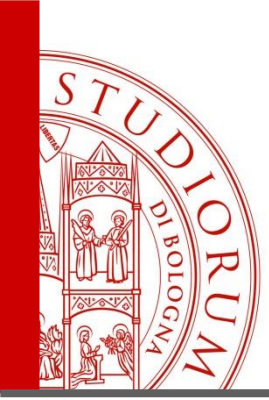


CSS – 3.0

CSS

- Colori
- Sfondo
- Bordi e Box
- Testo, Liste e Immagini
- Transizioni e Animazioni
- Responsive Design e Media Query

Descrivere brevemente le principali novità introdotte da CSS3: Nuove proprietà per il background, i gradienti, bordi con border radius e usare l'immagine come bordo

Colore - Valori

- I colori possono essere specificati nei seguenti modi:
  - Keyword (es: **red**, **green**, …) come avete visto negli esempi della scorsa lezione. Trovate la lista completa qui https://www.w3.org/wiki/CSS/Properties/color/keywords
  - Notazione esadecimale: **`#RRGGBB`** (che è possibile abbreviare nella forma **`#RGB`** in caso di valori duplicati).
  - Notazione decimale: **`rgb(val, val, val)`** dove **`val`** è un valore tra 0 e 255.
  - Notazione decimale con trasparenza: **`rgba(val, val, val, opa)`** dove **`val`** è un valore tra 0 e 255 e **`opa`** è un valore tra 0 e 1 dove:
    - 0 indica la trasparenza totale
    - 1 l'assenza totale di trasparenza (stesso effetto di rbg).
  - **`opacity`**: proprietà che può essere usata in combinazione con un colore definito in **rgb** (con uno qualsiasi dei metodi precedenti) e che gestisce la trasparenza sia dello sfondo che del testo di un elemento.

Colore - Valori

– HSL: acronimo di Hue, Saturation e Lightness, rappresenta uno spazio colorimetrico diverso. Viene specificato come **`hsl(h, s, l)`**.
I tre valori indicano rispettivamente:

- **h**: è un grado di angolazione del cerchio cromatico (ammette quindi valori da 0 a 360)
- **s**: indica la saturazione del colore (in percentuale)
- **l**: indica la luminosità (sempre in percentuale).

```
hsl(0, 100%, 30%);
hsl(0, 100%, 50%);
hsl(0, 100%, 70%);
hsl(0, 100%, 90%);
```

opacitá

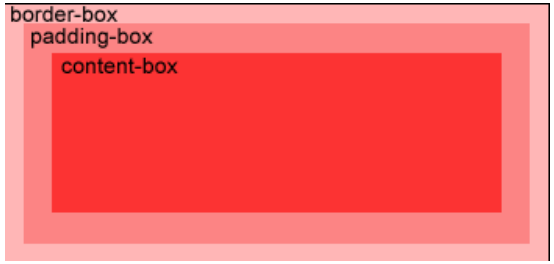
– HSLA: estensione di HSL che include il canale alpha.

Sfondo – CSS 2

- È possibile gestire lo sfondo di un elemento usando le seguenti proprietà:
  - **`background-color`**: permette di specificare il colore di sfondo.
  - **`background-image`**: permette di specificare l'url di un immagine di sfondo (es: `url("..img/prova.png")`).
  - **`background-repeat`**: permette di specificare se e come l'immagine deve essere ripetuta. Valori: **`repeat`**, **`repeat-x`**, **`repeat-y`**, **`no-repeat`**.
  - **`background-attachment`**: permette di specificare il meccanismo di scrolling: **`scroll`** o **`fixed`**.
  - **`background-position`**: permette di specificare la posizione dell'immagine. Accetta due valori: posizione orizzontale e verticale. È possibile specificarli in lunghezza, percentuale o con una keyword (**`top`**, **`bottom`**, **`right`**, **`left`** e **`center`**).
- Esiste la proprietà abbreviata **`background`**.

Sfondo – CSS3

- CSS3 aggiunge le seguenti proprietà:
  - **`background-size`**: permette di specificare la dimensione dell'immagine di sfondo. Esempio:
    ```
    background-size: width height;
    ```
  - **`background-origin`**: permette di posizionare l'immagine di sfondo nel content-box, nel padding-box oppure nel border-box.



