**INIZIARE CON IL WEB**

Installazione software base

Un editor di testo a piacere:

(<https://code.visualstudio.com/>) VISUAL STUDIO CODE

BROWSER WEB (Chrome, Mozilla Firefox, Safari, IE…)

EDITOR GRAFICO (<https://www.gimp.org/>) GIMP, (<https://www.getpaint.net/>) Pain.NET, Photoshop…

SISTEMA DI CONTROLLO DELLE VERSIONI DEL SOFTWARE (GIT)

Un client FTP per trasferire file sul server (Filezilla, WINSCP..)

Un sistema di automazione come GRUNT o GULP per eseguire processi ripetitivi, come minificare il codice ed eseguire test

Template (come UIKIT)

E altro ancora…

**PRIMA COSA: Pianificare**

Prima di fare qualsiasi cosa occorre avere delle idee.

Le domande a cui dare una risposta:

Di cosa tratta il sito web ?(animali, città , viaggi, libri, film…)

Che informazioni presenti sull’argomento? Scrivi un titolo, pochi paragrafi, e a un’immagine che vuoi mostrare

Come appare il sito web? Ovvero, quale colore di sfondo uso, il tipo di carattere

Immagini (<https://www.google.com/imghp?gws_rd=ssl> ) Google Image

FONT (<https://fonts.google.com/>) Google font

Un sito web è compost da molti file: testo, codice, media e prima di pubblicarlo dobbiamo assemblarlo sul nostro computer.

La struttura che ha sul nostro computer deve essere esattamente quella che avrà su un server web.

Vediamo un struttura tipica di un sito web

1. index.html l’inizio del nostro sito web
2. Una cartella images
3. Una cartella styles (o css o stili, come preferisci)
4. Una cartella scripts ( per il codice javascript)

Vedere **esempioSrc.html** per il caricamento di un’immagine

**HTML**

Che cos’è l’HTML?

HTML (**H**yper**t**ext **M**arkup **L**anguage) è il codice che viene utilizzato per strutturare una pagina web e i suoi contenuti.

Ad esempio il contenuto potrebbe essere strutturato all’interno di un insieme di paragrafi, o in una lista a pallini, oppure presentato attraverso immagini o tabelle di dati.

HTML fornisce gli strumenti per strutturare una contenuto che vogliamo presentare nel nostro sito web.

E’ definito come linguaggio di markup ( di marcatura) e consiste in una serie di elementi che si usano per inglobare diverse parti del contenuto in modo tale da farlo apparire in un determinato modo o da farlo comportare in un determinato modo.

Il conenuto è inserito nei cosiddetti tag che sono delle etichette che creano gli elementi.

I tag permettono di creare paragrafi, link ipertestuali, mostrare del testo in grassetto, in corsivo, di un certo colore e via così.

Vediamo un esempio:

Ho il seguente testo

My cat is very grumpy

(grumpy = irritabile, scontroso)

Voglio che questa riga appaia in un paragrafo singolo

<p>My cat is very grumpy</p>

**Vediamo l’anatomia di un elemento HTML**



1. Il tag di apertura ( nel nostro caso <p>) consiste nel nome dell’elemento racchiuso tra parentesi angolari

Tale tag stabilisce dove comincia l’elemento o da dove inizia ad aver effetto.

1. Il tag di chiusura ( </p>) dice dove finisce l’elemento o dove finisce il suo effetto .

(Uno degli errori tipici di chi scrive pagine HTML è dimenticarsi di chiudere un tag aperto. Questa dimenticanza può portare a strani comportamenti della pagina web)

1. Il contenuto è ciò che vogliamo venga manipolato dai tag. In questo caso è solo testo.
2. Tutti i componenti nel loro insieme formano **l’elemento**

Gli elementi possono anche avere **attributi**, come nel seguente caso:



Gli attributi contengono delle informazioni supplementari sull’elemento , che non vuoi che appaiano nel contenuto.

Qui **class** è il nome dell’attributo e **editor-note** è il valore dell’attributo

L’attributo class permette di dare all’elemento un identificativo che può essere utilizzato in seguito per individuare l’elemento con informazioni di stile o altre cose.

Le caratteristiche di un attributo devono essere le seguenti:

* Un attributo deve avere uno spazio tra esso e l’elemento a cui si applica ( o l’attributo precedente se l’elemento ha più di un attributo
* Il nome dell’attributo seguito dal segno =
* Il valore dell’attributo tra apici

**Nota**: Valori di attribute semplici che non contengano spazi (o uno dei seguenti caratteri  " ' ` = < > ) possono rimanere senza apici, ma si raccomanda di quotarli tutti, dato che rende il codice più leggibile e consistente.

