Lezione 3: CSS (Base)

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 <font>, 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 (CCS2.1)
    - 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

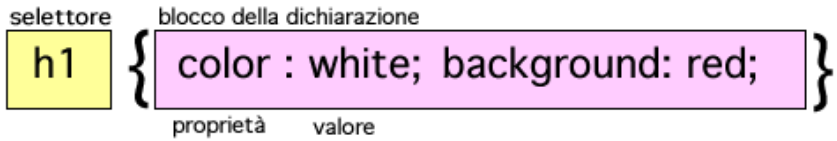
A cosa servono: alcuni esempi

- 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 tabelle, 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
  - Coppie 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;}
- Selettore-semplice.html

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;
      }
- Selettore-semplice-combinazione.html

Selettori (attributi)

- Selettori di attributi
  - [attribute]: seleziona tutti gli elementi con attributo attribute
    - a[target] {
      background-color: yellow;
      }
  - [attribute=value]: seleziona tutti gli elementi con la coppia (attribute,value)
    - a[target=_blank] {
      background-color: yellow;
      }
  - [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;
      }
  - [attribute|=value]: uguale al precedente tranne per il fatto che il valore dell'attributo è una sequenza di parole separate da –
- Selettore-semplice-combinazione-attr.html

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
- 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>
```

- Selettore-semplice.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>

- Selettore-semplice-con-css-integrato.html

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>
- Selettore-semplice-con-css-inlinea.html

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 pagine Web target
  - Se ho 50 pagine web che si devono comportare in maniera omogenea devo definire il CSS interno 50 volte
- CSS esterni lo definisco una volta e riferisco 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 altrimenti perdono di significato

Selettore class

- Selettori che non mappano direttamente a uno specifico tag
  - Ad esempio cambiare il colore di parola all'interno di un tag paragrafo

- In questo caso 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

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
  - ....
  - Il testo all'interno del paragrafo è colorato di rosso

Seconda modalità

- Esiste un secondo tipo di sintassi:
  - <elemento>.nome_della_classe
  - Più restrittivo
  - 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:
  - ...
- ma non questo:
  - ...

Esempio

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

- Selettore-Class-ID.html

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 (anche se non esclusivo)
  - 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
  - p:active, a:active {
    background-color: yellow;
    }
  - a:hover {
    background-color: yellow;
    }
- Selettore-Pseudo-Class.html

Selettore id

- Identificano lo stile di un singolo elemento all'interno della pagina HTML
  - Usato quando si è sicuri che quell'elemento comparirà una sola volta
- 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
  - ....
  - Il testo all'interno del paragrafo è colorato di rosso

- Selettore-Class-ID.html

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;
    }
- Selettore-Pseudo-Elementi.html

Combinazione di class e id

- Si possono combinare ricorsivamente
  - Anche se poi perdono di significato

- #one.two { color: red; }
  - Utilità limitata soprattuto 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 di stile e risoluzione dei conflitti

Ereditarietà di stile e risoluzione dei conflitti

- Ereditarietà
  - Un elemento contenuto in un altro (child) eredita tutte le proprietà di stile dal suo elemento padre
  - Tuttavia, se il figlio (o discendente) ha proprietà in conflitto con le proprietà definite dall'elemento padre, il conflitto viene risolto in favore dell'elemento figlio
  - Proprietà più specifica vince

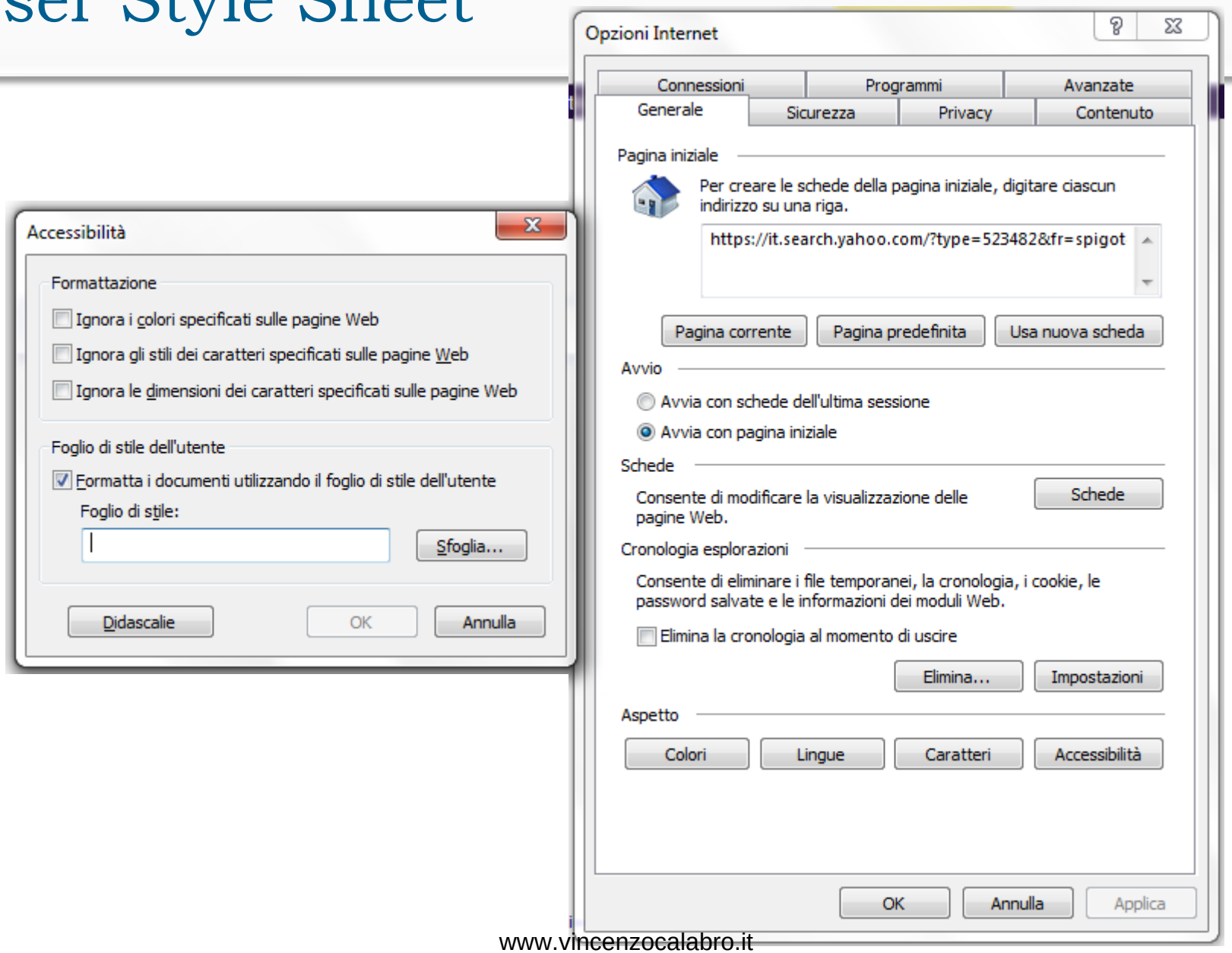
Ereditarietà di stile e risoluzione dei conflitti

- 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;
    }
    ```
    ```

User Style Sheet



Regole CSS: Valutazione a cascata

- User agent deve applicare un algoritmo per stabilire i valori di una combinazione di elementi e proprietà

1. Per prima cosa bisogna selezionare tutte le dichiarazioni che applicano a un elemento/proprietà per il media type richiesto
   - Le dichiarazioni si applicano se il selettore è consistente con l'elemento considerato
   - Media type definisce lo stile in base al tipo di media scelto

```
@media screen {
    p {
        font-family: verdana, sans-serif;
        font-size: 17px;
    }
}
```

