

HTML

- Elementi per la gestione di
 - Liste, immagini, tabelle ...
- Generazione di richieste al server via
 - Link
 - Form
 - Widget associati: input, bottoni, ...
- Progetto di riferimento
 - <https://github.com/egalli64/nesp> (*modulo 2a*)
 - È una web app Node / Express

Liste

- Le più usate, con e senza ordine, hanno la stessa struttura
 - Gli elementi contenitore, **ol** e **ul**, contengono un elemento **li** per ogni voce
 - Una lista può contenere altre liste
- Ogni **li** in **ol** ha un indice crescente da 1, modificabili via **reversed** e **start** nella **ul** ogni list item è identificato a un pallino (o altro)
 - CSS per stilare gli elementi list item
 - list-style-type in ol → decimal, lower-roman, lower-alpha, ... in ul → disc, circle, square, ...
 - list-style-position: inside, ...
 - list-style-image: poco configurabile, si preferisce usare background-image
- La lista di definizioni, **dl**, meno usata
 - Contiene combinazioni di **dt** (definition term) e **dd** (definition of definition)

Immagini

- **img**

- Elemento vuoto, non ha tag di chiusura, tutte le informazioni sono negli attributi
- **src**: l'indirizzo della risorsa, che può essere locale o meno
 - ``
 - ``
- **alt**: testo alternativo, da mostrare se l'immagine non è accedibile (e indicizzabile da motori di ricerca)
- **title**: testo aggiuntivo mostrato quando il puntatore passa sull'immagine
 - ``
- **height, width**: dimensioni dell'immagine – se non sono indicate, si usano le dimensioni originali
 - Specificandone una l'altra viene calcolata dal browser. Entrambe: l'immagine può essere distorta
 - Valore assoluto (pixel): ``
 - Percentuale sul contenitore dell'immagine: ``


- L'elemento **figure** è un contenitore per **img** e di **figcaption** per la descrizione relativa

Tabelle

- Collezione di dati in formato tabellare
 - **table** righe (alto → basso) di celle (sinistra → destra)
 - **caption** testo che descrive la tabella
 - **thead** riga (o righe) di intestazione
 - **tr** riga nella tabella (table row)
 - **td** cella contenente dati (table datum)
 - attributi colspan, rowspan
 - **th** Cella di intestazione
 - attributo scope indica se “row” o “col”
 - **tfoot** riga (o righe) di sommario finale

```
<table>
  <caption>My table</caption>
  <thead>
    <tr>
      <th scope="row"></th>
      <th scope="col">Left</th>
      <th scope="col">Right</th>
    </tr>
  </thead>
  <tr>
    <th scope="row">Top</th>
    <td>LT</td><td>RT</td>
  </tr>
  <tr>
    <th scope="row">Bottom</th>
    <td>LB</td><td>RB</td>
  </tr>
</table>
```

Rendering standard: nessun contorno a tabella e celle (CSS)



	Left	Right
Top	LT	RT
Bottom	LB	RB

iframe

- Permette l'embedding nella pagina di un area sotto responsabilità esterna
 - Gli “inline frame” vanno usati con cautela, il provider deve essere affidabile
- L'attributo chiave è **src**
- L'elemento è generato dal sito ospite
 - OpenStreetMap: tasto “share” sulla destra, tab HTML
 - Google Maps: condividi o incorpora mappa, incorporare una mappa
 - ...

```
<iframe src="https://www.openstreetmap.org/export/embed.html?bbox=9.19%2C45.46%2C9.19%2C45.46">
</iframe>
```

```
<iframe src="https://maps.google.it/maps?q=duomo+milano&output=embed">
</iframe>
```

Caratteri speciali

- Alcuni caratteri non utilizzabili in HTML, o non disponibili su normali tastiere, sono resi con “entity”, stringhe che iniziano con “&” e finiscono con “;”