  - ***Sfondi con immagini multiple***: è possibile dichiarare più immagini come sfondo. Il risultato è la sovrapposizione di tutte le immagini.
    Esempio:
    ```
    background: url(img1.jpg) left top no-repeat,
    url(img2.jpg) right bottom no-repeat, url(img3.jpg) left
    top repeat;
    ```
    NB: nell'esempio precedente, **`img1`** è quella visualizzata sopra a tutte le altre mentre **`img3`** è quella visualizzata sotto.
    NB2: come immagini di sfondo possono essere usate anche GIF animate

CSS3 – gradienti lineari

- **`linear-gradient`**: permette di specificare un gradiente lineare come sfondo.
  Sintassi: **`linear-gradient(direction, color-stop1, color-stop2, ...);`** dove:
  - **`direction`** è opzionale, di default è dall'alto verso il basso, altrimenti accetta come valori angoli (es: **`90deg`**) o keyword (es: **`to bottom right`**).
  - È possibile specificare un numero di colori a piacere (almeno 2).

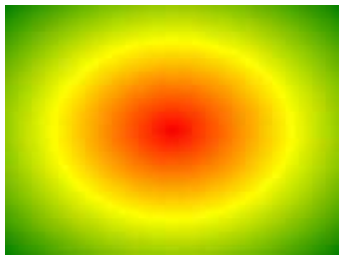
- Esempio:
  `background: linear-gradient(red, yellow, green);`
- **`repeating-linear-gradient`**: è possibile impostare la ripetizione del gradiente lungo l'elemento per il quale si imposta lo sfondo.
  Sintassi: **`repeating-linear-gradient(direction, color-stop1, color-stop2 dimension, ...);`**

- closest-side: Il gradiente si interrompe quando l'estensione (il raggio, per i cerchi; i raggi orizzontale/verticale, per le ellissi) incontra il lato del box più vicino al centro.
- closest-corner: Il gradiente si interrompe quando l'estensione incontra l'angolo del box più vicino al centro. Qui il "cerchio finale" sarà più piccolo rispetto a farthest-corner, perché deve appena arrivare a toccare uno degli angoli minori.
- farthest-side: Come closest-side, ma la dimensione si estende fino al lato più lontano dal centro.

CSS3 – gradienti radiali

- farthest-corner (default): Il gradiente si interrompe quando l'estensione incontra l'angolo del box più lontano dal centro, disegnando l'area di sfumatura più ampia possibile.

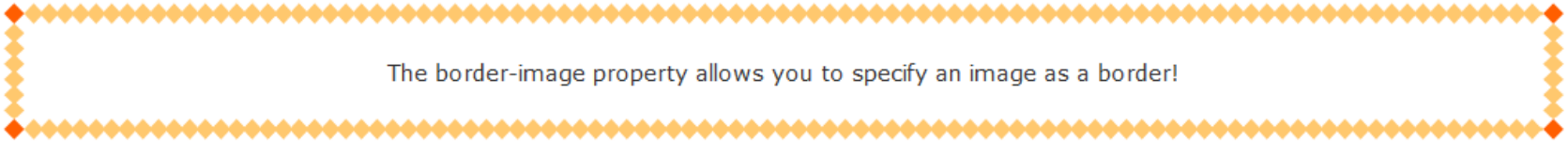
- **`radial-gradient`**: permette di specificare un gradiente radiale come sfondo.
  Sintassi: **`radial-gradient(shape size at position, start-color, ..., last-color);`**
  dove:
  - *shape:* **`ellipse`** (default) o **`circle`**
  - *size at position:* **`closest-side`**, **`farthest-side`**, **`closest-corner`**, **`farthest-corner`**
  - Anche qui è possibile specificare un numero di colori a piacere (almeno 2).