```
@media screen and (min-width: 480px) {
    body
        background-color: lightgreen;
    }
}
```

http://www.w3schools.com/css/tryit.asp?filename=trycss3_media_queries1

Regole CSS: Valutazione a cascata

- User agent deve applicare un algoritmo per stabilire i valori di una combinazione di elementi e proprietà

2. Il primo ordinamento è fatto in base a peso e origine
   - Per dichiarazioni normali, gli style sheet dell'autore della pagina sovrascrivono gli style sheet dell'utente che sovrascrivono quelli di default
   - Per dichiarazioni "importanti" (!important), gli style sheet dell'utente della pagina sovrascrivono gli style sheet dell'autore che sovrascrivono quelli di default
     - #example { font-size: 14px !important; }
   - Uno style sheet importato ha la stessa origine dello style sheet che lo importa

Five origin/importance levels:
1. user/important
2. author/important
3. author/normal
4. user/normal
5. user agent/normal

Regole CSS: Valutazione a cascata

- User agent deve applicare un algoritmo per stabilire i valori di una combinazione di elementi e proprietà

3. Il secondo ordinamento è fatto in base alla specificity
   - Selettori più specifici sovrascrivono quelli generali
   - Pseudo-elementi e pseudo-classi sono considerati come elementi e classi normali

Specificity:
1. `style` attribute
2. rule with selector:
   1. ID
   2. class/pseudo-class
   3. descendant/element type
   4. universal (*)
3. HTML attribute (ad esempio attribute align)

Regole CSS: Valutazione a cascata

- User agent deve applicare un algoritmo per stabilire i valori di una combinazione di elementi e proprietà

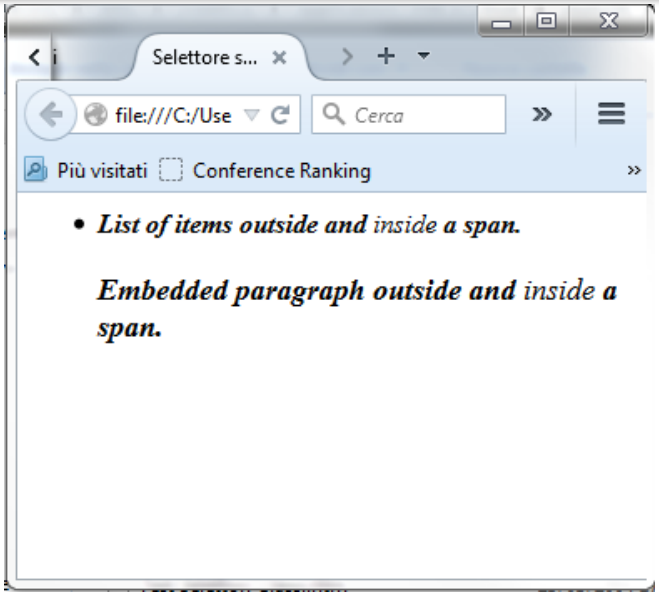
4. Il terzo ordinamento è fatto in base all'ordine
   - Se due regole hanno stesso peso, origine e specificity, vince l'ultimo specificato
   - Regole in style sheet importati sono considerate prima di quelle regole direttamente nello style sheet
   - Concettualmente c'è uno style sheet unico e regole definite dopo hanno maggiore priorità

Ereditarietà

- Cosa succede se nessuna dichiarazione applica a una proprietà di un elemento?
- Generalmente, valore ereditati dall'elemento antenato più vicino che ha un valore per quella proprietà
- Se nessun antenato ha un valore (o la proprietà non supporta ereditarietà), il CSS definisce un valore iniziale che viene usato

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

- Valore delle proprietà
  - Specified: valore contenuto nella dichiarazione
    - Absolute: valore determinate senza riferimento al contesto (ad es., 2cm)
    - Relative: valore dipende dal contesto (ad es., larger)
  - Computed: rappresentazione assoluta di un valore relativo (ad es., larger potrebbe essere 1.2 x del font size del padre)
  - Actual: valore usato dal browser (ad es., il valore computed deve essere arrotondato)

- Molte proprietà ereditano un valore computed
  - Eccezione: line-height (lo vediamo dopo)

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
- 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
  - Lenght
  - Percentage
  - Relative size
- Importante perché monitor hanno risoluzione diversa, la dimensione può essere modificata manualmente e visualizzazione fatta attraverso mobile device

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
- Definisce la dimensione come la dimensione di default del browser

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

Font-size: relative-size

- Un insieme di keyword interpretate come relative alla dimensione del font dell'oggetto padre
- Possibili valori includono larger e smaller
- Font.html

Proprietà aggiuntive

Property	Possible values
`font-style`	`normal` (initial value), `italic` (more cursive than normal), or `oblique` (more slanted than normal).
`font-weight`	`bold` or `normal` (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).
`font-variant`	`small-caps`, which displays lowercase characters using uppercase glyphs (small uppercase glyphs if possible), or `normal` (initial value)

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 (rbg(255,0,0)) o hex (#ff0000)

CSS box model: Border

- Border-* può avere da uno a 4 valori
- **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;
}
```

- Border.html

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 i 4 possibili valori dell'attributo border

CSS box model: Margin

- Esempi

```
p {
    margin-top: 100px;
    margin-bottom: 100px;
    margin-right: 150px;
    margin-left: 50px;
}

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

- Margin.html

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 i 4 possibili 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;
}
```

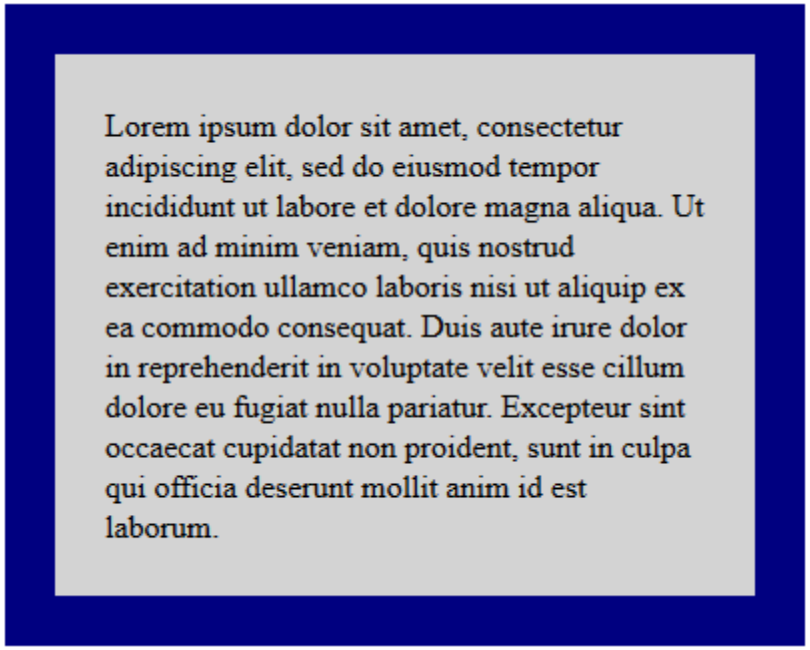
- Padding.html

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>
```

- CSS-box-model.html

CSS box model: Esempio



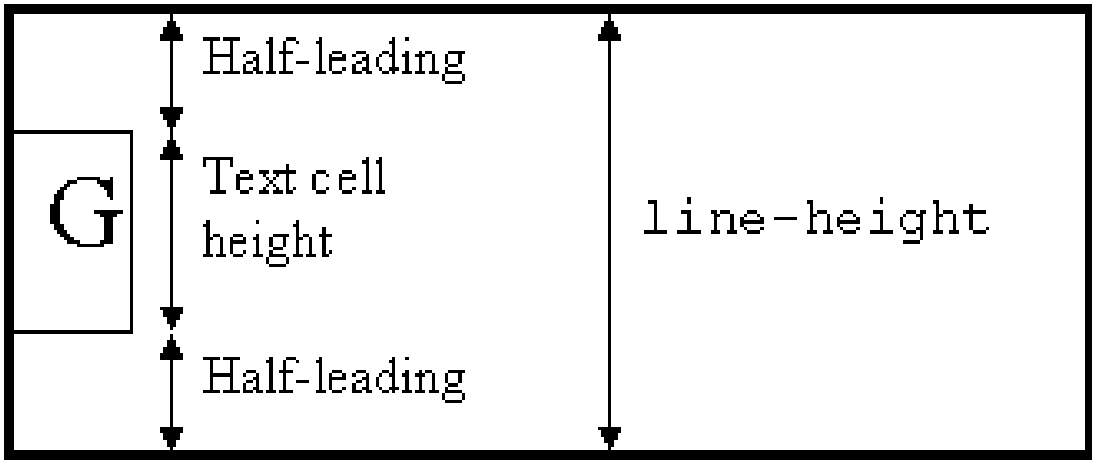
Line-height

- 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

- Scorciatoia 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
    …

- Unordered list
  -
    list 1
    list 2
    …

- Definition list
  - <dl>
    <dt>list 1</dt>
    <dd>list 2</dd>
    …
    </dl>

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;
}
```

- Liste.html

CSS Custom Properties (Variabili CSS)

CSS Custom Properties (Variabili CSS)

- Siti web complessi contengono quantità molto grandi di CSS, spesso con molti valori ripetuti
  - Ad esempio, lo stesso colore potrebbe essere utilizzato in centinaia di luoghi diversi, richiedendo una ricerca globale e la sostituzione se è necessario modificare il colore

- CSS Custom Properties (chiamate anche variabili CSS o custom variable) consentono di archiviare e riutilizzare un valore in un documento
  - Sono definite dagli autori CSS
  - Vengono impostati utilizzando la notazione --nome: valore (ad esempio, --main-color: black;) e sono accessibili utilizzando la funzione var() (ad esempio, color: var(--main-color);).
  - Permettono di definire identificatori semantici. Ad esempio, --main-text-color è più facile da capire di #00ff00, soprattutto se lo stesso colore viene utilizzato anche in altri contesti.

