|  |
| --- |
| **Laboratorio 2 – Martedì 13/10/2020** |
| **Specifiche**   * Specifiche HTML5: <https://www.w3.org/TR/html5/> * Specifiche CSS3: <https://www.w3.org/TR/css-syntax-3/> |
| **Ripasso sui selettori**   * I selettori permettono di selezionare uno o più nodi dell’albero DOM. Successivamente con le parentesi graffe si associano proprietà grafiche a questi nodi. * **Sito per sperimentare selettori in modo dinamico**: <https://www.w3schools.com/cssref/trysel.asp> * **Selettori basilari**:   + tag elemento *tag* (per esempio html, body, p, div, span…)   + .classe elemento appartenente alla classe *classe*   + tag.classe elemento *tag* appartenente alla classe *classe*   + #nomeid elemento avente id *nomeid* * **Combinazione di selettori**:   + Selector1, selector2 elemento Selector1 o elemento Selector2   + Selector1 Selector2 elemento *Selector2* discendente dell’elemento *Selector1*   + Selector1 > Selector2 elemento *Selector2* il cui genitore stretto è *Selector1*   + Selector1 + Selector2 elemento *Selector2* posto subito dopo l’elemento *Selector1*   + Selector1 ~ Selector2 elemento *Selector2* fratello di *Selector1* (stanno sullo stesso livello, cit).  Pensare al DOM e al livello in cui si trovano gli elementi. * **Selezione a partire da attributi**:   + [attribute] elemento che ha definito un certo attributo   + [attribute = value] elemento che ha definito un certo attributo ponendo un valore value   + [attribute |= value] elemento che ha definito un certo attributo il cui valore inizia con value   + [attribute $= value] elemento che ha definito un certo attributo il cui valore termina con value   + [attribute ~= value] elemento che ha definito un certo attributo il cui valore contiene la parola value   + [attribute \*= value] elemento che ha definito un certo attributo il cui valore contiene la sottostringa value |
| **Ripasso sulle pseudo-classi**   * Selector:hover  per associare a un elemento *Selector* proprietà che si manifestano quando ci troviamo sopra l’elemento col cursore del mouse. * Selector:active  per associare a un elemento *Selector* proprietà che si manifestano quando clicchiamo col cursore sopra l’elemento ma non abbiamo ancora sollevato il dito dal tasto del mouse. * Selector:visited  per associare proprietà a link che sono già stati visitati dall’utente * Selector:link  per associare proprietà a link che non sono stati ancora visitati dall’utente. * Selector:checked  per associare proprietà ad elementi input (*radio* e *checkboxes*) che sono stati selezionati * Selector:nth-child(n)  elemento *Selector*, n-esimo figlio del padre. * Selector:nth-of-type(n) n-esimo elemento *Selector* figlio del padre. * Osservazione sulle ultime due pseudoclassi: può essere un numero, ma anche una formula () o una keyword. In particolare   + *odd*, per considerare gli elementi in posizione dispari   + *even*, per considerare gli elementi in posizione pari |
| **Esercizio grafica sito [Parte 1]**   * I selettori sono cose che si comprendono soprattutto attraverso l’esercitazione. * Prendiamo un file HTML che consiste in un sito classifica di piattaforme social. * In questo momento è presente solo codice HTML: il nostro compito è migliorare la grafica col CSS. * **Link del codice**: <https://jsbin.com/tocozut/13/edit?html,output> (in caso di problemi col link vedere avanti) * **Cose richieste dal docente**:   + Impostare la lunghezza di ogni immagine al 100px   + Mettere un bordo tratteggiato di 1px a tutte le sezioni   + Colorare lo sfondo di rosso del titolo di Youtube   + Colorare lo sfondo di giallo di tutti i link "Torna su"   + Colora di rosso la scritta "100 lingue"   + Colora di sfondo verde la scritta "primo servizio"   Evidenziare la differenza tra parent e ancestor   * + Metti un bordo spesso 5px intorno all'immagine di Youtube   Provare la differenza tra attribute \*= value e attribute ~= value   * + Quando si passa sopra i link "Torna su" cambiare il background in rosso e mettere il font a 50px * **Soluzioni**:   + Impostare la lunghezza di ogni immagine al 100px  img { width: 100px; }   + Mettere un bordo tratteggiato di 1px a tutte le sezioni  section { border: 1px **dotted**; }   + Colorare lo sfondo di rosso del titolo di Youtube  #Youtube h1 { background-color: red; }   + Colorare lo sfondo di giallo di tutti i link "Torna su"  .top {  background-color: yellow; }   + Colora di rosso la scritta "100 lingue".  