- Esempio:
  `background-image: radial-gradient(red, yellow, green);`
- **`repeating-radial-gradient`**: è possibile impostare la ripetizione del gradiente lungo l'elemento per il quale si imposta lo sfondo.
  Sintassi: **`repeating-radial-gradient(shape size at position, start-color dimension, ..., last-color dimension);`**

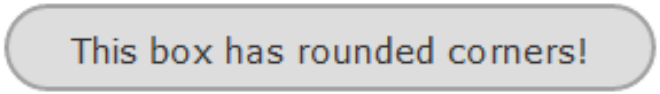
Bordi – CSS 3

- **`border-image`**: permette di specificare una immagine che viene usata come bordo.
  Esempio: `div {border-image: url(border.png) 30 30 round;}`



  Dove **`border.png`** è 

- **`border-radius`**: permette di specificare bordi arrotondati.
  Proprietà estese: **`border-top-left-radius`**, **`border-top-right-radius`**, **`border-bottom-right-radius`** e **`border-bottom-left-radius`**
  Esempio: `div {border: 2px solid; border-radius: 25px;}`



Box

- **`box-shadow`**: permette di specificare l'ombra del box.
- **`resize`**: permette all'utente di ridimensionare i box.
- **`box-sizing`**: permette di far rientrare le dimensioni di padding e bordi nel computo di **`width`** e **`height`** (**`border-box`**). Di default ha valore **`content-box`**.

Rende ridimensionabile dall'utente un elemento che di solito ha dimensioni fisse

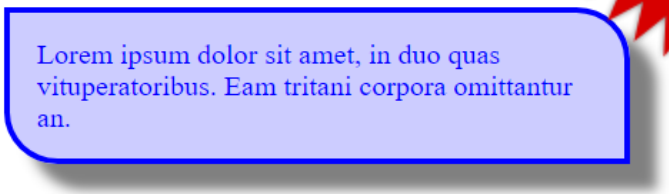
border-box: width e height includono anche padding e bordo, semplificando i calcoli di layout

content-box (default): width e height definiscono solo la dimensione dell'area contenuto; padding e bordo si sommano fuori da questi valori.

Prima domanda

BONUS

- **DOMANDA 1:**
  Quale insieme di regole di stile consente al browser di rendere un testo come quello mostrato qui a lato?



☐
```
border-radius: 30px 30px;
border: 3px blue solid;
box-shadow: grey 15px 15px 0px;
background-color: rgba(0, 0, 255, 0.2);
color: blue;
```

☐
```
border-radius: 0px 30px;
border: 3px blue solid;
box-shadow: grey 5px 5px;
background-color: rgba(0, 0, 255, 0.2);
color: blue;
```

☒
```
border-radius: 0px 30px;
border: 3px blue solid;
box-shadow: grey 15px 15px 10px;
background-color: rgba(0, 0, 255, 0.2);
color: blue;
```

☐
```
border-radius: 0px 0px;
border: 3px blue solid;
box-shadow: 15px 15px 10px;
background-color: rgb(0, 0, 255);
color: blue;
```

Gestione del testo

- Esistono diverse proprietà per la gestione del testo che si occupano:
  - Dell'aspetto dei caratteri
  - Della formattazione del testo

Aspetto dei caratteri

```
#tower-of-pisa {
font-style: italic;
}
```

- Un font è insieme completo di caratteri contraddistinti da un particolare disegno.
- È possibile specificare le seguenti proprietà:
  - **`font-family`**: specifica il nome di uno o più font (es: *Verdana* e *Helvetica*) o un font generico (*serif, sans-serif, monospace, cursive* e *fantasy*).
  - **`font-style`**: specifica lo stile: *normal*, *oblique*, *italic*. Determina lo stile "inclinato" del font.
  - **`font-variant`**: applica l'effetto maiuscoletto (*small-caps*), di default *normal*. Abilita varianti "speciali" dei caratteri
  - **`font-weight`**: specifica il peso, in diversi modi: Indica il peso (spessore) del font.
    - Valori numerici: da *100* a *900*
    - Parole chiave: assolute (*normal* e *bold*) e relative (*bolder* e *lighter*).
  - **`font-size`**: specifica la dimensione dei caratteri, espressa come:
    - Dimensione assoluta: pixel, punti o keyword (*xx-small*, *x-small*, *small*, *medium*, *large*, *x-large*, *xx-large*).
    - Dimensione relativa: em, ex, percentuale o keyword (*smaller*, *larger*).
- Esiste la proprietà abbreviata **`font`** che comprende tutte le precedenti