Funzione var()

- La funzione var() viene utilizzata per inserire il valore di una variabile CSS
  - Sintassi: var(--name, value)
    - name: il nome della variabile che inizia con -- ed è case sensitive (required)
    - value: il valore di scorta nel caso in cui la variabile non sia trovata

- I vantaggi dell'utilizzo di var() sono
  - Codice più leggibile
  - Più semplice modificare i valori del colore

Funzione var()

- Le variabili CSS non funzionano all'interno delle media query e container query
- La funzione var() può essere utilizzata al posto di qualsiasi valore in qualsiasi proprietà di un elemento
- La funzione var() non può essere utilizzata come nome di proprietà, selettore o altra in aggiunta ai valori delle proprietà
- Variabili CSS sono soggette a ereditarietà

Funzione var()

- Le variabili CSS hanno accesso al DOM, il che significa che è possibile creare variabili con ambito **locale** o **globale**, modificare le variabili con JavaScript e modificare le variabili in base alle media query
  - Variabili globali possono essere utilizzate attraverso l'intero documento
  - Variabili locali possono essere utilizzate solo all'interno del selettore in cui sono dichiarate

- Una variabile con ambito globale è dichiarata all'interno del selettore :root (elemento radice del documento – pseudoclass)

- Una variabile con ambito locale è dichiarata all'interno del selettore che la utilizzerà

Esempio 1

https://www.w3schools.com/css/css3_variables.asp

- Versione tradizionale (no variabili CSS)

```css
body {
  background-color: #1e90ff;
}

h2 {
  border-bottom: 2px solid #1e90ff;
}

.container {
  color: #1e90ff;
  background-color: #ffffff;
  padding: 15px;
}

button {
  background-color: #ffffff;
  color: #1e90ff;
  border: 1px solid #1e90ff;
  padding: 5px;
}
```

Esempio 1

- Versione con variabili CSS: due variabili globali (--blue e --white)

scope names values

```
:root {
  --blue: #1e90ff;
  --white: #ffffff;
}

body {
  background-color: var(--blue);
}

h2 {
  border-bottom: 2px solid var(--blue);
}

.container {
  color: var(--blue);
  background-color: var(--white);
  padding: 15px;
}

button {
  background-color: var(--white);
  color: var(--blue);
  border: 1px solid var(--blue);
  padding: 5px;
}
```

Esempio 1

- Versione con variabili CSS: modifica variabili globali (--blue e --white)

```
:root {
  --blue: #6495ed;
  --white: #faf0e6;
}

body {
  background-color: var(--blue);
}

h2 {
  border-bottom: 2px solid var(--blue);
}

.container {
  color: var(--blue);
  background-color: var(--white);
  padding: 15px;
}

button {
  background-color: var(--white);
  color: var(--blue);
  border: 1px solid var(--blue);
  padding: 5px;
}
```

new values

Esempio 2

https://developer.mozilla.org/en-US/docs/Web/CSS/Using_CSS_custom_properties

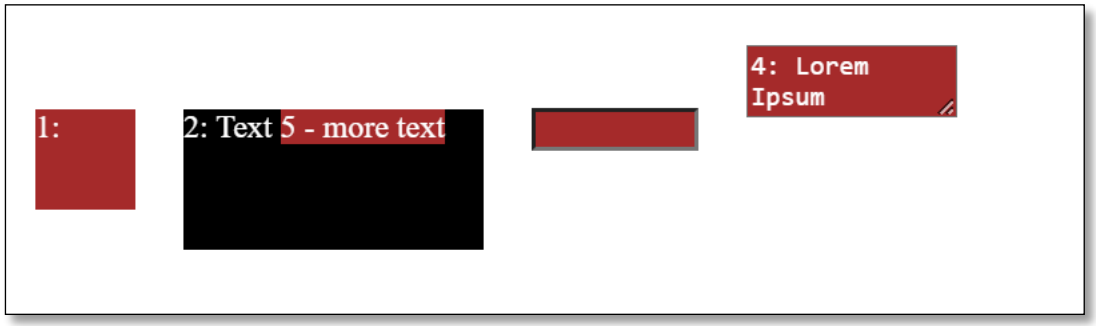
- Versione tradizionale (no variabili CSS)

```css
.one {
  color: white;
  background-color: brown;
  margin: 10px;
  width: 50px;
  height: 50px;
  display: inline-block;
}

.two {
  color: white;
  background-color: black;
  margin: 10px;
  width: 150px;
  height: 70px;
  display: inline-block;
}
.three {
  color: white;
  background-color: brown;
  margin: 10px;
  width: 75px;
}
.four {
  color: white;
  background-color: brown;
  margin: 10px;
  width: 100px;
}

.five {
  background-color: brown;
}
```

```html
<div>
  <div class="one">1:</div>
  <div class="two">2: Text <span class="five">5 - more text</span></div>
  <input class="three" />
  <textarea class="four">4: Lorem Ipsum</textarea>
</div>
```



Esempio 2

- Versione con variabile CSS e pseudo class :root

```
:root {
  --main-bg-color: brown;
}

.one {
  color: white;
  background-color: var(--main-bg-color);
  margin: 10px;
  width: 50px;
  height: 50px;
  display: inline-block;
}

.two {
  color: white;
  background-color: black;
  margin: 10px;
  width: 150px;
  height: 70px;
  display: inline-block;
}
.three {
  color: white;
  background-color: var(--main-bg-color);
  margin: 10px;
  width: 75px;
}
.four {
  color: white;
  background-color: var(--main-bg-color);
  margin: 10px;
  width: 100px;
}

.five {
  background-color: var(--main-bg-color);
}
```

Ereditarietà

- Le variabili CSS vengono ereditate
- Se non viene impostato alcun valore per una variabile CSS su un dato elemento, viene utilizzato il valore del suo genitore

```
<div class="one">
  <div class="two">
    <div class="three"></div>
    <div class="four"></div>
  </div>
</div>
```

```
.two {
  --test: 10px;
}

.three {
  --test: 2em;
}
```

- In questo caso, i risultati di var(--test) sono:
  - Per l'elemento class="two": 10px
  - Per l'elemento class="three": 2em
  - Per l'elemento class="four": 10px (ereditato dal suo genitore)
  - Per l'elemento class="one": valore non valido, che è il valore predefinito di qualsiasi proprietà personalizzata

