HTML

- Markup Language
  - Scrittura via text editor
  - Rendering e debugging via browser
- Progetti di riferimento
  - https://github.com/egalli64/nesp *(modulo 1)* – richiede npm install
    - VS Code https://code.visualstudio.com/
    - Node.js https://nodejs.org/
  - https://github.com/egalli64/mswa *(modulo 1)*
    - Eclipse, Prospettiva JavaEE
    - Target runtime, un Java EE Web/Application Server come
      - Apache Tomcat http://tomcat.apache.org/ – RedHat JBoss/WildFly https://wildfly.org/

HTML: HyperText Markup Language

- Tim Berners-Lee @CERN ~1990
- World Wide Web Consortium (W3C)
  - HTML5 2014
- Descrive come rappresentare pagine web
- Nell'uso normale, la pagina viene acceduta con un URL (Uniform Resource Locator), via protocollo HTTP e ne viene fatto il rendering con un browser
  - Elevata permissività sintattica
- Struttura ad albero, ogni nodo è un elemento
  - DOM: Document Object Model

```
<!DOCTYPE html>
<!-- my hello page -->
<html lang="en">

<head>
   <meta charset="utf-8">
   <title>Hello</title>
</head>

<body>
   <p>Hello world!</p>
</body>

</html>
```

hello.html

Web Developer Tools

- Firefox / Chrome (DevTools)
- Scorciatoia comune per l’attivazione: ctrl+shift+i
  - Settings (F1), Advanced settings, Disable HTTP cache
  - Tab Debugger, accesso al codice
  - Tab Console, visualizzazione log
  - Tab Inspector, HTML widget
  - Tab Style Editor, CSS

Elemento

- Singolo componente di un documento HTML
- Normalmente delimitato da open – close tag
  - In alternativa, elementi vuoti non hanno il close tag
  - I tag name sono case-insensitive
- Può **contenere** testo e altri elementi
- Può avere attributi nella forma *nome="valore"*
  - L'ordine degli attributi di un elemento non è significativo
  - Gli attributi booleani sono nella forma *nome* o *nome="nome"*
- "!" indica che è un *non*-elemento
  - DOCTYPE tipo di documento. Aiuta il browser a interpretare correttamente il codice sorgente (qui: HTML5)
  - Commenti HTML: <!-- … -->

```
<!DOCTYPE html>
<!-- my hello page -->
<html lang="en">

<head>
    <meta charset="utf-8">
    <title>Hello</title>
</head>

<body>
    <p>Hello world!</p>
</body>

</html>
```

hello.html

head vs body

- html
  - Contiene l’intero codice HTML della pagina
  - l’attributo **lang** specifica il linguaggio del contenuto
- head
  - Informazioni *sulla* pagina
- body
  - Informazioni *nella* pagina
  - Contenuto che vogliamo mostrare all’utente

```
<!DOCTYPE html>
<!-- my hello page -->
<html lang="en">

<head>
   <meta charset="utf-8">
   <title>Hello</title>
</head>

<body>
   <p>Hello world!</p>
</body>

</html>
```

hello.html

head

- Gli elementi in head hanno lo scopo di descrivere la pagina corrente
  - title: il titolo della pagina, solitamente mostrato dal browser nella barra del titolo
    - Comunemente usato dal browser e motori di ricerca come riferimento al documento
  - meta: informazioni aggiuntive, come l'encoding usato e indicazioni per i motori di ricerca

    ```
    <meta charset="utf-8">
    <meta name="description" content="Writing HTML code">
    <meta name="keywords" content="html, head, title, meta">
    ```
  - Per altri tipi di metadati vedi
    - The Open Graph protocol https://ogp.me/

Elementi di blocco vs inline

- Le due principali proprietà display degli elementi
- Blocco
  - Inizia a capo a sinistra e arriva in fondo a destra all'interno del contenitore in cui si trova
  - L'elemento che segue sarà su una nuova linea
  - Di solito rappresentano elementi strutturali della pagina
  - Un blocco non dovrebbe essere contenuto da un inline
- Inline
  - Contenuti in un blocco, occupano solo lo spazio necessario, a partire dal vicino di sinistra
  - Non implicano un andata a capo alla loro fine
  - Spesso associati a un paragrafo (elemento "p")