Formattazione del testo

- **`color`**: colore del testo espresso con: nome del colore (es: red) o valori esadecimali (es: #rrggbb) o rgb (es: rgb(0,255,2).
- **`letter-spacing`**: *normal* o valore in pixel. Definisce lo spazio aggiuntivo orizzontale tra i caratteri
- **`line-height`**: interlinea espresso in lunghezza o percentuale. Imposta l'altezza della riga (interlinea), ovvero la distanza tra le basi delle line box.
- **`text-align`**: *left, right, center* o *justify*.
- **`text-decoration`**: *none, underline, overline* o *line-through*.
- **`text-direction`**: da destra a sinistra (*rtl)* o viceversa (*ltr).*
- **`text-indent`**: indentazione della prima riga di testo, espressa come lunghezza o in percentuale. rientranza
- **`text-overflow`**: permette di specificare il comportamento nel caso in cui porzioni di testo fuoriescano dal box che lo contiene.
- **`text-shadow`**: permette di specificare l'ombreggiatura di un testo come *h-shadow v-shadow blur-radius color.*
- **`text-transform`**: *none, capitalize, uppercase* o *lowercase.*

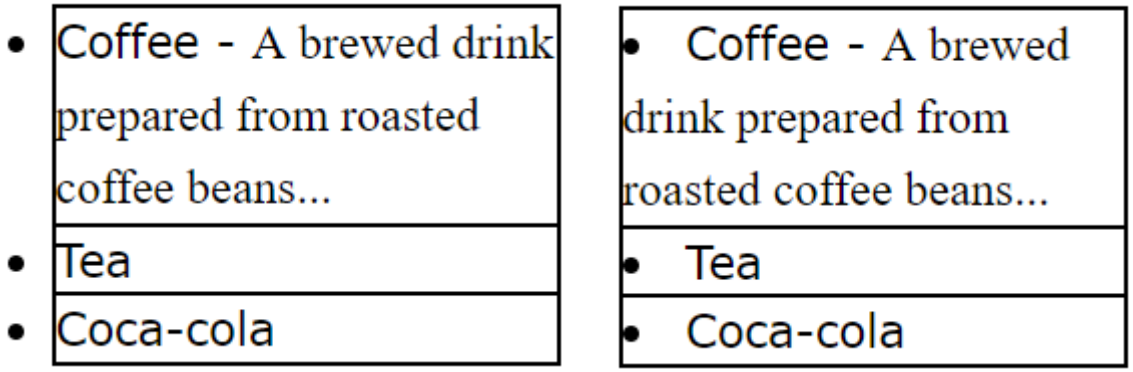
(prima lettera di ogni parola maiuscola)

Formattazione del testo (2)

- **`white-space`**: specifica come sono gestiti spazi bianchi e andate a capo. Possibili valori:
  - normal: sequenze di spazi bianchi collassati in uno solo, il testo va a capo quando necessario.
  - nowrap: sequenze di spazi bianchi collassati in uno solo, il testo andrà a capo solo in corrispondenza di un br.
  - pre: sequenze di spazi bianchi saranno mantenute, il testo andrà a capo solo in corrispondenza di un br o un line break.
  - pre-line: sequenze di spazi bianchi collassati in uno solo, il testo andrà a capo se necessario o in corrispondenza di un line break.
  - Pre-wrap: sequenze di spazi bianchi saranno mantenute, il testo andrà a capo solo in corrispondenza di un br o un line break.
- **`word-wrap`**: permette di forzare l'andata a capo per le parole molto lunghe che non rispettano i bordi dell'elemento contenitore.
- **`word-spacing`**: *normal* o valore in pixel. (Imposta lo spazio tra le parole usando il valore del font o valore manuale)
- **`vertical-align`**: allineamento degli elementi inline. Possibili valori: *baseline, sub, super, top, text-top, middle, bottom, text-bottom*, o percentuale (riferita all'interlinea).

Liste

- **`list-style-position`:** specifica la posizione del marker, se dentro al testo (*inside*) o fuori (*outside*).



- **`list-style-image`:** specifica un'immagine come marker.
- **`list-style-type`:** specifica il tipo di marker. Ne esistono tantissimi sia per **`ul`** (disc, circle, square, …) che **`ol`** (upper-roman, lower-alpha, …).
- **`list-style`**: proprietà abbreviata.

