offset non impatta nessun altro elemento
- ▶

```
span.right {  
    position: relative;  
    right: 30px;  
}  
  
▶ Stesso effetto con  
left:-30px
```



Position: fixed

- ▶ Position: fixed
 - ▶ Elemento rimosso dal normal flow, (generalmente) posizionato in relazione al blocco contenitore stabilito dal viewport (ad es., la porzione di schermo visualizzata) → elemento nella stessa posizione anche quando la pagina viene scorsa
 - ▶ Spostato di un offset (top, right, bottom, left)
 - ▶ top (bottom): distanza fra il margine esterno del top edge (bottom edge) e il bordo interno del top edge (bottom edge) del blocco contenitore
 - ▶ right (left): distanza fra il margine esterno del right edge (left edge) ed e il bordo interno del right edge (left edge) del blocco contenitore
- ▶

```
div.fixed {  
    position: fixed;  
    bottom: 0;  
    right: 0;  
    width: 300px;  
    border: 3px solid #8AC007;  
}
```
- ▶ 5-noNF-posfixed.html



Position: absolute

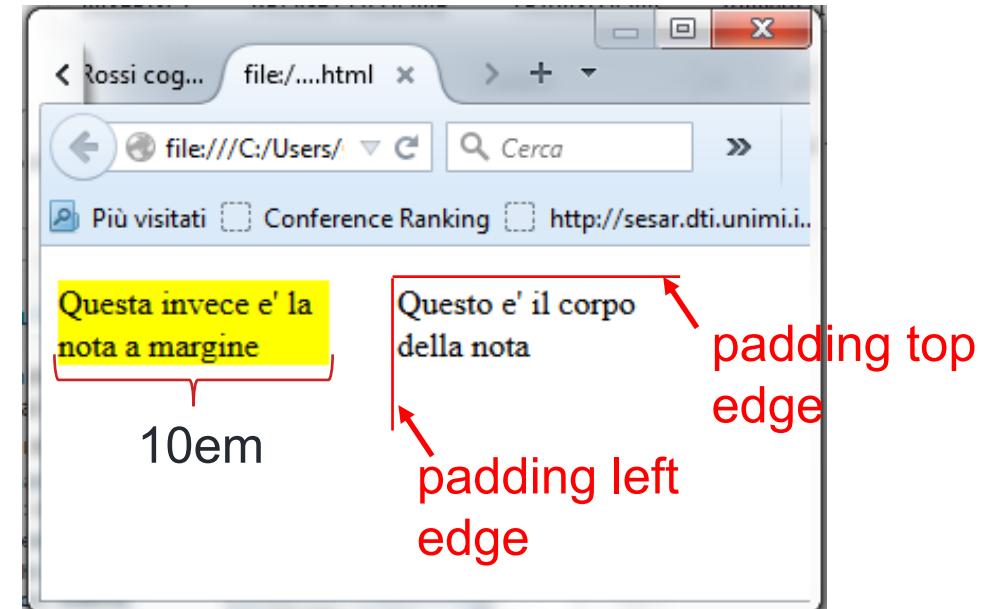
- ▶ Position: absolute
 - ▶ Elemento rimosso dal normal flow, (generalmente) posizionato in relazione al più vicino blocco contenitore antenato posizionato NON static (se non esiste, si considera il body del documento)
 - ▶ Spostato di un offset (top, right, bottom, left)
 - ▶ top (bottom): distanza fra il margine esterno del top edge (bottom edge) e il bordo interno del top edge (bottom edge) del blocco contenitore
 - ▶ right (left): distanza fra il margine esterno del right edge (left edge) ed e il bordo interno del right edge (left edge) del blocco contenitore
 - ▶ [6-noNF-posabsolute.html](#)



Position: absolute

► Absolute positioning

```
p {  
    position: relative;  
    margin-left: 10em;  
    width: 8em;  
}  
  
.marginNote {  
    position: absolute;  
    top: 0;  
    left: -10em;  
    width: 8em;  
    background-color: yellow;  
}  
  
<p>  
    Questo e' il corpo della nota <span class="marginNote">Questa invece e' la nota a  
    margine</span>  
</p>
```