Si va nel complicato: osservo dove si trova la scritta. L’espressione è contenuta in uno strong collocato all’interno di uno span. Lo span è a sua volta posto all’interno di un div e il div si trova in un paragrafo. Il paragrafo si trova dentro article e infine article è posto all’interno di una sezione con id facebook. Non abbiamo bisogno di indicare tutta la discendenza: ci limiteremo a scrivere quanto segue  #Facebook span strong {  background-color: red; } *// Se usassi il selettore > sarei obbligato a indicare tutto il percorso detto prima.*   + Colora di sfondo verde la scritta "primo servizio"   Evidenziare la differenza tra parent e ancestor   Andiamo a tappe #Facebook div em {  background-color: green; } con questo selettore non andiamo a colorare solo l’elemento richiesto ma anche la scritta “14 maggio 2008”. Come possiamo distinguere i due elementi? Osserviamo che:   * + - L’elemento giusto si trova all’interno di un div     - Quello in più si trova all’interno di uno span     - Entrambi gli elementi sono inclusi all’interno di un altro div.   Possiamo risolvere richiedendo discendenza diretta. Segue #Facebook div > em {  background-color: green; }   * + Metti un bordo spesso 5px intorno all'immagine di Youtube   Provare la differenza tra attribute \*= value e attribute ~= value  Soluzione semplice:  img[alt='logo youtube'] {  border: 5px solid red;  }  Prima soluzione alternativa: img[alt\*='tube'] {  border: 5px solid red;  }  Seconda soluzione alternativa: img[alt~='youtube'] {  border: 5px solid red;  }  La prima soluzione alternativa “matcha” una sottostringa, la seconda una parola. Questo significa che se ponessi “tube” nella tilde il CSS non potrebbe trovare l’area che mi interessa definire graficamente.   * + Quando si passa sopra i link "Torna su" cambiare il background in rosso e mettere il font a 50px .top:hover {   background-color: red;  font-size: 50px;  } |
| **CSS Positioning**   * Nella lettura della pagina HTML gli elementi vengono letti in sequenza e posti all’interno di un flusso. * L’attributo position permette di alterare il normale comportamento degli elementi * **Valori possibili**:   + static: rappresenta la posizione normale che ciascun elemento occupa nel flusso del documento (valore di default).   + relative: l’elemento viene posizionato relativamente al suo box contenitore. In questo caso il box contenitore è rappresentato dal posto che l’elemento avrebbe occupato nel normale flusso del documento. La posizione viene impostata con le proprietà top, left, bottom, right.     - Le coordinate in top, left, bottom, right dipendono dal box contenitore. Porre queste coordinate tutte uguali a zero significa collocare l’elemento in alto a sinistra nel contenitore.     - L’area dove l’elemento avrebbe dovuto collocarsi a comportamento normale rimane vuota e non viene riempita      * + absolute: l’elemento, o meglio, il box dell’elemento, viene rimosso dal flusso del documento ed è posizionato in modo assoluto in base ai valori forniti con le proprietà top, left, bottom, right.     - Finisce in cima al documento, se scorro l’elemento sparisce.     - In un certo senso è come se avessi messo un vetro trasparente sopra il sito posizionando in cima a questo vetro l’elemento. Se io inquadro la parte bassa del vetro l’elemento sparisce.   + fixed: usando questo valore il box dell’elemento viene, come per absolute, sottratto al normale flusso del documento. La differenza sta nel fatto che per fixed il box contenitore è la viewport.     - La viewport può essere immaginata come la finestra di casa.     - L’elemento con position fixed si comporterà come una tenda: indipendente da quello che si vede dalla finestra rimarrà sempre lì. * **Esercitazione**:   + Per esercitarci usiamo il seguente codice   <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8"> <title>Test CSS positioning</title>  </head>  <body>  <section>  <span id="static">static</span>  <span id="relative">relative</span>  <span id="fixed">fixed</span>  <span id="absolute">absolute</span>  </section>  <style>  section { height: 300px; border: 1px solid;}  span { border: 1px solid; }  #static { position: static; }  #relative { position: relative; }  #fixed { position: fixed; }  #absolute { position: absolute; }  </style>  </body>  </html>  Se manipoliamo il CSS impostando valori per le proprietà top, left, bottom e right individueremo al volo le differenze. |
