CSS II Parte - Appunti

Page Layout e Graphic Design

- **Graphic Design:** Arte e mestiere di creare stile e presentazione visuale di un testo o documento multimediale.
  - Tipografia (typesetting)
  - Organizzazione della pagina (page layout)
  - Organizzazione iconografica (visual art curation)
- **Page Layout:** Disposizione armoniosa degli elementi visuali sulla pagina.
  - Dimensioni, posizione, aree vuote.
  - Paginazione: Problemi di fronte/retro e pagine affiancate.

Aspetti del Page Layout

- **Orientamento:**
  - Portrait (verticale)
  - Landscape (orizzontale)
  - Tradizione: libri (portrait), cinema/TV (landscape), computer (landscape), smartphone (portrait), tablet (variabile).
- **Aspect Ratio:** Rapporto tra altezza e larghezza.
  - Quadrato (1.0000)
  - US Letter (1.2941)
  - 4/3 (1.3333) - TV analogiche
  - 16/9 (1.7777) - TV digitali, smartphone
  - ISO 216 (√2 or 1.4142) - Carta europea (A4, A3, ecc.)
  - Regola aurea (1.6180) - Rapporto ideale
- **Dimensioni:**
  - ISO 216: Standard internazionale per la carta (serie A, B, C).
    - Si basa sul piegare a metà il lato lungo, mantenendo l'aspect ratio.
    - A0: $1 m^2$.
- **Risoluzioni:**
  - Densità degli elementi visuali (DPI in stampa, PPI negli schermi).
  - *Pixel density* è più preciso di "risoluzione" per gli schermi.
  - Schermi meno densi delle stampanti.

Tipo	Risoluzione (DPI/PPI)
Stampante	60-2400 DPI
Schermo	72-807 PPI

- **Griglie:** Struttura bidimensionale per allineare gli elementi.
  - Dense (senza margini) o sparse (con celle vuote).
  - Twitter Bootstrap: Griglia a 12 colonne.
- **Sezione Aurea:** Rapporto aureo ($\varphi \approx 1.618$).
  - Considerato piacevole all'occhio.
  - Usato in tipografia e page design (Jan Tschichold).

Selettori e Regole CSS

- **Selettore:** Specifica un elemento o una classe di elementi HTML/XML.
  - Esempi: `h1`, `#p1`, `.codice`, `p.codice`, `img[alt]`.
- **Regola:** Blocco di statement associati a un selettore.
  - Sintassi: `selettore { statement; statement; ... }`
  - Esempio: `h1 { color: white; background-color: black; }`

Tipi di Selettori

1. **Universale, tipo, classe e id:**

Pattern	Significato	Esempio
`*`	Qualunque elemento	`*`
`E`	Elemento di tipo E	`h1`
`E.nomeclasse`	Elemento di classe nomeclasse	`p.codice`
`.nomeclasse`	Qualunque elemento di classe nomeclasse	`.codice`
`E#ilmioid`	Elemento con id ilmioid	`tr#abc1`
`#ilmioid`	Qualunque elemento con id ilmioid	`#abc1`

2. **Pseudo-elementi:**

Pattern	Significato	Esempio
`E::first-line`	Prima riga formattata dell'elemento E	`p::first-line`
`E::first-letter`	Prima lettera formattata dell'elemento E	`p::first-letter`
`E::before`	Contenuto generato prima dell'elemento E	`q::before`
`E::after`	Contenuto generato dopo l'elemento E	`q::after`

3. **Prossimità:**

Pattern	Significato	Esempio
`E F`	Elemento F discendente di un elemento E	`table th`
`E > F`	Elemento F figlio di un elemento E	`tr > th`
`E + F`	Elemento F successivo diretto di un elemento E	`label + input`
`E ~ F`	Elemento F successivo di un elemento E	`h1 ~ p`

4. **Attributi:**

Pattern	Significato	Esempio

`E[foo]`	Elemento E con attributo foo	`img[alt]`
`E[foo="bar"]`	Elemento E con attributo foo uguale a "bar"	`table[border="1"]`
`E[foo~="bar"]`	Elemento E con attributo foo che contiene la parola "bar"	`p[class~="codice"]`
`E[foo^="bar"]`	Elemento E con attributo foo che inizia per "bar"	`p[class^="cod"]`
	`a[href]`	`a[href]`
	`a[href^='http']`	`a[href^='http']`