Gli elementi possono essere innestati (ovvero l’uno dentro l’altro)

<p>My cat is <strong>very</strong> grumpy.</p>

Qui la parola very viene enfatizzata.

Alcuni elementi inoltre possono essere vuoti, ovvero non prevedono contenuto.

Un esempi tipico è il tag <img>

<img src="images/firefox-icon.png" alt="My test image">

Ci sono due attributi , ma non c’è il tag di chiusura </img> e nessun contenuto.

Questo perché un’immagine non prevede un contenuto per essere visualizzata.

Anatomia di una pagina HTML

Gli elementi HTML da soli non servono a molto, devono essere combinati insieme per ottenere un documento html

Vediamo l’esempio di index.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>My test page</title>

</head>

<body>

<img src="images/firefox-icon.png" alt="My test image">

</body>

</html>

1. <!DOCTYPE html> è il tipo di documenti. Quando html era ancora giovane (1991/1992) i doctype erano stati pensati per agire da collegamento a un insieme di regole che una pagina HTML doveva seguire per essere considerata una buona pagina HTML. Questo significava controllo automatico degli errori e altre cose utili. Adesso di questo aspetto non frega più nulla a nessuno. Si utilizza comunque come artefatto storico da includere per far funzionare correttamente la pagina. Vedremo dopo come.
2. <html></html> è il tag di inizio di un documento html. E’ noto come l’elemento root.
3. <head></head> l’elemento head. Funge da contenitore di tutto per tutto quello che serve a una pagina HTML che non sia contenuto. Per esempio in head risiedono parole chiave, il titolo della pagina html, una descrizione, link a fogli di stile (css) che servono a presentare la pagina in un determinato modo e a script javascript per rendere la pagina dinamica.
4. <meta charset="utf-8"> - imposta l’insieme di caratteri utilizzato nella pagina. In questo caso si usa UTF-8 che è una codifica che contiene la maggior parte dei linguaggi esistenti al mondo. Non c’è ragione per non mettere questo tag e può evitare guai futuri.
5. <title></title> - il titolo della pagina. Appare nel browser quando la pagina è caricata. Viene anche utilizzato per dare un nome alla pagina quando viene messa tra i preferiti.
6. <body></body> - E’ il tag che ospita il contenuto vero e proprio della pagina: testo, video, audio, giochi e così via.

Immagini

Poniamo ancora una volta l’attenzione sul tag <img>

<img src="images/firefox-icon.png" alt="My test image">

Come abbiamo visto incorpora un’immagine nella nostra pagina nella posizione in cui appare il tag. Ottiene questo attraverso l’attributo src che contiene il percorso della nostra immagine.

Abbiamo anche aggiunto l’attributo alt (alternative) . In questo attributo si inserisce la descrizione testuale dell’immagine, per gli utenti che non possono vederla, di solito per le seguenti ragioni

* Hanno problemi di vista.
* Qualcosa è andato storto e l’immagine non si visualizza.

Il testo presente dentro alt deve fornire una breve descrizione dell’immagine che si vuole visualizzare.

Nel nostro esempio il testo “My test image” non va bene per nulla, per esercizio proviamo a cambiarlo.

Marcare il testo

Headings

Gli elementi headings servono per specificare che alcune parti del contenuto sono intestazioni ( o sottointestazioni).

Allo stesso modo in cui un testo ha un titolo , un eventuale sottotitolo, capitoli e paragrafi , un documento HTML fa la stessa cosa.

HTML ha 6 livelli di headings: da <h1> (il più grande) a <h6>

<h1>My main title</h1>

<h2>My top level heading</h2>

<h3>My subheading</h3>

<h4>My sub-subheading</h4>

Come esercizio aggiungiamo delle intestazioni alla nostra pagina.

Paragrafi

Come visto precedentemente i tag <p> contengono del testo.

<p>This is a single paragraph</p>

Esercizio: Subito sotto il tag <img> aggiungiamo un paragrafo con del testo

Liste

Molto contenuto del web è rappresentato da liste, HTML ha degli elementi dedicati per esse.

Marcare del contenuto come lista significa creare due elementi (la lista e i suoi elementi)

Esistono due tipi di lista: ordinata (ordered <ol> ) e non ordinata (unordered <ul>).

Ogni elemento di qualsiasi delle due liste è contrassegnato con il tag <li>

<p>At Mozilla, we’re a global community of</p>

<ul>

<li>technologists</li>

<li>thinkers</li>

<li>builders</li>

</ul>

<p>working together ... </p>

Esercizio: aggiungi una lista ordinata e una disordinata al tuo sito.

Links

I link sono fondamentali in una pagina html. Sono ciò che rendono il web…un web.