Flow processing

Normal Flow Layout

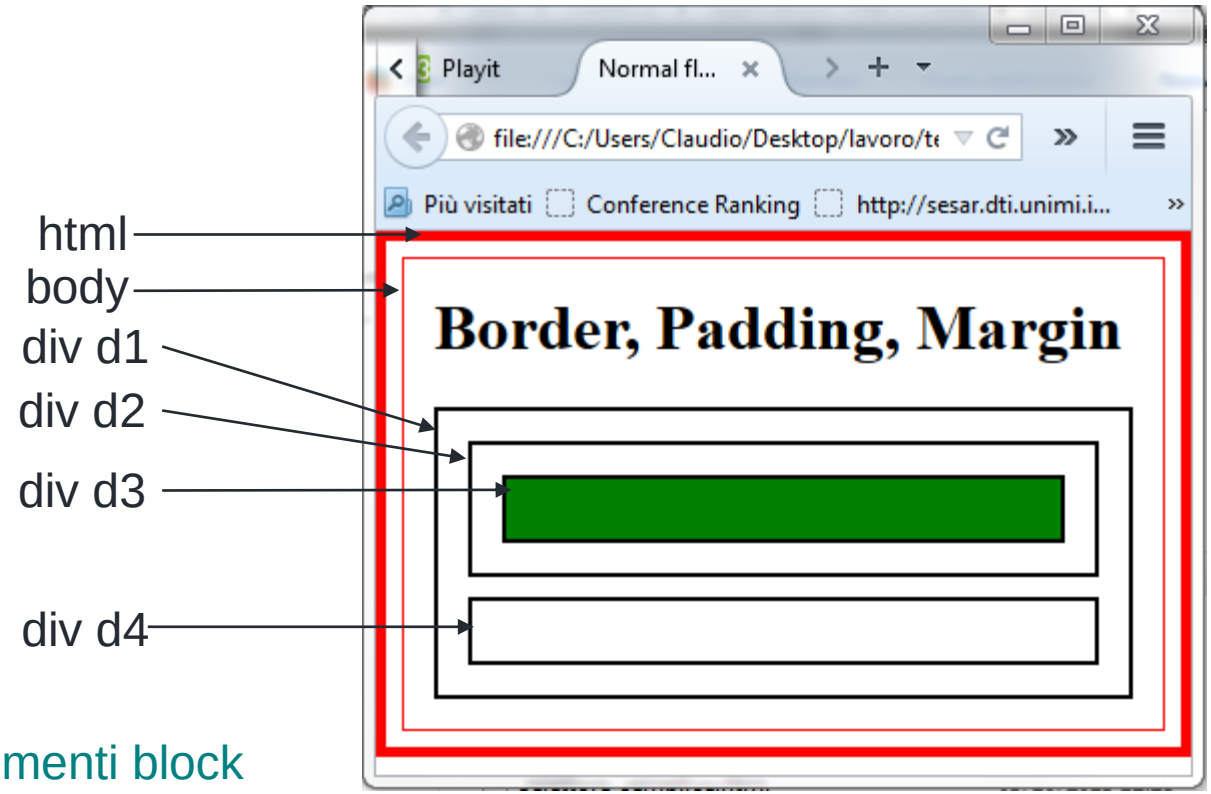
- In normal flow processing, ogni elemento ha una box corrispondente
  - L'elemento HTML si riferisce a una box chiamata initial containing block
    - Corrisponde all'intero documento
  - Box degli elementi figli sono contenuti in box degli elementi padri
  - Elementi block: fratelli sono messi uno 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>
```

Normal Flow Layout



Gli elementi block compaiono in ordine rispetto all'ordinamento nel document HTML
Normal-flow-1.html

Normal Flow Layout: Display

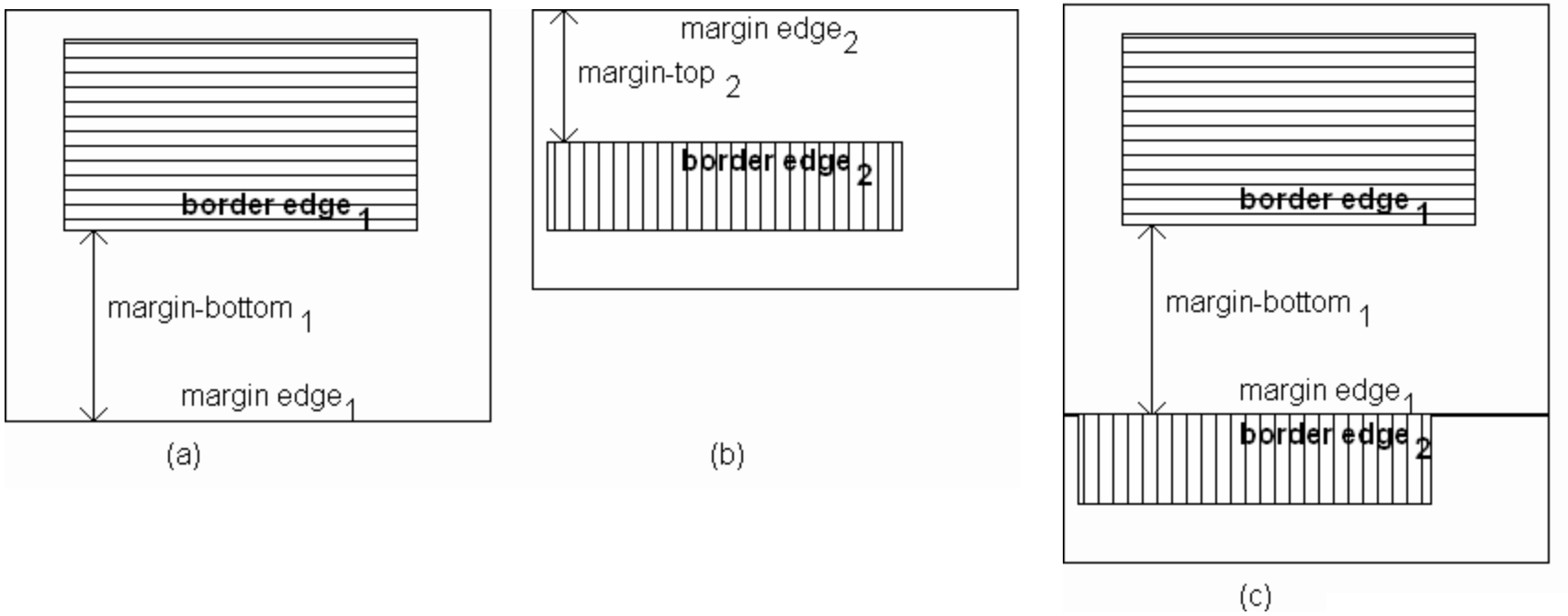
- Proprietà display
  - Controlla il layout e dice se e come un elemento deve essere mostrato
  - Valore di default block o inline

Normal Flow Layout: 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 *li* e quelli relativi alle tabelle hanno la proprietà display con valore inline

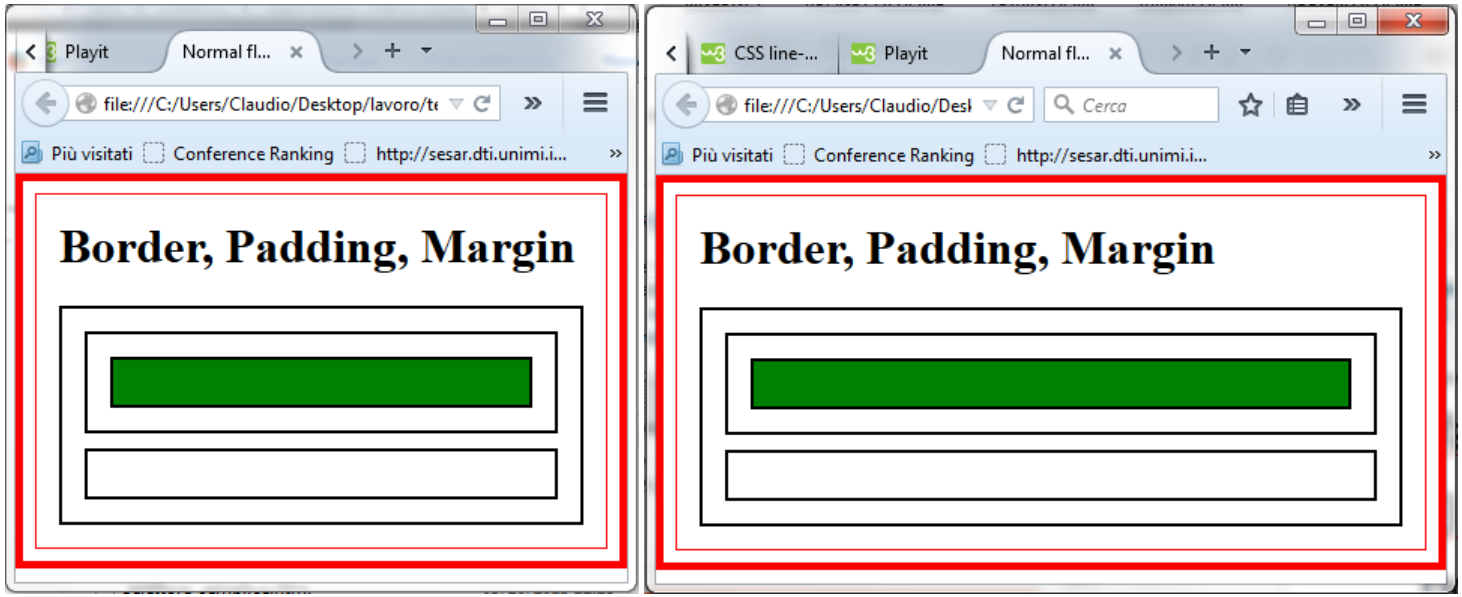
Normal Flow Layout: Block

- 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



Normal Flow Layout: Block

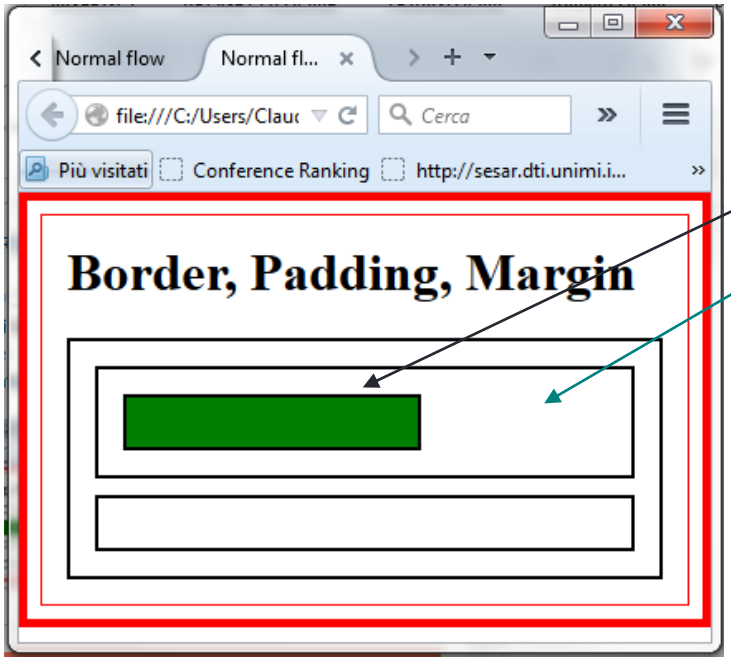
- Valore iniziale della proprietà width è auto
- Elementi block occupano l'area di contenuto più larga possibile
  - Tutto fatto rispettando margini e padding



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

Normal Flow Layout: Block

- Possibile specificare la lunghezza usando CSS width
  - Possibile specificare la lunghezza come percentuale del contenuto dell'elemento padre
  - Normal-flow-2.html

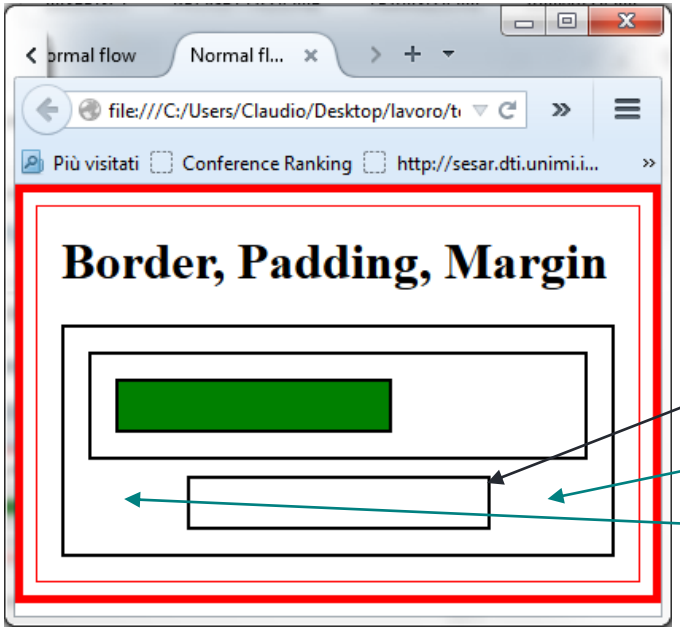


#d3 {width: 50%}

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

Normal Flow Layout: Block

- Possibile specificare la lunghezza usando CSS width