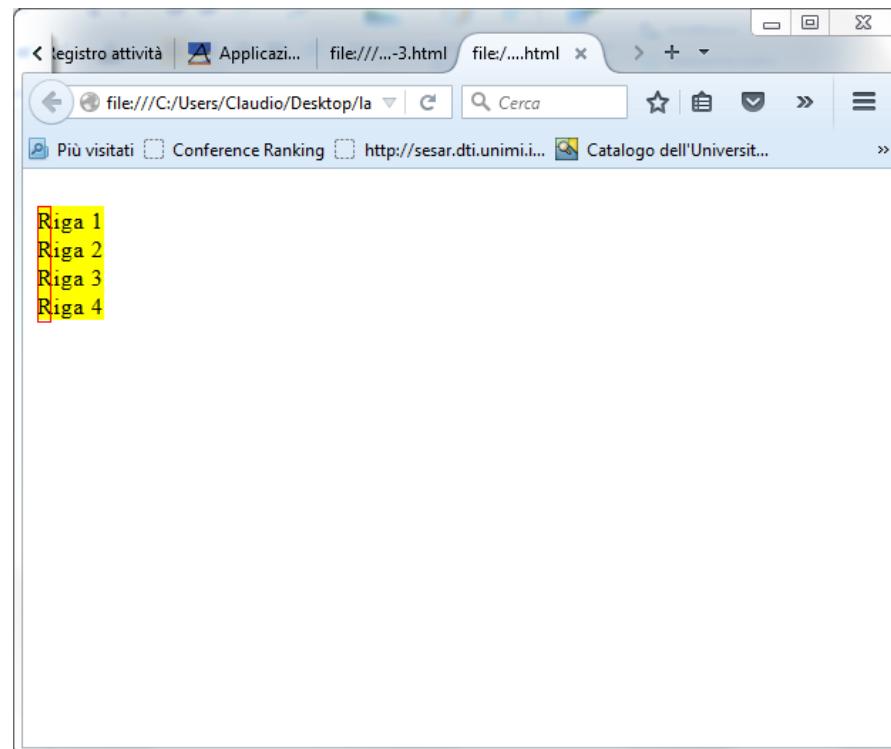
6b-noNF-posabsolute-b.html

Sovrapposizione elementi

- ▶ z-index: specifica la gestione di elementi che si sovrappongono
 - ▶ Elementi con z-index maggiore coprono elementi con z-index minore.

```
#text {  
    position: absolute;  
    top: 10px; left: 10px;  
    background-color: yellow;  
    letter-spacing: 0.1ex;  
    z-index: 1;  
}  
  
#overlay {  
    position: absolute;  
    top: 10px; left: 10px;  
    width: 1.1ex; height: 5em;  
    border: solid red 1px;  
    z-index: 2;  
}
```

7-zindex.html



Proprietà float

- ▶ La proprietà float posiziona un elemento sul lato sinistro o destro del suo contenitore
 - ▶ Ciò permette a testo ed elementi inline di circondarlo
 - ▶ Un elemento float è rimosso dal normal flow, ma resta parte del flow (a differenza del posizionamento absolute)
 - ▶ Float permette anche di creare pagine a colonne
- ▶ Contenuto che circonda l'elemento scorre intorno ad esso sul lato opposto a quello indicato come valore di float
 - ▶ Assume valore left, right, none
 - ▶ Left: elemento a sx del contenitore, testo scorre a dx
 - ▶ Right: l'elemento a dx del contenitore, testo scorre a sx
 - ▶ None: posizionamento normale



Proprietà clear

- ▶ Float sposta un elemento dal flusso normale del documento → è possibile che si trovi in posizioni non desiderate (ad es., al fianco di altri elementi che devono invece essere separati)
- ▶ La proprietà clear impedisce che al fianco di un elemento compaiano altri elementi con float
 - ▶ Si applica solo agli elementi **blocco** e non è ereditata.
- ▶ Specifica su quale lato di un elemento non possono essere visualizzati elementi float
- ▶ Si possono evitare elementi
 - ▶ A sinistra **clear: left**
 - ▶ A destra **clear: right**
 - ▶ A sinistra e destra **clear: both**
- ▶ 9-clear.html



Struttura della pagina

Come definire la struttura della pagina

- ▶ Originariamente veniva usato la struttura tabella di HTML
 - ▶ Tag <table>
 - ▶ Ai tempi era l'unico modo possibile
 - ▶ Spesso si usavano tavelle innestate una nell'altra