Filtri per immagini

- Con la proprietà **`filter`** è possibile applicare alle immagini effetti visivi di vario genere.
- Esempio:

```
img{
    filter: grayscale(100%);
}
```

Filter – Possibili valori

- **`blur`**`(px)`: consente di applicare una sfocatura.
- **`brightness`**`(%)`: consente di regolare la luminosità.
- **`contrast`**`(%)`: consente di regolare il contrasto.
- **`drop-shadow`**`(hs vs b s c)`: consente di specificare un'ombreggiatura.
- **`grayscale`**`(%)`: converte l'immagine in bianco e nero.
- **`hue-rotate`**`(deg)`: applica una rotazione di deg gradi della tonalità rispetto al cerchio cromatico.
- **`invert`**`(%)`: inverte i colori dell'immagine.
- **`opacity`**`(%)`: consente di regolare il livello di opacità dell'immagine.
- **`saturate`**`(%)`: consente di regolare la saturazione dell'immagine.
- **`sepia`**`(%)`: converte l'immagine in seppia.
- **`url`**`()`: indica l'url di un file XML con un filtro SVG da applicare all'immagine.

Seconda domanda

BONUS

- **DOMANDA 2:**
  Considerando la foto nelle slide precedenti (con l'ananas), con quali regole css posso ottenere l'effetto riportato qui a lato?

☒
```
filter: sepia(100%);
border-radius: 150px;
opacity: 0.5;
```

☐
```
filter: saturate(10);
border-radius: 0px;
opacity: 0.5;
```

☐
```
filter: blur(10px);
border: 5px solid grey;
opacity: 0.5;
```

☐
```
filter: sepia(50%);
border-radius: 0px;
opacity: 1;
```

At rules

- Le ***At rules*** sono regole precedute da una @ e servono per specificare determinati comportamenti.
- Si dividono in:
  - Regular rules: `@[KEYWORD] (RULE);`

    Esempi:
    - `@charset`: permette di specificare l'encoding
    - `@import`: permette di importare regole da un altro file `.css`
  - Nested rules: `@[KEYWORD]{ regole css }`

    Esempi:
    - `@font-face`: permette di definire un font custom
    - `@support`: permette di verificare se una determinata regola è supportata o meno dal browser.

Transizioni

- Le transizioni sono effetti che permettono di applicare passaggi graduali da uno stile all'altro per un determinato elemento.
- Gestibile tramite le seguenti proprietà:
  - **`transition-property`**: proprietà che viene modificata (richiesta)
  - **`transition-duration`**: durata della transizione (richiesta)
  - **`transition-timing-function`**: velocità di esecuzione della transizione
    Questa proprietà determina la curva di velocità della transizione, ovvero come la velocità della transizione varia nel tempo.
    Esempio:
    - ease-out: inizio veloce, poi rallentamento.
    - ease-in-out: inizio e fine lenti, accelerazione al centro.
  - **`transition-delay`**: indica quando la transizione inizia
- Esiste la proprietà abbreviata **`transition`**.
- Usando **`transition`** è possibile anche specificare più transizioni per elementi diversi. È sufficiente separarli con una ",".
  Esempio = transition: background-color 0.5s ease, transform 1s ease-in-out;

Animazioni - Keyframes

- Con la regola **`@keyframe`** è possibile definire delle animazioni, che coinvolge una o più proprietà.
- Sintassi:

```
@keyframes nome{
    selettoreKeyFrame{ …}
}
```

dove

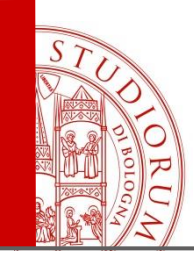
  - *nome*: sarà il nome della nostra animazione.
  - *selettoreKeyFrame*: è la percentuale dell'animazione. Consistono in valori da 0% a 100% o nelle keyword from(0%) e to(100%).

- Una volta definita una animazione è necessario definire a quale elemento applicarla usando la proprietà **`animation`**.
- Sintassi:

```
animation: nome durata;
```

Animazioni - Proprietà

- Proprietà per le animazioni:
  - **`animation-name`**: nome dell'animazione
  - **`animation-duration`**: durata dell'animazione
  - **`animation-timing-function`**: velocità di esecuzione dell'animazione
  - **`animation-delay`**: indica quando l'animazione inizia
  - **`animation-iteration-count`**: indica quante volte deve essere ripetuta l'animazione
  - **`animation-direction`**: indica se l'animazione deve essere eseguita al contrario oppure no
  - **`animation-play-state`**: indica se e quando l'animazione deve essere eseguita oppure deve essere messa in pausa
  - **`animation-fill-mode`**: indica lo stato finale dell'elemento animato, una volta terminata l'animazione



Ancora sui layout

- Ritorniamo sull'esempio di layout della lezione precedente.
- Perché secondo voi non è sufficiente impostare un layout fluido?

