Corso di programmazione frontend

Pagine e siti web

E’ un documento ipertestuale pubblicato sul World Wide Web e leggibile dall'utente mediante un apposito programma detto browser.

Un sito web è un insieme di pagine web collegate da uno o più nomi a dominio (o sottodomini) con il quale possono essere raggiunte dall'utente

Stile

Struttura/contenuto Comportamentale

HTML5 (1)

E’ un linguaggio di markup utilizzato per definire struttura e contenuto di pagine web

```
<html>
<head>
</head>
  <body>
    <div id="header">
      --- Titolo e Testata ---
    </div>
    <div id="body">
      <div id="menu">
      --- Voci di Menu ---
      </div>
      <div id="main_content">
        <div class="post">
        --- Un Post ---
        </div>
        <div class="post">
        --- Un altro Post ---
        </div>
      </div>
    </div>
  </body>
</html>
```

HTML5 (2)

Caratteristiche principali

- Tag
  marcatori che evidenziano non tanto l'aspetto, quanto il senso, il ruolo, o l'organizzazione che vogliamo assegnare ai contenuti

- Attributi (id, class)
  servono per meglio specificare la funzione o la tipologia dell'elemento, per memorizzare dati o per arricchire di significato il contenuto

- Contenuto

```
<html>
<head>
</head>
 <body>
  <div id="header">
   --- Titolo e Testata ---
  </div>
  <div id="body">
   <div id="menu">
   --- Voci di Menu ---
   </div>
   <div id="main_content">
    <div class="post">
    --- Un Post ---
    </div>
    <div class="post">
    --- Un altro Post ---
    </div>
   </div>
  </div>
 </body>
</html>
```

CSS (1)

Il CSS è un linguaggio utilizzato per la formattazione dei siti web e, più in generale, dei documenti HTML. L'acronimo CSS significa Cascading Style Sheets ossia fogli di stile. Si tratta di un linguaggio di formattazione dei testi che integra il linguaggio HTML

```
<style type="text/css">
    b { color: #FF0000; }
</style>
```

CSS (2) - selettori

I selettori indicano a quali elementi in una pagina html la regola verrà applicata, le proprietà e i relativi valori si curano della presentazione di tali elementi.

```
selettore{
    proprietà: valore;
    proprietà: valore;
    proprietà: valore;
}
```

CSS (3) - esempi

Selettore universale (universal selector)

Selettore di classe (class selector)

Selettore di tipo (type selector)

Selettore di ID (id selector)

Pseudoselettori

NB Gerarchia dei selettori (>)

```
* {
    color: green;
  }

 .nome_classe {
 [...]
 }

selettore1 {
 proprietà1:valore1;
 proprietà2:valore2;
}

 #nome_identificatore {
 [...]
 }

Selettore1:hover {

}

.nome_classe > div {

}
```

Javascript (1)

JavaScript è il principale linguaggio di programmazione per lo sviluppo di web application

ORIENTATO AGLI EVENTI

```
<script type=“text/javascript">
                .....
</script>
```

Javascript (2) - Eventi

Per evento si definisce qualcosa che accade nel documento

- Caricamento
- Click
- Movimento del mouse
- Input dell'utente in un campo di testo

```
<ELEMENTO on-nomeEvento="codice JavaScript"

elemento.addEventListener("click",e=>{
……
})
```

Javascript (3) - Funzioni

una funzione è un insieme di istruzioni racchiuse in un blocco di codice, che può essere contraddistinto da un nome, può accettare argomenti o parametri di ingresso e restituire valori

```
function(può essere vuoto)
 {
   istruzioni;
   return espressione;
 }
```

Javascript (4) - Callback

Un callback è una funzione passata come argomento di un'altra funzione.

```
function myCalculator(num1, num2, myCallback) {
  let sum = num1 + num2;
  myCallback(sum);
}
```

Javascript (4) - Definizione di variabili

Una variabile è uno “spazio di memoria con nome” utilizzato per salvare dati. Possiamo usare le variabili per memorizzare informazioni extra, visitatori e altri dati

```
var message;
let message;
const myBirthday = ’18.04.1982’;


//Uso di let e var


if (true) {
  let test = true; // use “let"
}

alert(test); // ReferenceError: test is not defined
```

Javascript (5) - Document Object Model

Il DOM è un modello che descrive come i diversi oggetti di una pagina sono collegati tra loro.

TAG -> NODI

Può essere manipolato con JS

```
getElementById()

getElementsByTagName()

querySelector()

querySelectorAll()

setAttribute()

getAttribute()

createElement()

appendChild()
```

Javascript (6) - Programmazione async - Promise

la “promise” è una particolare funzione che rende il risultato disponibile quando è pronto.

Esiste una sintassi speciale per lavorare con le promise in un modo più comodo, chiamata async/await. È sorprendentemente facile da capire e usare.

Sync vs Async -> WEB

```
let promise = new
Promise(function(resolve,reject) {

});

//old style
Promise.then(result=>{})


await promise()
```

Javascript (7) - Rest API

Un'API REST ti consente di eseguire l'invio e il recupero di dati da un datastore. Questo potrebbe essere il tuo database o il server di terze parti

GET — Ottieni dati dall'API. Ad esempio, ottieni un utente Twitter in base al suo nome utente.

POST — Invia i dati all'API. Ad esempio, crea un nuovo record utente con nome, età e indirizzo e-mail.

```
// richiesta GET.
fetch('https://api.github.com/users')
    // gestisci il successo
.then(response => response.json())
    // converti a json
.then(json => console.log(json))
    // stampa i dati sulla console
.catch(err =>
console.log('Request Failed', err));
// gestisci gli errori
```

Javascript (8) - Rest API

Fetch funziona con le Promise, non con le funzioni callback.

Il primo parametro della funzione Fetch dovrebbe essere sempre l'URL. Fetch può prendere un secondo parametro di tipo JSON con opzioni come metodo

```
// richiesta GET.
fetch('https://api.github.com/users')
    // gestisci il successo
.then(response => response.json())
    // converti a json
.then(json => console.log(json))
    // stampa i dati sulla console
.catch(err =>
console.log('Request Failed', err));
// gestisci gli errori
```

Javascript (8) - POST

Per una richiesta POST, si usa la proprietà "body" per passare una stringa JSON come input.

Permette di inviare dati

```
// dati da mandare alla richiesta POST
let _data = {
  title: "foo",
  body: "bar",
  userId:1
}

fetch('https://
jsonplaceholder.typicode.com/posts', {
  method: "POST",
  body: JSON.stringify(_data),
  headers: {"Content-type": "application/
json; charset=UTF-8"}
})
.then(response => response.json())
.then(json => console.log(json));
.catch(err => console.log(err));
```

Javascript (9) - JSON

JavaScript Object Notation è un formato per lo scambio dati basato sul linguaggio di programmazione JavaScript.

I tipi di dati supportati da questo formato sono:
- booleani (true e false);
- interi, numeri in virgola mobile;
- stringhe racchiuse da doppi apici (");
- array (sequenze ordinate di valori, separati da virgole e racchiusi in parentesi quadre []);
- array associativi (sequenze coppie chiave-valore separate da virgole racchiuse in parentesi graffe); (JSON)
- null.

```
const profile = {
  name: 'Jane Doe',
  'favorite-game': 'Stardew Valley',
  subscriber: false
}
```