Testo

- h1..h6
  - Titoli (heading) di parti del testo, ci si aspetta un solo H1 per pagina
- p
  - Paragrafo, unità di base per la suddivisione del testo
- b, i, u, sup, sub, …
  - Formattazione del testo, (bold → grassetto), (italic → corsivo), sottolineato, esponente, pedice, …
  - Obsoleti se indicano solo uno stile – andrebbe usato CSS
- strong, em
  - Enfasi, importante, equivalenti a e ma veicolano più facilmente il loro senso (per cui sono detti “semantici”)
- br
  - HTML ignora molteplicità di spazi, tab, andate a capo, etc. Ogni gruppo è interpretato come un singolo spazio bianco.
  - Per forzare l’andata a capo si usa
, elemento che non ha tag di chiusura
- hr
  - Per separare blocchi nella pagina si può usare l’elemento vuoto horizontal ruler <hr>

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

Liste

- **ol**
  - Contiene un elemento li (list item) per ogni voce
  - Lista ordinata in cui ogni voce ha un indice crescente
    - Inizia da 1, a meno che non sia specificato specificato l'attributo start
    - L'attributo booleano reverse fa sì che l'indice sia decrescente
- **ul**
  - Lista senza ordine, come ol ogni voce è un li, ma identificata da un pallino (o altro) e non un indice
- Una lista può contenere altre liste
  - Ci si aspetta comunque che "li" inizi con testo, e poi segua l'eventuale sublista

- **dl** *(poco usata)*
  - Lista di definizioni, dl può contenere ogni combinazione dei due elementi:
    - dt (definition term), termine da definire; dd (definition of definition), definizione del termine

Link

Gestione dell'ipertestualità nelle pagine HTML

- a – href
  - anchor to a hypertext reference, "ancora" l'elemento alla risorsa definita in href
    - Risorsa interna alla web app: index page
    - Elemento nella pagina corrente
      - Dato un elemento con id: <h1 id="top">Hello</h1>, un anchor può linkarlo: the top
    - href a un elemento in un risorsa nel web: https://www.w3.org/#w3c_crumbs
    - Possibile, ma poco usato, per email: site administrator
  - 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, _blank
- base – href (nella sezione head)
  - URL base da usare nelle seguenti reference della pagina

Immagini

- img – src, alt, title, height, width
  - *Elemento vuoto, non ha tag di chiusura*, tutte le informazioni sono negli attributi
  - src: l'indirizzo della risorsa, che può essere locale o meno
    - <img src="../img/flower.jpg">
    - <img src="https://developer.mozilla.org/static/img/opengraph-logo.72382e605ce3.png">
  - 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
    - <img src="../img/flower.jpg" alt="A flower" title="A flower">
  - height, width: dimensioni dell'immagine
    - Se nessuna delle due è indicata, si usano le dimensioni originali
    - Specificandone una l'altra viene calcolata dal browser. Entrambe: l'immagine può essere distorta
    - Valore assoluto (pixel): <img src="../img/flower.jpg" width="150" height="150">
    - Percentuale sul viewport corrente: <img src="../img/flower.jpg" width="50%">
- figure (HTML5) contiene img e la descrizione relativa come figcaption

iframe

- Inline frame – permette l'embedding di un'altra pagina HTML in quella corrente
- L'attributo chiave è src, generato dal sito ospite

```
<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&amp;output=embed">
</iframe>
```