<	<	€	€
>	>	¢	¢
&	&	©	©
"	"	®	®
 	 	 	

Notazioni alternative
Uso di entity number
Esadecimale o decimale

- <https://dev.w3.org/html5/html-author/charref>

Request – Response

- Una web app è divisa in
 - Frontend (sulla macchina dell'utente)
 - Backend (sul server)
- La comunicazione avviene via scambio tra FE e BE di richieste e risposte via HTTP
- Una **request** viene indirizzata al server usando il protocollo HTTP
 - Tra i metodi HTTP in questo contesto si usano solo **GET** e **POST**
 - Lista completa su MDN: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>
- Il server genera una **response** che viene ritornata al chiamante
 - Status code (**200** → OK, **404** → Not found, **500** → Server Error, ...)
 - Risorsa richiesta (quando disponibile)
- In una web app classica la risposta è gestita dal browser

Link

- Semplice gestione dell'ipertestualità nelle pagine HTML
 - Genera una richiesta via GET, è possibile passare parametri per concatenazione nella forma nome=valore
- Elemento **a** con attributo **href** “anchor to a hypertext reference”
- Le richieste per una risorsa interna alla web app sono nella forma `index page`
 - Per richieste esterne si indica l'URL completo, es: <https://www.w3.org/community/webbed/wiki/HTML/Training/Hyperlinks>
- È possibile referenziare un elemento in una pagina
 - Es. nella pagina corrente, dato `<h1 id="top">Hello</h1>` ci si collega con `the top`
- Possibile, ma poco usato, per email: `site administrator`
 - Quale programma aprire la scrivere l'email è solitamente determinato dal sistema operativo in uso
- Tre stati: non visitato, attivo, visitato
- L'attributo **title** può essere utilizzato per dare informazioni aggiuntive al link
- L'attributo **target** specifica come aprire la risorsa, ad es. `_self` (default), `_blank`
- L'elemento **base** con attributo **href** (nella sezione head): URL base da usare nelle seguenti href della pagina

Form

- Gestisce l'interazione con l'utente
- Permette di inviare dati al backend via **submit** indicando negli attributi
 - **method**: il metodo HTTP usato, **GET** o **POST**
 - Opzionale, il default è GET
 - **action**: URI della risorsa richiesta
 - Opzionale, il default è la pagina corrente
- Il form contiene *widget*
 - elementi HTML visualizzati in modo standard
 - Se hanno l'attributo **name**, sono associati a parametri per la request
 - Visti come coppie nome-valore
 - Un apposito bottone (di submit) è usato per far partire la richiesta



Label e fieldset

- Una **label** chiarisce il ruolo del widget associato
 - L'attributo **for** la collega all'elemento con l'id corrispondente
 - Oppure, si riferisce all'elemento contenuto nella label
- Un **fieldset** raggruppare campi correlati
 - Al suo interno, **legend** ne offre una descrizione

```
<fieldset>
  <legend>User</legend>
  <label for="fn">First name</label> <input id="fn" name="first">
  <label>Last name: <input type="text" name="last"></label>
</fieldset>
```

input text

- L'elemento **input** è vuoto, non ha closing tag
 - Alcuni attributi comuni a tutti gli input:
 - Il nome dell'elemento in **name** e il suo valore in **value** sono usati per generare la request
 - **autofocus**, indica quale elemento vogliamo sia selezionato per primo nel form
 - **disabled**, il valore non modificabile e non fa parte della request
 - Quasi tutti:
 - **size** dimensione approssimativa del box, come numero di caratteri visualizzabili
 - **required** indica che un parametro è obbligatorio
 - **readonly**, il valore non può essere modificato dall'utente
 - Specifici per text (et al.)
 - **placeholder** visualizza una indicazione per l'utente su quello che ci si aspetta come input
 - **maxlength** fissa la lunghezza massima del valore
 - **pattern**, la regular expression che il value deve rispettare (uso di: `?*| [-]{n}`)



input + type – textarea

- L'attributo **type** determina il tipo di **input** specifico, tra cui:

- text (default)
- password (dati sensibili)
- hidden (parametro nascosto)
- date (scelta di un giorno)
- range (intervallo)
- color (scelta di un colore)
- file, email, url, ... *(altri input nelle slide successive)*

```
<input type="text" name="user" value="Bob" >
```