Ancora sui layout

- Ritorniamo sull'esempio di layout della lezione precedente.
- Perché secondo voi non è sufficiente impostare un layout fluido?
- Se si prova a restringere la finestra del browser, si può notare che con il layout fluido tutti gli elementi contenuti si restringeranno in maniera appropriata fino ad un **certo punto** (che, dipendendo dai contenuti del sito, varia da sito a sito).
- Oltre questo punto è necessario cambiare la disposizione degli elementi all'interno della pagina. Questo può essere fatto con le **media query**.

Media Query

- Le media query permettono di applicare (o meno) delle regole CSS in base al tipo e alle caratteristiche del dispositivo su cui si visualizza la pagina Web.
- È possibile specificarle in due modi:
  - Direttamente nell'attributo *media* nel tag link che importa il foglio di stile
    `<link rel="stylesheet" media="media-query" href="style.css"/>`
  - Con il costrutto **`@media`** direttamente nel codice CSS.
- Sintassi:
  `@media not|only mediatype and (mediafeature and|or|not mediafeature) { codice css }`
- In pratica, viene associata un'espressione ad un insieme di regole CSS. Se quest'espressione risulta vera, le regole vengono applicate, altrimenti no.

Media Query – Media Type

- **All**: indica tutti i media type per tutti i tipi di dispositivi. È il valore di default.
- **Print**: serve per specificare le stampanti.
- **Screen**: serve per specificare uno schermo generico(desktop, tablet, smartphone,…).
- **Speech**: serve per specificare gli screen reader, dispositivi che utilizzano la sintesi vocale per «leggere» il contenuto della pagina

destinato alla stampa paginata o alla modalità anteprima di stampa

destinato agli schermi. È il media type più usato per definire layout responsive basati sulla larghezza o risoluzione del viewport.

Media Query – Media Features

- Ne esistono tante, a cui ne saranno aggiunte altre con CSS 4.
- Le più utilizzate sono:
  - **`width`**: indica la larghezza della finestra del browser (il viewport). Accetta i prefissi *min-* e *max-*
  - **`orientation`**: indica l'orientamento del dispositivo, *landscape* o *portrait*.
- Attenzione ad utilizzare **`device-width`** al posto di **`width`**! Infatti, **`device-width`** indica la larghezza del dispositivo. Se ridimensionate la finestra del browser, la larghezza del dispositivo rimane invariata! Per questo motivo è preferibile utilizzare **`width`**.

Breakpoints

- «Si restringeranno in maniera appropriata fino ad un **certo punto**»
- Come faccio ad identificare il punto giusto?
- In generale, esistono dei *breakpoints* che sono utilizzati generalmente per identificare smartphone, tablet e pc.
- I range sono:
  - < 768 per smartphone
  - >= 768 e <1024 per tablet
  - >= 1024 per desktop
- Ma possono essere usati anche più range! Esempio:
  - < 576 extra small device
  - >= 576 e < 768 small device
  - >= 768 e < 992 medium device
  - >= 992 e <1200 large device
  - >= 1200 extra large device

Media Query – Esempi

- `@media print{ }`
- `@media screen and (min-width: 480px) { }`
- `@media screen and (max-width: 699px) and (min-width: 520px) {}`
- `@media screen and (max-width: 699px) and (min-width: 520px), (min-width: 1151px) { }`
- `@media only screen and (orientation: landscape) { }`

Media Query – Esempi

- **`@media print{ }`**
  Le regole vengono applicate se il dispositivo di riferimento è la stampante.
- **`@media screen and (min-width: 480px) { }`**
  Le regole vengono applicate se il dispositivo di riferimento è uno schermo e la sua dimensione è almeno 480px.
- **`@media screen and (max-width: 699px) and (min-width: 520px) {}`**
  Le regole vengono applicate se il dispositivo di riferimento è uno schermo, la sua dimensione è almeno 520px ma minore di 700px.
- **`@media screen and (max-width: 699px) and (min-width: 520px), (min-width: 1151px) { }`**
  Le regole vengono applicate se il dispositivo di riferimento è uno schermo, la sua dimensione è almeno 520px ma minore di 700px OPPURE la sua dimensione è almeno 1151px.
- **`@media only screen and (orientation: landscape) { }`**
  Le regole vengono applicate se il dispositivo di riferimento è uno schermo e la larghezza del documento è maggiore dell'altezza dello stesso.