Tabelle

- Gestione di dati in formato tabellare
- **table**
  - Tabella descritta come collezione di righe (dall'alto verso il basso), a loro volta descritte come collezione di celle (da sinistra a destra)
- **tr**
  - Riga nella tabella (table row)
- **td**
  - Descrive una singola cella (table datum)
  - Attributi colspan, rowspan
- th
  - Descrive una cella di intestazione
  - L'attributo opzionale scope indica se "row" o "col"

```
<table>
   <tr>
      <th></th>
      <th scope="col">Left</th>
      <th scope="col">Right</th>
   </tr>
   <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

Struttura della pagina

- Da HTML5, sono disponibili elementi semantici per rappresentare le principali sezioni che normalmente sono usate in una pagina web
  - header: intestazione del sito web, di solito ripetuta nelle pagine
  - nav: navigation bar, link alle sezioni principali del sito
  - main: il contenuto principale della pagina
    - article, section, div: aiutano a organizzare main
  - aside: informazioni aggiuntive, link a risorse associate
  - footer: informazioni che devono essere presenti ma di importanza secondaria

Blocco = div, inline = span

- L’elemento div rappresenta un contenitore generico
- La sua controparte inline è span
  - Spesso usato per stilare o gestire via JavaScript solo una parte di un “p”
- Sono flessibili ma non veicolano facilmente il loro senso
- Gli elementi semantici
  - Sono preferiti dai motori di ricerca
    - (vedi SEO: Search Engine Optimization)
  - Semplicità di lettura e comprensione anche nello sviluppo della pagina

id vs class

- L’attributo id identifica univocamente un elemento all’interno di una pagina
  - Un elemento può avere un solo id
- L’attributo class identifica un gruppo di elementi in un pagina
  - Un elemento può avere più classi, specificate in un singolo attributo class, separate da uno spazio
- I nomi di id e class sono case-sensitive, non possono contenere spazi
- Sono attributi “globali”, ogni elemento può averli
  - https://www.w3.org/wiki/HTML/Attributes/_Global
- L’uso di class e id è fondamentale nell’interazione tra HTML con CSS e JavaScript

Interazione con utente

- L’elemento form è uno tra i principali strumenti per gestire l’interazione con l’utente
- Nelle web app classiche, hanno lo scopo di permettere l’invio di dati al backend
- Il form contiene widget (elementi HTML visualizzati in modo standard), ognuno dei quali è usato per generare un parametro con i dati da inviare

Request – Response

- Il submit di un form genera una request che viene indirizzata al server usando il protocollo HTTP specificando
  - Metodo usato, tipicamente GET o POST
  - URL destinatario
  - Parametri associati, visti come coppie name → value
- Il server gestisce la request e alla fine genera una response che viene ritornata al chiamante
- Il browser mostra il risultato all'utente

form

- Gli attributi fondamentali di un elemento form sono:
  - action: URL dove devono essere mandati i dati (default, la pagina corrente)
  - method: quale metodo HTTP deve essere usato per spedire il messaggio (default GET)

```
<form action="..." method="post">
  <div>
    <label for="name">Name:</label>
    <input type="text" id="name" name="sender">
  </div>
  <div>
    <label for="msg">Message:</label>
    <textarea id="msg" name="message"></textarea>
  </div>
  <div>
    <button type="submit">Send</button>
  </div>
</form>
```

Submit di un form

- In questo esempio l'input dell'utente avviene via:
  - input-text (stringa di testo)
  - textarea (blocco di testo)
- L'attributo name in ogni widget determina l'associazione con il parametro passato al server
- Le label chiariscono il ruolo del widget associato
  - L'attributo for collega una label al controllo con quell'id
- Il button-submit reagisce a un click dell'utente eseguendo l'azione del form

```
<form action="..." method="post">
  <div>
    <label for="name">Name:</label>
    <input type="text" id="name" name="sender">
  </div>
  <div>
    <label for="msg">Message:</label>
    <textarea id="msg" name="message"></textarea>
  </div>
  <div>
    <button type="submit">Send</button>
  </div>
</form>
```

input text

- input
  - *Elemento vuoto, non ha closing tag*
  - Alcuni attributi comuni a tutti gli input:
    - Il nome dell'elemento è in name, il suo valore in value, usati dal submit
    - 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 (validazione HTML5) 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)
  - file, email, url, … *(altri input nelle slide successive)*