Per aggiungere un link si utilizza il tag <a>

Seguire i seguenti passi per creare un link:

1. Scegliere del testo
2. Incorporarlo nel tag <a></a>
3. Aggiungere l’attributo href al tag <a> che ci indica la destinazione a cui verremo portati dal link.

Es:

<a href="https://www.mozilla.org/en-US/about/manifesto/">Mozilla Manifesto</a>

Si ottengono dei risultati impredicibili se si omette la parte http o https (detta protocollo).

Sembra banale ma verifica sempre , una volta creato un link, che ti porti dove ti aspetti.

href potrebbe sembrare una scelta piuttosto oscura per un nome attributo, in prima istanza. Se hai problemi a ricordarlo , ricorda che sta per  ***h***ypertext ***ref***erence.

Esercizio: Aggiungi un link alla tua pagina

CSS

CSS (Cascading Style Sheet) è il codice che si utilizza per applicare degli stili alle pagine web.

Cos’è davvero CSS?

Come HTML. CSS non è un vero linguaggio di programmazione, e non è nemmeno un linguaggio di markup (di impaginazione, di marcatura).

E’ un linguaggio di fogli di stile.

Questo significa che permette di applicare degli stili in maniera selettiva agli elementi di una pagina HTML.

Per esempio, per selezionare tutti gli elementi paragrafo di una pagina HTML e mettere il testo in rosso si scrive così:

p {

color: red;

}

Proviamolo:

creiamo un nuovo file, incolliamo lo stile sopra e salviamo il file nella cartella styles con il nome di style.css

Apriamo poi il file index.html e incolliamo la seguente riga da qualche parte tra <head> e </head>

<link href="styles/style.css" rel="stylesheet">

Come dovrai aver visto , gli elementi paragrafo sono diventati di colore rosso.

Anatomia di un CSS

Vediamo in dettaglio lo stile applicato sopra:



L’intera struttura è chiamata rule set (abbreviata in rule)

Vediamo le singole parti:

**Selettore**: Il nome dell’elemento HTML all’inizio del rule set. Seleziona l’elemento (o gli elementi) a cui viene applicato lo stile. (in questo caso l’elemento p)

Per applicare lo stile a un altro elemento basta cambiare il selettore.

**Dichiarazione:** Una singola regola, come color: red; specifica a quale proprietà dell’elemento viene applicato lo stile.

**Proprietà:** Modi nei quali si può applicare uno stile a un dato elemento HTML. (in questo caso color è una proprietà dell’elemento p). In CSS, devi scegliere quali proprietà vuoi modificare nella tua regola.

**Valore della proprietà:** alla destra della proprietà dopo i due punti ( : ) e prima del punto e virgola ( ; ) abbiamo il valore della proprietà che è uno dei molti valori possibili. (rosso è uno dei tanti colori)

Altre importanti questioni sintattiche da notare:

* Ciascuna rule set (a parte il selettore) deve essere racchiusa da parentesi graffe.
* Dentro ciascuna dichiarazione, bisogna utilizzare il simbolo dei due punti per separare la proprietà dal suo valore
* All’interno di ciascuna rule set , bisogna utilizzare il punto e virgola per separare le diverse dichiarazioni.

Pertanto, per modificare più di una proprietà alla volta, basta elencare le varie dichiarazioni con un punto-e-virgola.

Es:

p {

color: red;

width: 500px;

border: 1px solid black;

}

Selezionare più di un elemento

Si possono selezionare più tipi di elementi a cui applicare la regola.

Basta separarli con una virgola:

p, li, h1 {

color: red;

}

Tipi differenti di selettori

Esistono diversi tipi di selettori. Fino a ora abbiamo visto solo i selettori elemento, che selezionano tutti gli elementi di un dato tipo nel documento HTML. Ma possiamo effettuare anche selezioni più precise e specifiche.

Vediamo una tabella con i selettori più comuni.

|  |  |  |
| --- | --- | --- |
| **Nome selettore** | **Cosa seleziona** | **Esempio** |
| Selettore elemento | Tutti gli elementi HTML del tipo specificato | p  Seleziona tutti i <p> |
| Selettore ID | L’elemento della pagina contrassegnato con uno specifico attributo di nome id. In una data pagina può esistere solo un id per elemento. | #my-id  Seleziona <p id="my-id"> oppure <a id="my-id"> |
| Selettore CLASSE | Gli elementi della pagina che hanno una certa classe.(in una pagina possono capitare più istanze della stessa classe) | .my-class  Seleziona <p class="my-class"> e  <a class="my-class"> |
| Selettore ATTRIBUTO | Gli elementi di una pagina con uno specific attributo. | img[src]  Seleziona <img src="myimage.png"> ma non  <img> |
| Selettore PSEUDO-CLASSE | Gli elementi specificati , ma solo quando in uno specifico stato (ad esempio quando ci si passa sopra con il mouse) | a:hover  Seleziona <a>,ma solo quando il puntatore del mouse è sopra il link. |

Caratteri e testo

Aggiungiamo qualche altra informazione e altre regole al nostro file style.css per rendere la nostra pagina html più piacevole.