- **textarea**: blocco di testo su più righe, può contenere il testo di default

- cols: numero di colonne
- rows: numero di righe

```
<textarea name="comment">Your comment here.</textarea>
```

input radio

- Scelta di una opzione da una lista
 - Tutti gli elementi hanno lo stesso **name** e uno specifico **value**
 - Solo quello selezionato viene passato nella request
 - L'attributo **checked** indica la scelta corrente
- Uso di **label** per mostrare all'utente il nome dell'elemento corrente

```
<label><input type="radio" name="fav" value="Java">Java</label>  
<label><input type="radio" name="fav" value="Python" checked>Python</label>  
<label><input type="radio" name="fav" value="Cpp">C++</label>
```

gruppo
determinato
da "name"

input checkbox

- Scelta di una o più opzioni da una lista
 - Tutti gli elementi hanno lo stesso **name** e uno specifico **value**
 - Tutti quelli selezionati vengono passati nella request, ogni valore associato allo stesso name
 - L'attributo **checked** indica le scelte correnti
- Uso di **label** per mostrare all'utente il nome dell'elemento corrente

```
<label><input type="checkbox" name="lang" value="Java" checked>Java</label>  
<label><input type="checkbox" name="lang" value="Python">Python</label>  
<label><input type="checkbox" name="lang" value="Cpp" checked>C++</label>
```

gruppo
determinato
da "name"

select – option

- Scelta di una opzione da una lista
 - L'elemento **select** fa da container e definisce l'attributo **name**
 - Selezione di più opzioni abilitata con l'attributo **multiple**
 - Voci aggiuntive via control click
 - Ogni **option** definisce un **value** per una singola scelta
 - L'attributo **selected** identifica (una/la) scelta corrente

```
<select name="os">  
  <option value="none">-</option>  
  <option value="linux" selected>Linux</option>  
  <option value="windows">Windows</option>  
  <option value="macOs">MacOS</option>  
</select>
```

gruppo
determinato
da "name"

button

- Si può rendere con l'apposito elemento **button**
 - **type**
 - **submit** (default): determina il submit del form in cui è incluso
 - **reset**: riporta tutti i widget del form in cui è incluso allo stato iniziale
 - **button**: nessun effetto standard
 - content: cosa (HTML) viene mostrato all'utente nel bottone
- Equivalente agli elementi **input** type submit/reset/button
 - L'attributo **value** permette di specificarne il testo (solo caratteri) associato
- L'elemento **input** di type **image**
 - Combinazione di img e submit button
 - L'attributo **src** specifica l'indirizzo dell'immagine associata

audio & video

- Esecuzione di file **audio** / **video** in una pagina
- La sorgente, se unica, può essere indicata nell'attributo **src**
 - Altrimenti si mettono nel content elementi **source**, il browser sceglierà quale processare
- Nel content si specifica anche cosa renderizzare se il browser non supporta questo elemento
- L'attributo booleano **controls** visualizza il widget proprio del browser
 - Mette a disposizione comandi per la gestione delle principali funzionalità
 - Se si vuole una gestione più personalizzata occorre accedere alla API via JavaScript
- Tra gli altri attributi disponibili per audio e video
 - **loop**, booleano, se non specificato l'esecuzione termina normalmente
 - **preload**, può assumere i valori **none**, **metadata**, **auto**
- Solo per video sono disponibili gli attributi
 - **width**, per definire la dimensione in pixel della vista
 - **poster**, immagine da mostrare quando il video non è ancora scaricato

Grafica via canvas & svg

- L'elemento **canvas** mette a disposizione un'area utilizzabile via JavaScript
 - Vedi API Canvas e WebGL
- L'elemento **svg** definisce un viewport con un suo sistema di coordinate al cui interno è definito un frammento SVG
 - SVG (Scalable Vector Graphics) linguaggio basato su XML
 - Descrizione di immagini usando grafica vettoriale 2D
 - Tra i molti attributi
 - **viewBox**: coordinate del viewport del frammento SVG corrente
 - **stroke**: colore del pennello usato per disegnare le figure
 - **fill**: colore interno alle figure