  - Possibile specificare la lunghezza come percentuale del contenuto dell'elemento padre
  - Normal-flow-3.html

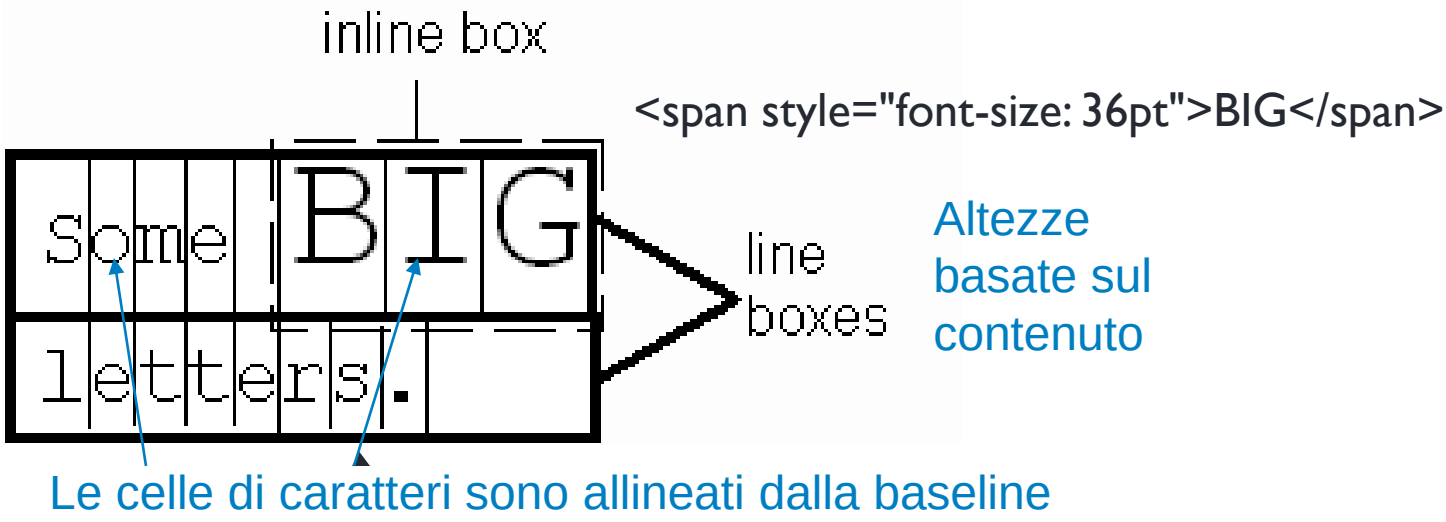


#d4 {width: 50%; margin-left: auto; margin-right: auto}

Possibile centrare l'elemento mettendo entrambi i margini ad auto

Normal Flow Layout: Inline

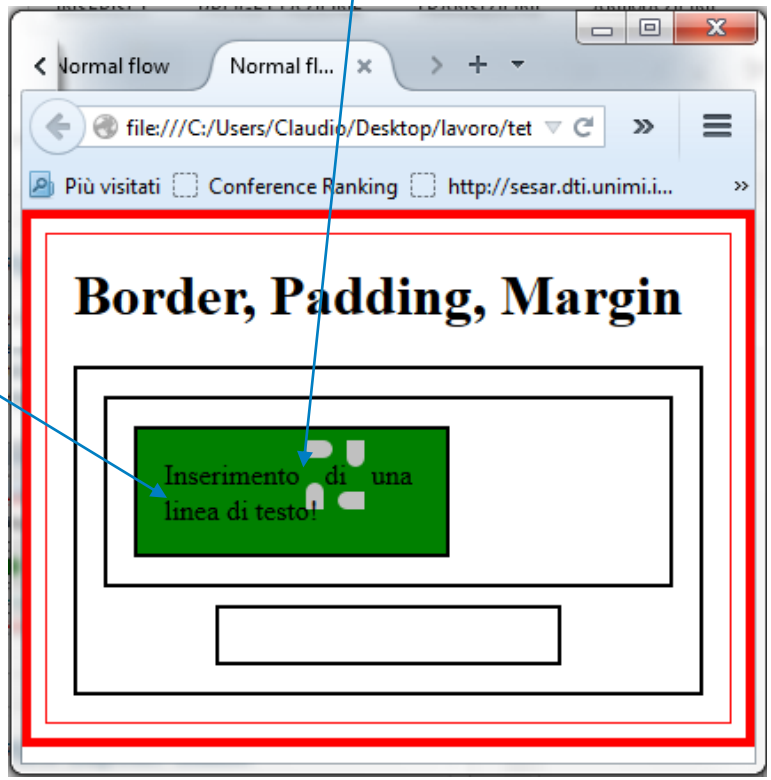
- Box che corrispondono a celle di caratteri o elementi inline (display: inline) sono messe uno in fianco alle altre in line box
- Elementi inline iniziano sempre sulla stessa riga e prendono la larghezza necessaria
- Line box sono messe una sopra le altre



Normal Flow Layout: Inline

- Padding/border/margin influenzano la larghezza ma non l'altezza delle box inline
  - Normal-flow-4.html

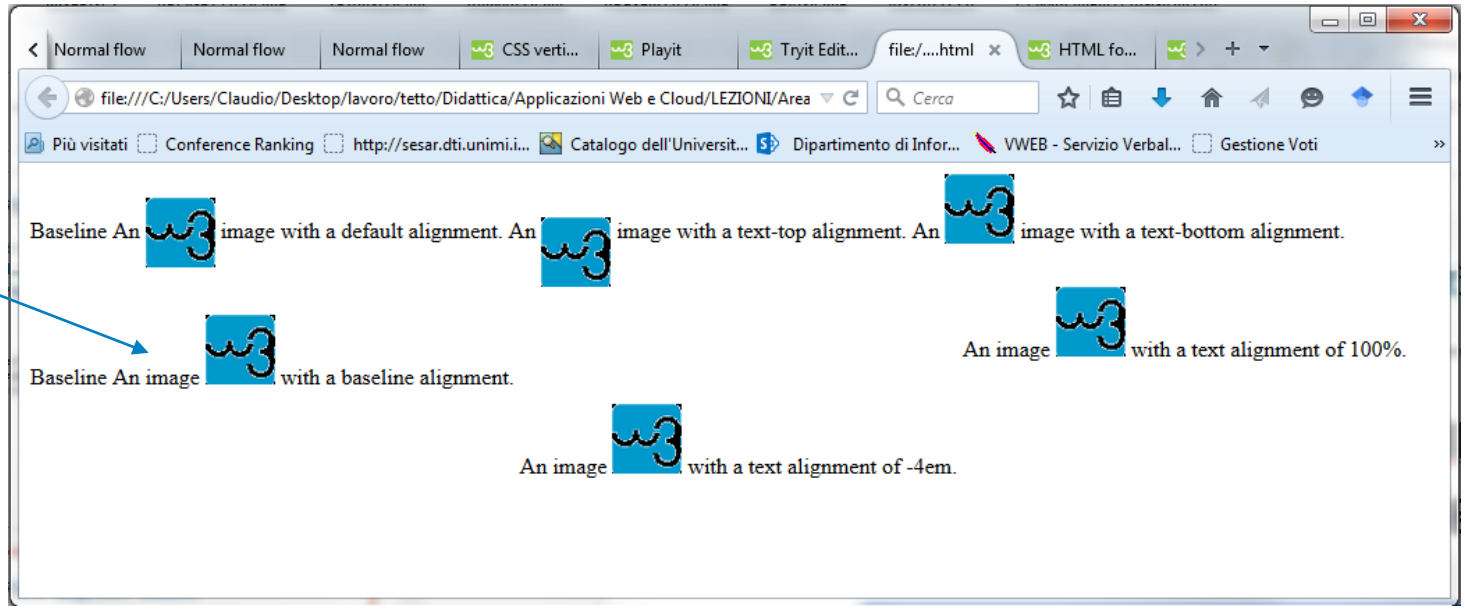
```
<div id="d3" class="shade">
   Inserimento <span style="border: dotted
   silver 10px">di</span> una linea di testo!
</div>
```



Normal Flow Layout: Inline

- Per posizionare elementi inline all'interno delle line box si usa la proprietà vertical-align
  - Normal-flow-5.html

Valore iniziale del vertical-align



Normal Flow Layout: None

- Il valore di default di un elemento può essere sovrascritto
- Ad esempio <li> definito come inline per ottenere un menù orizzontale
- li {
  display: inline;
  }

Normal Flow Layout: 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

- Visibility: significa che gli elementi non vengono visualizzati ma ci sono
  - Influenza il normal flow

Oltre il normal flow: Proprietà position

- Specifica il tipo di posizionamento
  - Static
  - Relative
  - Fixed
  - Absolute

Oltre il normal flow: Proprietà position

- Proprietà top, bottom, right, left hanno risultati diversi in base al valore di position

- Position: static
