CSS

- La proprietà display
  - Valori di base
  - Flexbox, grid, …
- Media query
- Preprocessori CSS
  - Sass, Less
- Progetto di riferimento
  - https://github.com/egalli64/nesp *(modulo 2b)*
    - Web App Node Express

La proprietà display

- Il rendering degli elementi è basato sul “normal flow”
  - Esaminato il content, gli si aggiunge il padding, border e margin
  - Se **block**: width = 100% del parent, height determinata dal content
  - Se **inline**: sulla riga corrente se c’è spazio, o sulla riga successiva
  - *Block flow direction*, basata sulla direzione di scrittura
- Per cambiare la modalità di un elemento, **display**
- Oltre a block e inline, può assumere altri due valori di base
  - **inline-block**: inline con padding e margini su tutti lati (come IMG)
  - **none**: l’elemento non viene mostrato

Flexbox

- Permette di disporre elementi in una dimensione
- Flex container, proprietà di visualizzazione flex:
  - **display**: **flex** → determina un flex container
  - Altre proprietà permettono di definire i dettagli
    - flex-direction, flex-wrap, justify-content, align-items, …
- Flex items, elementi inseriti in un flex container
  - **flex** combinazione di flex-grow, flex-shrink, flex-basis
    - **none**, equivale a 0 0 auto
    - **initial**, default, equivale a 0 1 auto
    - **auto**, equivale a 1 1 auto
    - **n** (intero positivo), equivale a n n 0

Grid

- Permette di disporre elementi su due dimensioni
- Gli elementi in un grid container usano il suo *layout grid*
  - **display**: **grid** container grid (per default con una sola colonna)
  - **grid-template-columns**
    - Specifica le dimensioni in frazioni per le colonne, es. 1fr 2fr 1fr
    - Se le colonne hanno la stessa dimensione, funzione CSS, es. repeat(3, 1fr);
  - **gap** distanza tra elementi
    - Equivalente alla coppia row-gap, column-gap

float

- **float**: **left** | **right** → elemento rimosso dal normal flow
  - Posizionato a sinistra | destra nel suo container
  - Il resto del content che segue viene posizionato allo stesso livello
    - **clear** forza un elemento che segue un “float” a stare sotto
      - Va specificato a quale float mi riferisco (left, right, both)
- Uso di un “block formatting context” (BFC) container
  - **display: flow-root**
  - Organizza gli elementi contenuti, float e altri (con eventuale clear)

Media query

- Regole CSS valide solo in certe condizioni
  - Ad esempio solo per un certo tipo di media: print, screen, speech
- Responsive Web Design
  - Regole subordinate alla dimensione del viewport
  - Con breakpoint si intende il punto in cui cambiano le regole
  - Esempio, per piccoli schermi:
    - @media screen and(max-width: 600px) { /* … */ }

Preprocessori

- Linguaggi che aumentano le funzionalità di CSS
- Il codice deve essere convertito in CSS per essere utilizzabile dai browser
- Per diffusione spiccano **Sass** (sintassi originale e **SCSS**) e **Less**

	Sass	Less
Installazione via Node	npm install -g sass	npm install -g less
Conversione a CSS	sass <src> <dest> sass --watch <srcDir>:<destDir>	lessc <src> [dest]
Variabili	prefissate da $	prefissate da @
Mixin	Regole parametrizzate Identificate come @mixin Referenziate via @include	Classi postfissate con ()
Nesting	Regole dentro altre regole	