Media Query – Esercizio

- Dato il sito riportato in figura, si vuole fare in modo che quando la dimensione dello schermo è inferiore a 768px, l'immagine non sia più floating ma segua il normale flusso della pagina.

"Lorem ipsum dolor sit amet, consectetur adipiscing ad minim veniam, quis nostrud exercitation ullamco reprehenderit in voluptate velit esse cillum dolore eu culpa qui officia deserunt mollit anim id est laborum tempor incididunt ut labore et dolore magna aliqua. U aliquip ex ea commodo consequat. Duis aute irure dc Excepteur sint occaecat cupidatat non proident, sunt amet, consectetur adipiscing elit, sed do eiusmod ten nostrud exercitation ullamco laboris nisi ut aliquip ex esse cillum dolore eu fugiat nulla pariatur. Excepteur anim id est laborum." "Lorem ipsum dolor sit amet, c magna aliqua. Ut enim ad minim veniam, quis nostru aute irure dolor in reprehenderit in voluptate velit ess proident, sunt in culpa qui officia deserunt mollit ani do eiusmod tempor incididunt ut labore et dolore ma nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate non proident, sunt in culpa qui officia deserunt mollit anim id est laborum." "Lorem ipsum d labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occ laborum." "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor i nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irur pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mc elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim v consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugi qui officia deserunt mollit anim id est laborum." "Lorem ipsum dolor sit amet, consectetur ac Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea con

Viewport Virtuale

- In alcuni casi, i dispositivi con schermo piccolo (come gli smartphone) renderizzano la pagina in una finestra (viewport) virtuale più grande dello schermo e poi restringono il risultato della renderizzazione in modo che tutto il contenuto sia visibile.
- Esempio: se lo schermo di un dispositivo è largo 640px, le pagine vengono renderizzate con una finestra virtuale di 980px e poi viene ristretta in modo che si adatti ad uno spazio di 640px.
- Questo succede perché molte pagine non sono ottimizzate per i dispositivi mobili e si vedevano male in schermi così piccoli.

Viewport Virtuale - Esempio

- Nel caso in cui il **viewport** virtuale sia di **980px**, allora la media query potrebbe non essere mai usata perché la condizione `max-width: 768px` potrebbe risultare sempre falsa!
- Per ovviare a questo problema è possibile utilizzare il metatag **viewport**.

Meta tag viewport

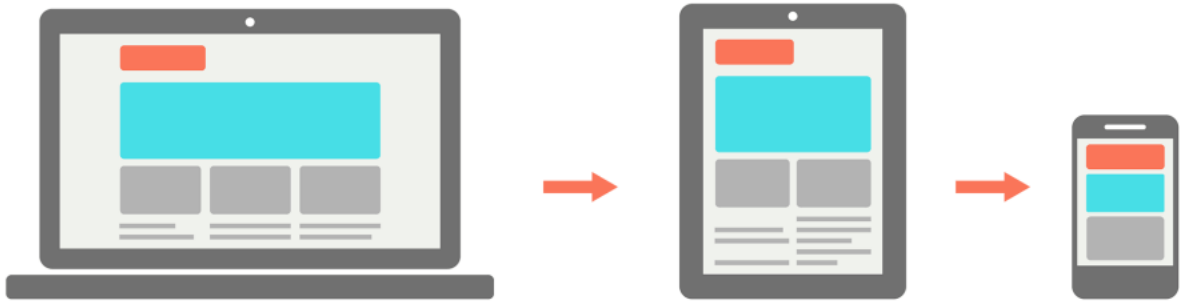
- Il meta tag **viewport** è stato introdotto con HTML5 proprio per ovviare a questo tipo di problemi.
- Il meta tag **viewport** di un sito ottimizzato per mobile ha solitamente il seguente contenuto:

```
<meta name="viewport" content="width=device-width,
initial-scale=1.0"/>
```