  - Valore di default
  - Elemento posizionato in base al normal flow
  - L'elemento div ha position static

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

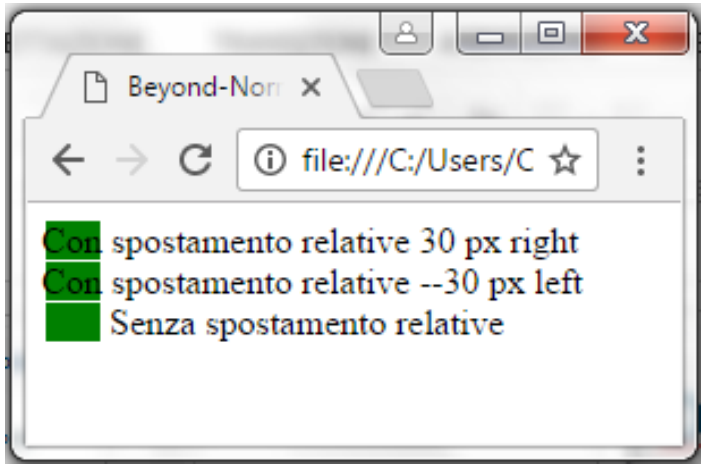
Oltre il normal flow: Proprietà position

- Position: fixed
  - Elemento posizionato nello stesso punto anche quando la pagina viene scorsa
  - Proprietà top, bottom, left, right usate per posizionare l'elemento

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

Oltre il normal flow

- CSS permette agli elementi di essere posizionati al di fuori del normal flow
- Position: relative (Beyond-Normal-flow-1.html)
  - Posizionamento relativo alla posizione tradizionale (quella con position: static)
  - Nessun elemento verrà posizionato per riempire il gap lasciato
  - span.right {
    position: relative;
    right: 30px;
    }
  - Stesso effetto con left:-30px



Oltre il normal flow

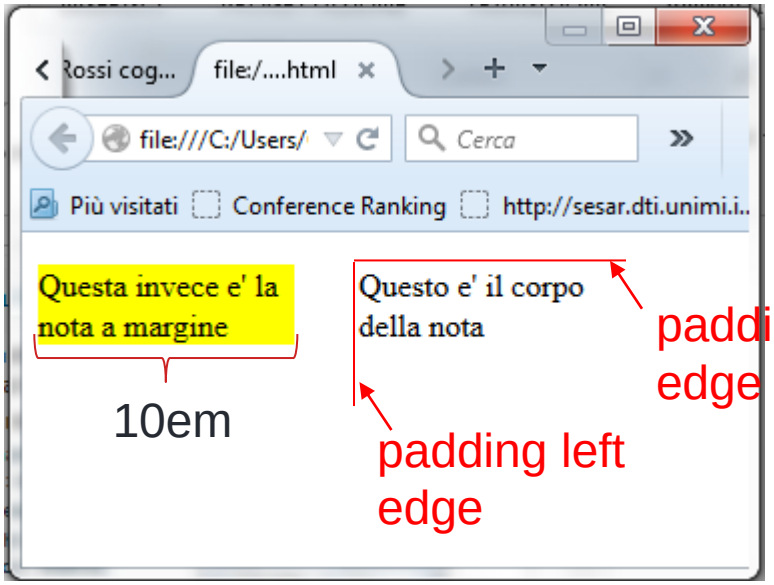
- CSS permette agli elementi di essere posizionati al di fuori del normal flow

- Position: absolute
  - Posizionamento box relativo alla posizione dell'elemento antenato (quello con position: relative)
  - Se non esistono antenati si considera il body e l'elemento mantiene la posizione quando la pagina viene scorsa

Oltre il normal flow

- Absolute positioning
  - Beyond-Normal-flow-2.html

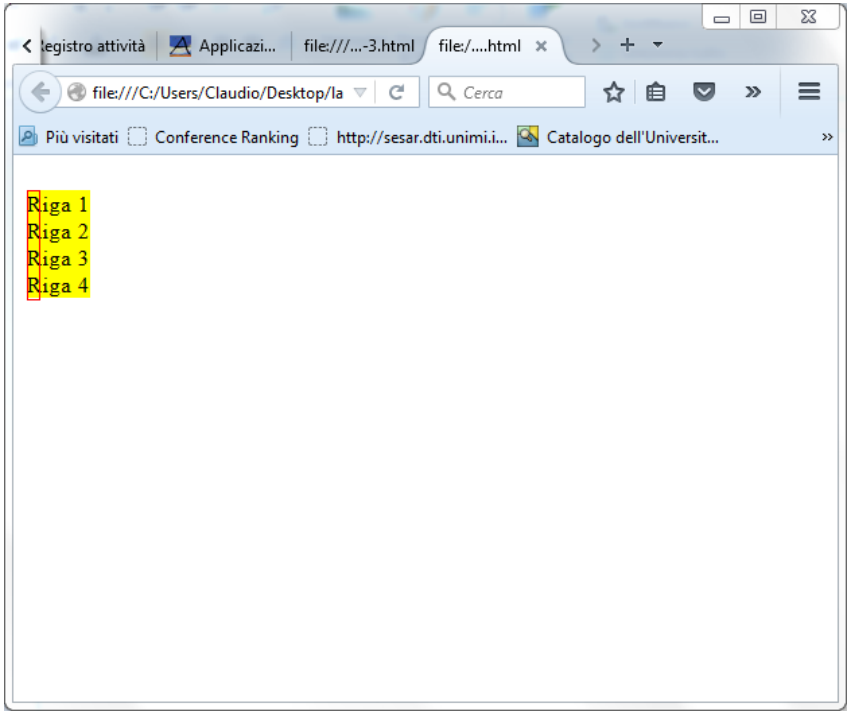
```
p {
        position: relative;
        margin-left: 10em;
        width: 8em;
}
.marginNote {
        position: absolute;
        top: 0;
        left: -10em;
        width: 8em;
        background-color: yellow;
}
<p>
   Questo e' il corpo della nota <span class="marginNote">Questa invece e' la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