5. **Pseudo-classi strutturali:**

Pattern	Significato	Esempio
`E:nth-child(n)`	Elemento E che è l'n-simo figlio di suo padre	`p:nth-child(odd)`
`E:nth-last-child(n)`	Elemento E che è l'n-simo figlio di suo padre (dall'ultimo)	`p:nth-last-child(1)`
`E:nth-of-type(n)`	Elemento E che è l'n-simo figlio di suo padre di quel tipo	`p:nth-of-type(even)`
`E:first-child`	Elemento E che è il primo figlio di suo padre	`h1:first-child`

6. **Pseudo-classi (altre):**

Pattern	Significato	Esempio
`E:active`	Elemento E attivato dall'utente	`a:active`
`E:hover`	Elemento E con puntatore sopra	`a:hover`
`E:enabled`	Elemento E di interfaccia abilitato	`input`
`E:checked`	Elemento E di interfaccia "checked"	`input:checked`

Proprietà CSS

- **Canvas e Viewport:**
  - **Canvas:** Area virtuale di posizionamento (piano cartesiano infinito).
  - **Viewport:** Parte visibile del canvas (dipende dalla dimensione dello schermo).
  - Disegno fuori schermo con coordinate negative.
  - Elemento HTML `<canvas>`.

Unità di Misura

1. **Basate su canvas e viewport:**

   - **Pixel (px):** Unità principale, ma poco affidabile (dipende dal dispositivo). Usare con cautela (eccezioni: `0px` e `1px`).
   - **Viewport width (vw):** 1% della larghezza del viewport.

- **Viewport height (vh):** 1% dell'altezza del viewport.
- **vmin:** Il minore tra vw e vh.
- **vmax:** Il maggiore tra vw e vh.
- Suggerimento: Usare unità di viewport per layout responsive.

2. **Flex (fr):**
   - Dimensione flessibile che rappresenta una frazione dello spazio rimanente nel contenitore.
   - Comodo per rapporti complessi.
     - Esempio: `grid-template-columns: 20% 2fr 1fr;`

Posizionamento della Scatola

- **Posizionamento:**
  - **static (default):** Posizione normale nel flusso.
  - **absolute:** Posizione specificata indipendentemente dal flusso.
  - **relative:** Spostamento dalla posizione naturale.
  - **fixed:** Posizione assoluta rispetto alla finestra (non scrolla).
  - **sticky:** Posizione naturale, ma fissa durante lo scrolling (finché il contenitore è visibile).
- **Proprietà:** `position`, `float`, `top`, `bottom`, `left`, `right`, `width`, `height`.
- `float`: sposta un box a destra o sinistra,
- **z-index:** Posizione nella pila di scatole sovrapposte (valore più alto = più vicino).
- **overflow:** Gestione del contenuto che non entra nella scatola.
  - `visible`: Espande la scatola.
  - `hidden`: Nasconde il contenuto extra.
  - `scroll`: Abilita lo scrolling.
  - `overflow-x` e `overflow-y`: Controllo separato per le scrollbar.

Colonne di Testo

- Gestione di colonne multiple con:
  - `column-width`
  - `column-gap`
  - `column-rule`

Layout in CSS

- **Proprietà `display`:** Gestisce natura e organizzazione della scatola.
  - Valori di default per ogni elemento HTML (es: `block`, `inline`, `table`, ecc.).
  - `display: none;`: Nasconde un elemento.
  - `display: contents;`: Fa sparire la scatola esterna, mantenendo il contenuto.
  - `display: grid;` and `display: flex;` permettono layout complessi.
- **Layout "naturale"** segue il flusso di `block` e `inline`.
- **Tecniche per layout personalizzati:**
  1. **Tabelle HTML:** Sconsigliato.
  2. **Float:** Limitato a tre scatole.
  3. **Positioning:** Complesso con valori proporzionali.
  4. **Tabelle CSS:** (`display: table`, `table-row`, `table-cell`). Complesso, senza `rowspan` o `colspan`.
  5. **Grid:** (`display: grid`).
     - Griglia con righe e colonne controllabili.