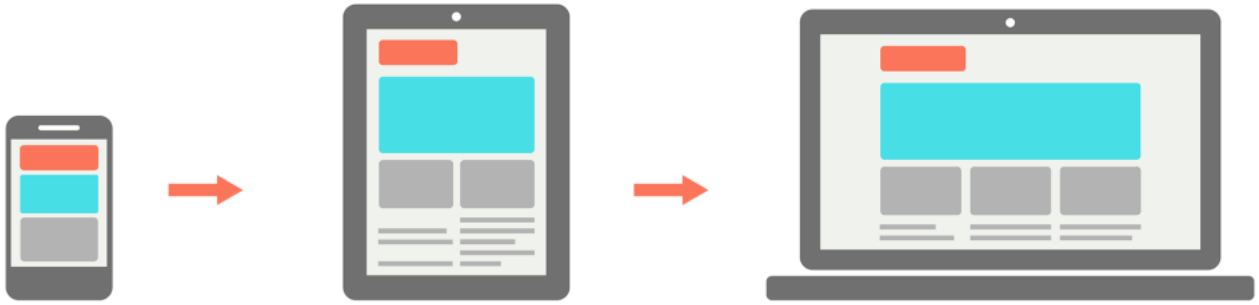
- Dove:
- **`width=device-width`**: imposta la larghezza del viewport in modo tale che segua la larghezza del display del device
- **`initial-scale=1.0`**: imposta il livello di zoom iniziale quando la pagina viene caricata per la prima volta dal browser

Tornando ai layout

- Nella lezione sul Web Design, avete visto due possibili approcci:
  - Desktop First



  - Mobile First



Desktop First vs Mobile First - Esempi

- A partire dal layout di esempio presentato nella scorsa lezione, si vuole fare in modo che:
  - Quando la larghezza dello schermo è compresa tra 768px e 1024px il menù deve occupare il 100% della pagina, i relativi link devono essere disposti orizzontalmente, l'**`article`** deve occupare il 70% della pagina mentre il restante spazio dovrà essere occupato dall'**`aside`**.
  - Quando la larghezza dello schermo è inferiore a 768px, l'**`article`** dovrà occupare il 100% della pagina mentre l'**`aside`** non dovrà essere visualizzato.

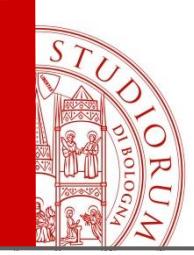
Responsive Design – Scrolling

- L'utente è abituato a fare lo scrolling verticale, sia su PC sia su device mobile
- Lo scrolling orizzontale invece è **SEMPRE SCONSIGLIATO** in termini di user experience!
- Alcune regole per evitare lo scrolling orizzontale

Utilizzare sempre le percentuali

1. Non usare elementi con larghezza prefissata, soprattutto se di grandi dimensioni. Ad esempio, se una immagine è mostrata ad una larghezza maggiore rispetto a quella del **viewport**, allora ci sarà scrolling orizzontale.
2. Non basarsi solo sulla larghezza di un unico **viewport**. Con **viewport** di altre dimensioni si potrebbe avere un effetto totalmente differente, soprattutto nel caso di elementi con larghezze espresse in pixel o in unità di misura assolute.
3. Usare le media query per offrire layout adeguati e adatti a display di diverse dimensioni.
4. Accertarsi che la somma di spazio occupata da elementi inline (o inline-block) non sia mai superiore al 100%, facendo attenzione anche agli spazi bianchi e alle andate a capo nel codice HTML.

"chiudere" i tag senza spazi: item
item.



Conclusioni

- CSS vuole risolvere la separazione tra contenuto (HTML) e presentazione.
- Usa una sintassi tutta sua, usabile sia all'interno del documento che in un documento autonomo
- Le implementazioni di CSS sono quanto di più variabile si possa trovare. Non esiste un browser che implementi tutto CSS esattamente, ci sono differente tra SO e SO, versione e versione, browser e browser.
- Bootstrap è un framework che vi faciliterà la vita nello sviluppo del CSS.

Riferimenti

- Standard completi:
  - CSS1, https://www.w3.org/TR/CSS1/
  - CSS2, http://www.w3.org/TR/CSS2
  - CSS3, https://www.w3.org/TR/2001/WD-css3-roadmap-20010523/
- Approfondimenti su «CSS3 Guida completa per lo sviluppatore», Peter Gasston, 2011. Disponibile in biblioteca

Domande?