```
<input type="text" name="user" value="Bob" maxlength="30">
<input type="password" name="pwd" maxlength="30" required>
<input type="hidden" name="invisible" value="notShowed">
<input type="date" name="milestone">
```

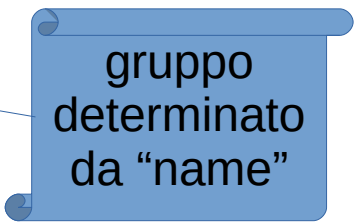
- textarea: blocco di testo su più righe, può contenere il testo di default
  - cols: numero di colonne
  - rows: numero di righe

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

input radio

- Scelta di una opzione da una lista
- L'attributo checked indica *la* scelta di default
- Al click del submit button, il radio button checked determina quale value viene associato al name e messo nella request

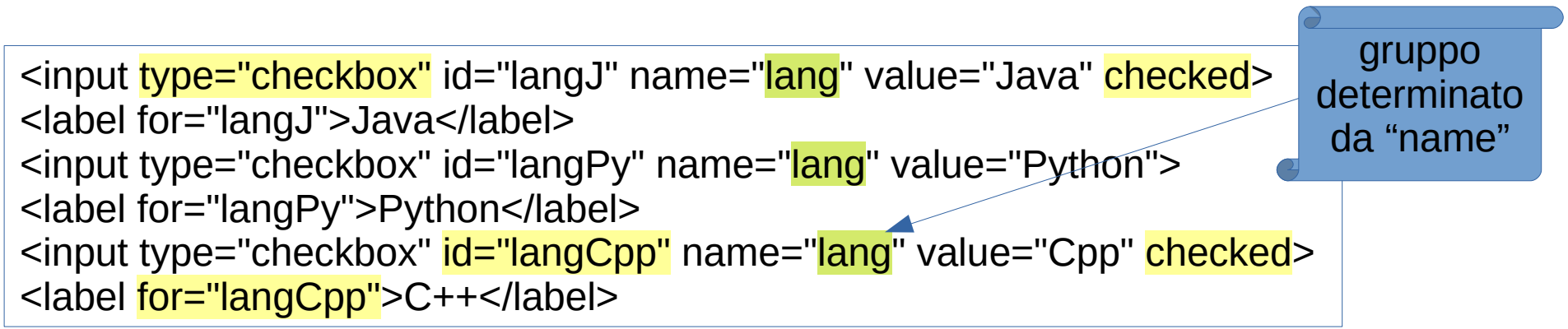
```
<input type="radio" id="favJ" name="fav" value="Java" checked>
<label for="favJ">Java</label>
<input type="radio" id="favPy" name="fav" value="Python">
<label for="favPy">Python</label>
<input type="radio" id="favCpp" name="fav" value="Cpp">
<label for="favCpp">C++</label>
```



input checkbox

- Scelta di più opzioni da una lista
- L'attributo checked indica *le* scelte di default
- Al click del submit button, se c'è almeno un checkbox checked, ogni valore viene associato al "name" (comune) e messo nella request

```
<input type="checkbox" id="langJ" name="lang" value="Java" checked>
<label for="langJ">Java</label>
<input type="checkbox" id="langPy" name="lang" value="Python">
<label for="langPy">Python</label>
<input type="checkbox" id="langCpp" name="lang" value="Cpp" checked>
<label for="langCpp">C++</label>
```



select – option

- Scelta di una opzione da una lista a scomparsa
  - select fa da container e definisce l'attributo name
    - Selezione di più opzioni (via ctrl-click) aggiungendo l'attributo multiple
  - option definisce il value per ogni singola scelta
    - L'attributo selected specifica (un/il) valore di default

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

button

- Elemento button
  - type
    - submit (default): determina il submit del form
    - reset: riporta tutti i widget del form allo stato iniziale
    - button: nessun effetto standard
  - content: cosa (HTML) viene mostrato all'utente nel bottone
- Elemento input type submit/reset/button
  - Uso di "value" per specificarne il testo (solo caratteri) associato
- Elemento input type image (via attributo src)
  - Combinazione di img e submit button

fieldset

- fieldset
  - Permette di raggruppare campi correlati, migliorando la leggibilità di un form
- legend
  - Descrive il fieldset corrente

```
<fieldset>
    <legend>User</legend>
    <label>First name: <input type="text" name="fname"></label>
    <label>Last name: <input type="text" name="lname"></label>
</fieldset>
```

Atomic Design

- Atomo
  - Elementi di base
- Molecola
  - Più atomi che concorrono ad uno scopo comune
- Organismo
  - Componente complesso di molecole cooperanti
- Template
  - Sorta di pagina astratta, completa ma generica
- Pagina
  - Risultato concreto