    - `grid-template-rows`, `grid-template-columns`, `gap`.
    - `grid-row`, `grid-column`, `grid-area`.
6. **Flexbox:** (`display: flex`).
    - Contenuti flessibili e distribuiti armonicamente.
    - `flex-direction`, `flex-wrap`, `justify-content`.
    - `flex-shrink`, `flex-grow`, `order`.

Altri Aspetti di CSS

- **Cascata, Ereditarietà e `!important`:**
    - **Ereditarietà:** Proprietà non specificate ereditano il valore dal contenitore (default: `inherit`).
        - Eccezioni: `display`, `background`.
    - **`!important`:** Aumenta la priorità di uno statement (anche rispetto a statement successivi).
- **Cascata:**
    - Algoritmo di ordinamento delle dichiarazioni.
    - Principi (dal più al meno importante):
        1. Media-type
        2. Importanza (`!important`)
        3. Origine (utente, autore, user agent)
        4. Specificità del selettore
        5. Ordine delle dichiarazioni
    - Ordine di precedenza:
        1. Dichiarazioni user agent
        2. Dichiarazioni utente
        3. Dichiarazioni normali autore
        4. Dichiarazioni importanti autore (`!important`)
        5. Dichiarazioni importanti utente (`!important`)
- **Trasformazioni CSS:**
    - Trasformazioni geometriche sulla scatola (dopo la sua generazione).
    - `transform: function(parameters);`
    - Funzioni: `translate()`, `scale()`, `rotate()`, `skew()`, ecc.
- **@rules (At-rules):**
    - Meta-regole del foglio di stile.
    - `@import`, `@charset`, `@namespace`, `@page`, `@font-face`, `@media`, `@keyframes`.
    - **`@font-face`:** Specifica font non installati.
    - **`@media`:** Regole dipendenti dal device (media queries).
- **Media Queries:**
    - Regole attivate in base a vincoli sul supporto (es: larghezza, altezza, colore).
    - Sintassi: `@media query { selettore { statement; statement; ... } }`
    - Operatori: `and`, `or`, `only`, `not`.
    - Features: `width`, `height`, `aspect-ratio`, `color`, `hover`, ecc.
- **Animazioni in CSS:**

    - **`@keyframes`:** Specifica stati (iniziale, finale, intermedi) di proprietà numeriche.
    - Proprietà: `animation-name`, `animation-delay`, `animation-duration`, `animation-iteration-count`, `animation-timing-function`, `animation`.

- **Limiti del CSS:**
    - Mancanza di flessibilità (regole non condividono valori).
    - Difficoltà di definire regole basate su altre regole o formule aritmetiche.
- **Pre-processori CSS (LESS, SASS, SCSS):**
    - Estendono CSS con variabili, mix-in, aritmetica.
    - Compilazione o interpretazione.
- **Variabili e calcoli aritmetici in CSS (Level 4):**
    - **Custom property:** Proprietà ad hoc (`--nome-variabile: valore;`).
    - **`:root`:** Selettore per il document element (scope globale).
    - **`var()`:** Accede al valore di una custom property.
    - **`calc()`:** Permette calcoli aritmetici.

Framework CSS: Twitter Bootstrap

- Librerie pronte con regole predefinite.
- **Pregi:**
    - Gestione differenze browser.
    - Accesso facilitato a effetti speciali.
    - Layout responsive.
    - Look integrato.
- **Limiti:** Uniformazione del look.
- **Twitter Bootstrap:**
    - Suite di classi CSS.
    - Layout responsive.
    - Griglia a 12 colonne.
    - Classi predefinite: `col-{size}-{12esimi}` (es: `col-xs-6`).
    - Navbar, finestre modali, ecc. (con un po' di Javascript).

Framework CSS: Tailwind

- Libreria CSS recente (2021).
- **Utility:** Classi CSS in rapporto 1-1 con proprietà e valori CSS.
- Prefissi per media query (dimensione dello schermo).

Responsive Web Design

- Progettazione per ottimizzare l'esperienza su tutti i device.
- **Elementi:**
    - Griglia fluida (dimensioni in %).
    - Immagini flessibili.
    - Uso di `@media` query.