Sovrapposizione elementi

- z-index: specifica la gestione di elementi che si sovrappongono
  - Chi sta sopra, chi sta sotto
  - Si possono definire valori positivi e negative
- Beyond-Normal-flow-3.html

```
#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;
}
```



Proprietà float

- Permette di posizionare testo intorno a un'immagine
  - Text wrap o runaround
- CSS ottiene questo effetto permettendo a elementi che seguono un elemento float in HTML di circondare l'elemento, cambiando la loro posizione
- Float permette anche di creare pagine a colonne
  - Assume valore left, right, none

Proprietà float

- Quando un elemento è annotato con float è tolto dal normale flusso HTML
- Quindi spesso possono esserci errori di visualizzazione se non si sta attenti
- Una soluzione è quello di forzare i due elementi parenti uno a sinistra e l'altro a destra
  - Nessuna dipendenza dall'ordine dei tag HTML nel body della pagine
  - Come alternative
    - Forzarli entrambi a sinistra in catena (stesso risultato)
    - Indicare il primo elemento con float:right (oppure float:left)

Proprietà clear

- Se si vuole forzare un elemento a stare da solo senza elementi a lato si usa la proprietà clear
- 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

Esempio

- Ritorniamo alla struttura della pagina della lezione precedente e consideriamo gli elementi in rosso

```
<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-->
```

Esempio: main, sidebar, footer

- Usiamo le proprietà float e clear per posizionare il main e la sidebar sulla stessa linea e il footer sotto di loro

- Partiamo da un esempio in cui spostiamo a destra il nostro sidebar usando la proprietà float
  - Quando la proprietà float:right è associato a sidebar, quest'ultimo esce dal flow normale

Esempio: main, sidebar, footer

```
#main {
        background-color: #47834B;
        margin: 10px;
}
#sidebar {
        width: 300px;
        float: right;
        background-color: #72C29B;
        margin: 0px 10px;
}
#footer {
        width: 1000px;
        height: 70px;
        background-color: #C4C4C4;
        padding: 10px;
        margin: 10px;
}
```

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

- div-con-css-errore-1.html

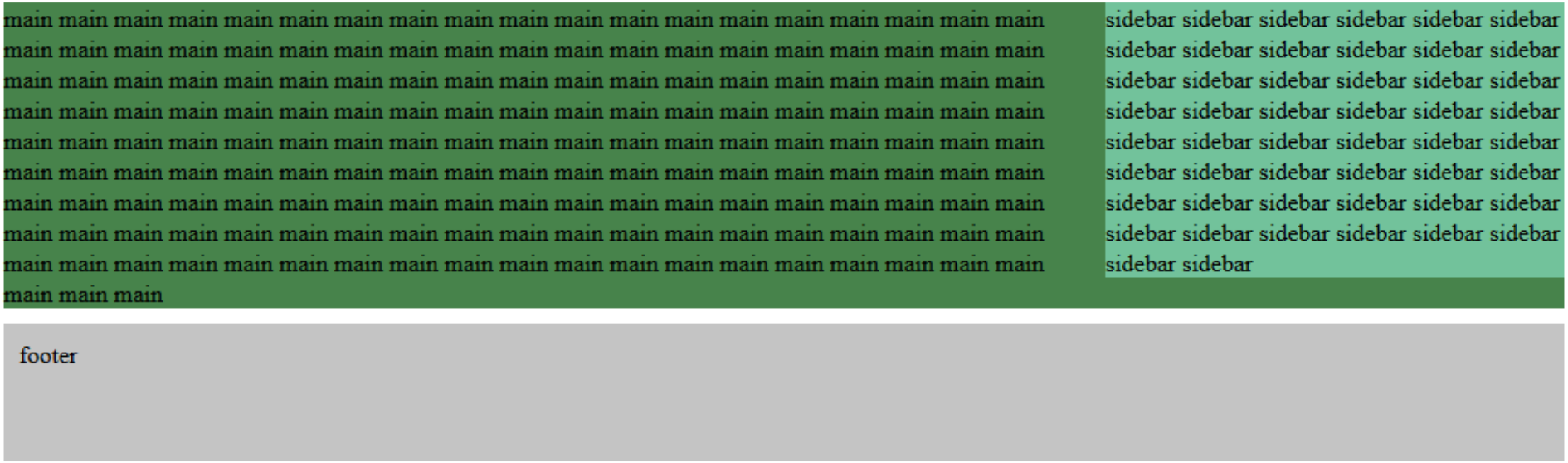
Esempio: main, sidebar, footer

- Siccome il sidebar contiene molto testo ingloba sia main che footer

- Il layout dipende
  - Dal contenuto testo degli elementi
  - Dalla dimensione della finestra: elementi hanno dimensione fissa e non vengono ridimensionati al ridimensionarsi della finestra

- Se aumentiamo il contenuto di main in modo da eccedere quello di sidebar, cosa succede?

Esempio: main, sidebar, footer


main main main

Esempio: main, sidebar, footer

- Aggiungiamo float: left allo stile CSS dell'elemento main e la dimensione dell'elemento width: 300px (div-con-css-errore-3.html)

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

Esempio: main, sidebar, footer

- Infine andiamo a forzare il footer alla fine della pagina con la proprietà clear

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

Esempio: main, sidebar, footer

- Facciamo anche qualche cambio stilistico a main e sidebar giocando con width e height

```
#sidebar {
        width: 300px;
        height: 300px;
        float: right;
        background-color: #72C29B;
        margin: 0px 10px;
}
```

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

div-a-3-con-css.html

Oltre il normal flow: Inline-block

- Originariamente griglie di elementi erano realizzate con i float
  - Supporto per resize automatico rispetto alla dimensione del browser
- Display: inline-block
  - Come elemento inline ma con height e width
- ```
  .floating-box {
      display: inline-block;
      width: 150px;
      height: 75px;
      margin: 10px;
      border: 3px solid #8AC007;
  }
  ```
- inline-block.html
  ```

Oltre il normal flow: Proprietà Grid

- Display: grid oppure Display: inline-grid
  - Definisce un elemento come un contenitore griglia in modalità block

- Proprietà grid raffina l'elemento contenitore con
  - grid-template-rows
  - grid-template-columns
  - grid-template-areas
  - grid-auto-rows
  - grid-auto-columns
  - grid-auto-flow

- grid.html

```
.grid-container {
  display: grid;
  grid: 150px / auto auto auto;
  grid-gap: 10px;
  background-color: #2196F3;
  padding: 10px;
}
```