Esempio: struttura con tabelle



Esempio: struttura con tabelle

```
<table width="799" border="0"
cellspacing="1" cellpadding="1">
<tr>
<td bgcolor="#000000">
<table width="800" height="485"
border="0">
<tr>
<td height="81" colspan="2"
bgcolor="#CCCCCC">
<table width="100%" border="0">
<tr>
<td bgcolor="#FF9966">&nbsp;</td>
<td bgcolor="#FF9966">&nbsp;</td>
<td bgcolor="#FF9966">&nbsp;</td>
</tr>
</table>
</td>
</tr>
<td width="191" bgcolor="#FFFFFF">&nbsp;</td>
<td width="599" bgcolor="#FFFFFF">&nbsp;</td>
</tr>
</table>
</td>
</tr>
</table>
```

12-struttura-tabelle.html



Come definire la struttura della pagina

- ▶ Layout molto semplice
- ▶ Definire contenuto, navigazione, relazione tra elementi potrebbe richiedere molte linee di codice
- ▶ CSS serve proprio per dividere stile da struttura, cosa che non è stato fatto nell'esempio precedente



Come definire la struttura della pagina

- ▶ Fixed layout vs flexible layout
- ▶ Fixed layout basato su un contenitore con lunghezza fissa
 - ▶ Attributo width dell'elemento contenitore
 - ▶ Ha il pregio di avere sempre lo stesso layout garantito
- ▶ Flexible layout si adatta alla risoluzione e dimensione dello schermo
 - ▶ Se fatto bene si presta alle caratteristiche del web



Struttura della pagina con tag <div>

```
<div id="container">
  <div id="header">
    <div id="navigation"></div><!--#navigation-->
  </div><!--#header-->
  <div id="sidebar"></div><!--#sidebar-->
  <div id="main"></div><!--#main-->
  <div id="footer"></div><!--#footer-->
</div><!--#container-->
```



Struttura della pagina con tag <div>

- ▶ Aggiungiamo dei colori e visualizziamo la struttura della pagina

```
<style type="text/css">  
#header {  
    background-color: #A4A875;  
}  
#navigation {  
    background-color: #DCDCDA;  
}  
#main {  
    background-color: #47834B;  
}  
#sidebar {  
    background-color: #72C29B;  
}  
#footer {  
    background-color: #C4C4C4;  
}  
</style>
```



Struttura della pagina con tag <div>

header

navigation

main

sidebar

footer



Come definire la struttura della pagina: CSS

- ▶ Definiamo lo stile e la posizione per ognuno dei blocchi della pagina
- ▶ Il contenitore della struttura

```
#container {  
    width:1040px;  
    margin:0 auto;  
}
```



Come definire la struttura della pagina: CSS

► Header della pagina

```
#header {  
    width: 1000px;  
    height: 140px;  
    background-color: #A4A875;  
    padding: 10px;  
    margin: 10px;  
    position: relative;  
}
```



Come definire la struttura della pagina: CSS

- ▶ Navigation posizionato all'interno di header

```
#navigation {  
    width: 980px;  
    height: 70px;  
    background-color: #DCDCDA;  
    margin: 10px;  
    position: absolute;  
    bottom: 10px;  
}
```



Come definire la struttura della pagina: CSS

► Main della pagina

```
#main {  
    width: 700px;  
    height: 300px;  
    float: left;  
    background-color: #47834B;  
    margin: 10px;  
}
```



Come definire la struttura della pagina: CSS

- ▶ La barra laterale usando la proprietà float

```
#sidebar {  
    width: 300px;  
    height: 300px;  
    float: right;  
    background-color: #72C29B;  
    margin: 10px;  
}
```



Come definire la struttura della pagina: CSS

► Il footer

```
#footer {  
    clear: both;  
    width: 1000px;  
    height: 70px;  
    background-color: #C4C4C4;  
    padding: 10px;  
    margin: 10px;  
}
```



Struttura della pagina con tag <div>



Conclusioni

- ▶ CSS (Cascading Style Sheet)
- ▶ Uno dei linguaggi fondamentali per lo sviluppo di applicazioni web
- ▶ CSS parallelo a HTML, associa uno stile al testo delle pagine