| **Floating box**   * Con **float** è possibile rimuovere un elemento dal normale flusso del documento e spostarlo su uno dei lati (destro o sinistro) del suo elemento contenitore. Il contenuto che circonda l’elemento scorrerà intorno ad esso sul lato opposto rispetto a quello indicato come valore di float. * Gli elementi **float** vengono resi automaticamente block-level: questo significa che si può attribuire loro una larghezza e/o un’altezza via CSS. * La proprietà **clear** serve a impedire che al fianco di un elemento compaiano altri elementi con il float. Si applica solo agli elementi blocco e non è ereditata. * **Utilità della proprietà float**: *sidebar*, contenuti laterali di una qualunque pagina. Non vi scappi in mente l’idea di utilizzare le tabelle per dividere il sito in colonne: le tabelle non sono fatte per questo. |
| **Esercizio: stile delle tabelle**   * Vogliamo rendere più accattivante lo stile della seguente tabella   <table id="classifica">  <caption><em>Top 10 Classifica Social</em></caption>  <thead id="classifica\_head">  <tr>  <th>Rank</th>  <th>Social</th>  <th>Active Users</th>  </tr>  </thead>  <tbody>  <tr>  <td>1</td>  <td>Facebook</td>  <td>2498 ML</td>  </tr>  <tr>  <td>2</td>  <td>Youtube</td>  <td>2000 ML</td>  </tr>  […]  <tr>  <td>10</td>  <td>Weibo</td>  <td>516 ML</td>  </tr>  <tfoot>  <tr>  <td colspan = "3" class="footer">  <a target="\_blank" href="#">Data Source</a>  </td>  </tr>  </tfoot>  </tbody>  </table>   * **Procediamo così**:   + Creiamo dei bordi, distinguiamo le varie celle della tabella th, td {   border: 1px solid black;  }   * + Di default i bordi delle celle sono distinti attraverso margine. Eliminiamo questo margine e poniamo un width al 100% in modo tale che la nostra tabella si estenda su tutta la pagina.   table#classifica {  width: 100%;  border-collapse:collapse;  }     * + Diamo uno stile diverso all’header della tabella (la prima riga). Impostiamo le seguenti proprietà:   thead {  line-height: 40px;  font-family: Verdana;  background-color: orange;  color: white;  }     * + Impostiamo un font diverso per le righe successive e aumentiamo la spaziatura (per le celle della tabella abbiamo impostato di default vertical-align: middle, quindi il testo sarà centrato verticalmente indipendentemente dall’altezza impostata).   tbody {  font-family: Arial;  }  tr {  height:40px;  }   * + Centriamo il testo in tutte le celle tranne in quelle relative ai social (seconda colonna)   td {  text-align: center;  }  th:nth-of-type(2), td:nth-of-type(2) {  text-align:left;  padding-left: 40px;  }   * + Impostiamo lo sfondo del footer della tabella uguale a quello della prima riga   tfoot {  background-color: orange;  color: white;  }   * + Coloriamo in modo diverso le righe in posizione pari della tabella   tr:nth-of-type(2n) {  background-color: lightgray;  }     * + Impostiamo un width per la prima e la seconda colonna: voglio ridurre la prima e allargare la seconda.   th:nth-of-type(1) {  width: 10%;  }  th:nth-of-type(2) {  width:70%;  }   * **Risultato finale**: |
| **Esercizio: scacchiera**   * Costruiamo una scacchiera a partire da una tabella avente otto righe e otto colonne (da questo si deduce il codice HTML della tabella). * Imposto le proprietà della cella   td {  border: 1px solid;  width: 50px;  height: 50px;  }   * Faccio collassare i bordi della cella (in modo tale da non avere margini fastidiosi)   table {  border-collapse: collapse;  }   * Cambio lo sfondo di alcune celle. Precisamente:   + nelle righe in posizione dispari coloro le celle in posizione dispari;   + nelle righe in posizione pari coloro le celle in posizione pari.   tr:nth-of-type(odd) > td:nth-of-type(odd), tr:nth-of-type(even) > td:nth-of-type(even) {  background-color: black;  }   * **Risultato**: |