```
<div class="grid-container">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
  <div class="item5">5</div>
  <div class="item6">6</div>
</div>
```

https://www.w3schools.com/cssref/pr_grid.php

Proprietà Grid: Struttura Pagina Web

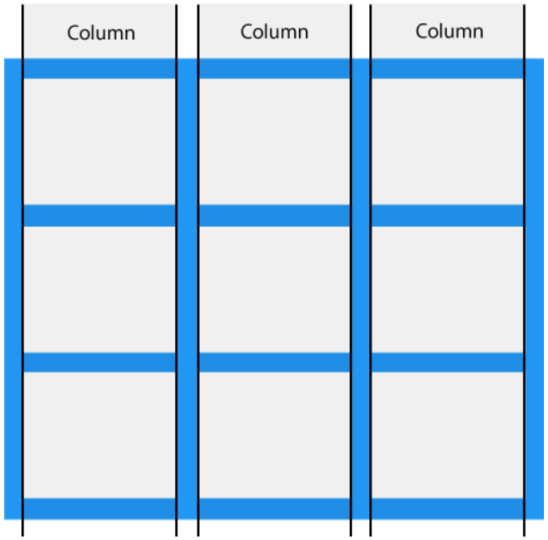
- Il modulo CSS Grid Layout offre un sistema di layout basato su griglia, con righe e colonne, che semplifica la progettazione di pagine Web senza dover utilizzare float e posizionamento

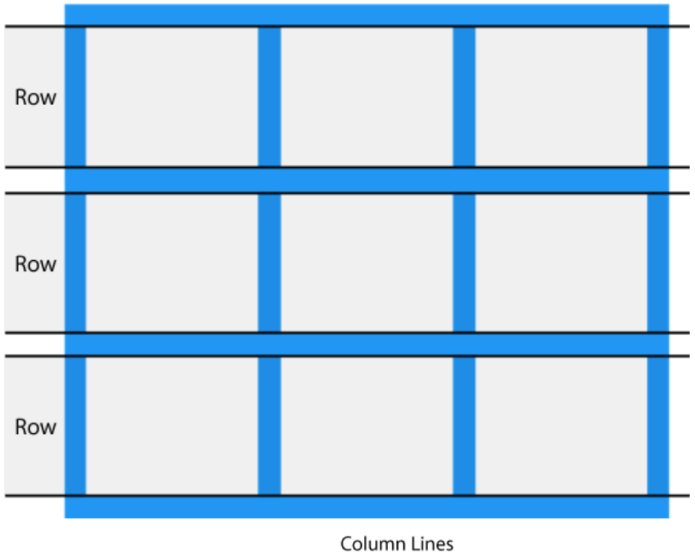
```css
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto;
  background-color: #2196F3;
  padding: 10px;
}
.grid-item {
  background-color: rgba(255, 255, 255, 0.8);
  border: 1px solid rgba(0, 0, 0, 0.8);
  padding: 20px;
  font-size: 30px;
  text-align: center;
}
```

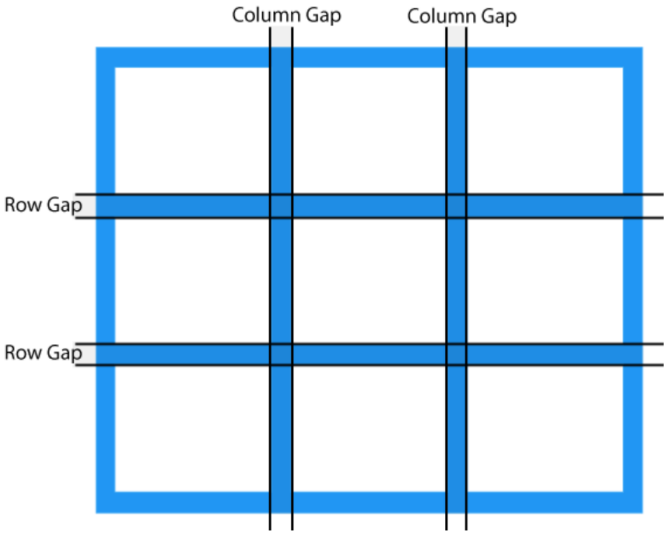
```html
<div class="grid-container">
  <div class="grid-item">1</div>
  <div class="grid-item">2</div>
  <div class="grid-item">3</div>
  <div class="grid-item">4</div>
  <div class="grid-item">5</div>
  <div class="grid-item">6</div>
  <div class="grid-item">7</div>
  <div class="grid-item">8</div>
  <div class="grid-item">9</div>
</div>
```

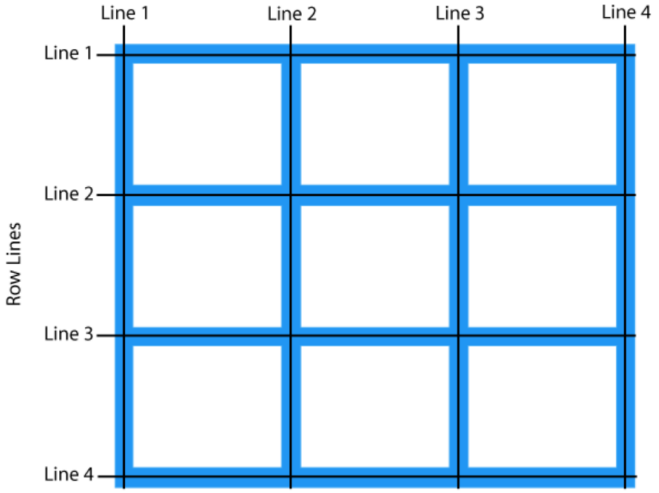
1	2	3
4	5	6
7	8	9

Proprietà Grid: Struttura Pagina Web









Oltre il normal flow: sommario

- Proprietà usate per specificare la posizione di elementi
  - position: static (default), fixed, relative, or absolute
    - Elemento positionato se proprietà diversa da static
    - Proprietà left, right, top, bottom si applicano a elementi posizionati
      - Valore di default auto
  - float: none, left, or right
    - Si applica a elementi con posizionamento statico e relative
  - grid

Struttura della pagina

Come definire la struttura della pagina

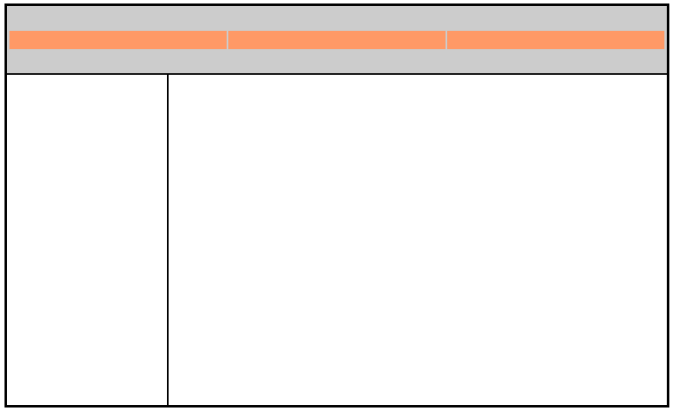
- Originariamente veniva usato la struttura tabella di HTML
  - Tag <table>
  - Ai tempi era l'unico modo possibile
  - Spesso si usavano tabelle innestate una nell'altra

- DEPRECATO!

Come definire la struttura della pagina

- Tabella con struttura esterna (struttura-con-tabelle_1.html)

```
<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"> </td>
<td bgcolor="#FF9966"> </td>
<td bgcolor="#FF9966"> </td>
</tr>
</table>
</td>
</tr>
<tr>
<td width="191" bgcolor="#FFFFFF"> </td>
<td width="599" bgcolor="#FFFFFF"> </td>
</tr>
</table>
</td>
</tr>
</table>
```



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

- Ritorniamo alla struttura della pagina della lezione precedente

```
<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

- 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

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

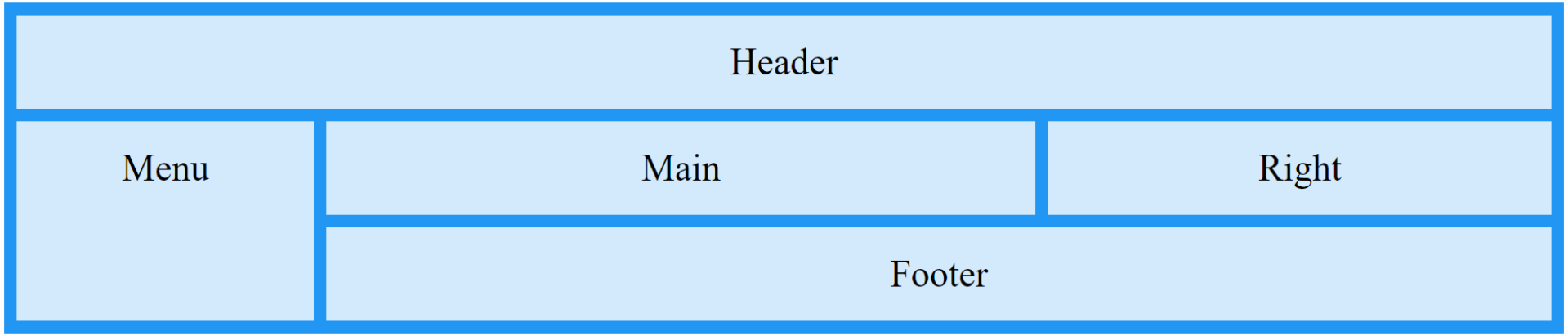
```
#footer {
        clear: both;
        width: 1000px;
        height: 70px;
        background-color: #C4C4C4;
        padding: 10px;
        margin: 10px;
}
```

Struttura della pagina con tag

div-con-css.html

Struttura della Pagina Web con Proprietà Grid

- Definizione dell'area di occupazione dei singoli blocchi
  - Approccio simile a quello usato dai framework (ad es. Bootstrap)
  - Viene definita per ogni componente della pagina la sua estensione in colonne e righe



https://www.w3schools.com/css/tryit.asp?filename=trycss_grid_layout_named

Proprietà Grid: Struttura Pagina Web

- Definizione dell'area di occupazione dei singoli blocchi
  - Approccio simile a quello usato dai framework
  - Viene definita per ogni componente della pagina la sua estensione in colonne e righe

```css
.item1 { grid-area: header; }
.item2 { grid-area: menu; }
.item3 { grid-area: main; }
.item4 { grid-area: right; }
.item5 { grid-area: footer; }

.grid-container {
  display: grid;
  grid-template-areas:
    'header header header header header header'
    'menu main main main right right'
    'menu footer footer footer footer footer';
  gap: 10px;
  background-color: #2196F3;
  padding: 10px;
}

.grid-container > div {
  background-color: rgba(255, 255, 255, 0.8);
  text-align: center;
  padding: 20px 0;
  font-size: 30px;
}
```

```html
<div class="grid-container">
  <div class="item1">Header</div>
  <div class="item2">Menu</div>
  <div class="item3">Main</div>
  <div class="item4">Right</div>
  <div class="item5">Footer</div>
</div>
```

https://www.w3schools.com/css/tryit.asp?filename=trycss_grid_layout_named

Come definire la struttura della pagina: Oltre vanilla CSS

- Framework per sviluppo siti responsive
- Bootstrap (https://getbootstrap.com/)
  - <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-4bw+/aepP/YC94hEpVNVgiZdgIC5+VKNBQNGCHeKRQN+PtmoHDEXuppvnDJzQIu9" crossorigin="anonymous">
- VEDREMO IN DETTAGLIO IN LABORATORIO

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