| **CSS Horizontal navigation bar**   * Supponiamo di avere una lista del seguente tipo:   <ul id=”barra”>  <li><a href="#Facebook">Facebook</a></li>  <li><a href="#Youtube" >Youtube</a></li>  <li><a href="#Whatsapp" >Whatsapp</a></li>  <li><a href="#other" >...</a></li>  </ul>   * Vogliamo creare una barra di navigazione orizzontale a partire da essa. * Per fare ciò dobbiamo intervenire sulle seguenti proprietà:   + float: left <--- modifico la disposizione degli elementi li sullo schermo   + list-style-type: none; <--- Elimino i simboli della lista (in ul)   + overflow: hidden; <--- In caso di trabocco da ul la parte traboccante viene nascosta   + text-decoration: none; <--- Elimino decorazioni tipiche delle anchors (li a).   + text-align: center; <--- centro il link all’interno dell’elemento li (li a). |
| **Scroll behavior**   * Abbiamo già visto che è possibile creare URL in grado di mandarci a specifiche aree della pagina. Ciò avviene sfruttando gli id degli elementi. * Se abbiamo la pagina già aperta e usiamo uno di questi URL lo spostamento sarà netto e immediato. * Col seguente codice avremo uno scroll transitorio dall’area in cui ci troviamo all’area dove è presente l’elemento con id indicato.   html {  scroll-behavior: smooth; } |
| **CSS transition**   * Le CSS transitions permettono di cambiare i valori delle proprietà CSS in modo graduale, rispettando certe tempistiche. * Per creare un effetto di transizione bisogna specificare due cose:   + La proprietà che si vuole modificare   + La durata della transizione * **Vediamo il seguente esempio**: abbiamo una serie di rettangolini con bordo blu e sfondo rosso e verde alternato. Se pongo il cursore sopra uno di questi quadratini, questo ruoterà di 90 gradi e cambierà progressivamente colore.      * + *transition.html*   <!DOCTYPE html>  <html>  <head>  <meta name="description" content="CSS transition">  <meta charset="utf-8">  <title>CSS transition</title>  <link rel = "stylesheet" type = "text/css" href ="css/transition.css" />  </head>  <body>  <div></div>  <div></div>  <div></div>  <div></div>  <div></div>  <div></div>  <div></div>  <div></div>  <div></div>  <div></div>  <div></div>  </body>  </html>   * + *transition.css*   */\* creare dei box rettangolari di colore rosso e bordo blu \*/*  div {  width: 50px;  height: 70px;  border: 3px solid blue;  background-color: red;  /\* rendere la transizione dolce di durata 2 secondi \*/  **transition: transform 2s, background-color 2s;**  margin: 2px;  }  /\* colorare di verde quelli pari \*/  div:nth-child(even) {  *background-color: green;*  }  /\* fare una rotazione di 90 gradi e cambiare il colore in giallo al passaggio del mouse \*/  div:hover {  **transform : rotate(90deg);**  background-color: yellow;  }   * + - Nel CSS si gestiscono due transizioni:       * Quella dal colore rosso al colore giallo.         + In div abbiamo indicato il colore dello sfondo iniziale (in div:nth-child(even)abbiamo il colore iniziale dei div pari)         + In div:hover abbiamo indicato il colore dello sfondo finale       * La rotazione di 90 gradi.         + In div:hover abbiamo indicato la rotazione usando la proprietà transform.     - Attraverso la proprietà transition in div abbiamo stabilito la durata di entrambe le transizioni: due secondi per la rotazione (transform 2s), due secondi per il cambio di colore (background-color 2s) |
| **Riprendiamo la scacchiera**   * Riprendiamo il codice della scacchiera e introduciamo una rotazione sull’asse Y delle celle della scacchiera (immaginiamoci il risultato finale come quelle porte dei grandi hotel che girano) * Le aggiunte sono evidenziate   td {  border: 1px solid;  width: 50px;  height: 50px;  **transition: 5s;**  }  **td:hover {**  **transform: rotateY(720deg);**  **}**   * + Con la transform si stabilisce la rotazione di 720 gradi solo sull’asse Y.   + Con la transition si indica che la transizione durerà 5 secondi. Non ho bisogno di indicare a quale effetto sto facendo riferimento (contrariamente all’esercizio precedente): ne ho uno solo. |