Cominciamo dal tipo di carattere e dal testo.

Andiamo su google fonts e aggiungiamo il link nell’elemento head della nostra pagina html.

<link href="https://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet">

Cancelliamo la regola sull’elemento p e sostituiamola con quella riportata nella pagina di google font per quel font.

html {

font-size: 10px; /\* px means "pixels": the base font size is now 10 pixels high \*/

font-family: "Open Sans", sans-serif; /\* this should be the rest of the output you got from Google fonts \*/

}

Font-family è il tipo di carattere che si vuole utilizzare nella nostra pagina web.

Questa è una regola globale, ovvero applicata all’intero documento HTML, come si evince dal selettore utilizzato.

Tutti gli elementi ereditano quel font e quella dimensione.

**Nota**: Qualsiasi cosa in un document CSS tra  /\* e  \*/ è un **commento** **CSS**, che il browser ignora quando mostra la pagina. E’ utile scrivere commenti per annotrarsi quello che si sta facendo.

Adesso aggiustiamo la dimensione del font all’interno di elementi h1, li e p

Centriamo anche il testo dell’intestazione , e impostiamo l’altezza delle righe e lo spazio tra le lettere per rendere il contenuto più leggibile.

Possiamo anche aggiustare il numero di pixel per raggiungere il risultato desiderato.

BOXES

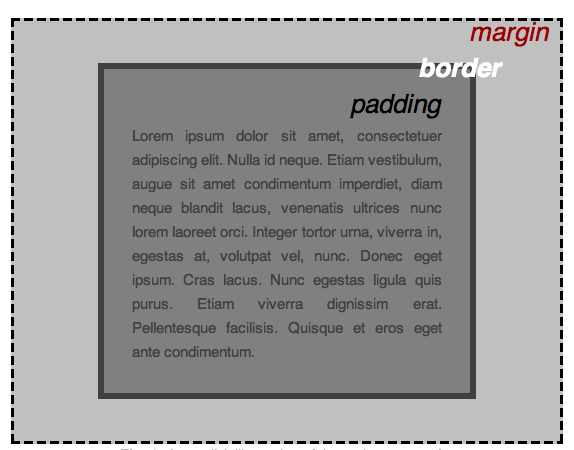
Una cosa che risulta evidente quando si scrivono dei file css è che la maggior parte riguarda i boxes (i riquadri) , impostare la loro dimensione , il loro colore, la loro posizione e così via.

La maggior parte degli elementi HTML presenti nella nostra pagina è pensabile come dei box impilati l’uno sull’altro



Non ci sorprende che il layout CSS è basato principalmente sul box model.

Ciascuno dei blocchi che prende spazio sulle nostre pagine ha delle proprietà come le seguenti:



* **padding**: lo spazio presente attorno al contenuto (ad esempio attorno al testo del paragrafo)
* **border**: la linea solida che si presenta appena fuori il padding
* **margin**: lo spazio attorno l’elemento esterno.

In questa sezione utilizzeremo anche:

* **width**: larghezza di un elemento
* **background-color**: il colore dello sfondo della pagina
* **color**: il colore del testo
* **text-shadow**: un’ombra vicino al testo dentro a un elemento
* **display**: la modalità di display di un elemento (la vedremo in seguito)

Cambiamo il colore di sfondo della pagina

html {

background-color: #00539F;

}

Aggiustiamo adesso il contenuto

body {

width: 600px;

margin: 0 auto;

background-color: #FF9500;

padding: 0 20px 20px 20px;

border: 5px solid black;

}

* *width: 600px;* — forza il body a essere largo sempre 600 pixel.
* *margin: 0 auto;* — Quando si impostano due valori su una proprietà come margin o padding, il primo valore influenza la parte superior e inferior dell’elemento (in questo caso vale 0) , e il secondo valore influenza la parte sinistra e destra (qui il valore vale auto, che è un valore speciale che divide lo spazio orizzontale disponibile in maniera equa tra sinitra e destra). Si possono utilizzare anche uno, tre o quattro valori.
* *background-color: #FF9500;* — come prima, questo influenza il colore di sfondo dell’elemento. Abbiamo utilizzato un arancio tendente al rosso per il body, opposto al blu scuro dell’elemento html. Per esercizio è consigliabile sperimentare.
* *padding: 0 20px 20px 20px;* — Abbiamo quattro valori impostati per il padding, per creare un pò di spazio attorno al nostro contenuto. In questo caso è stato scelto nessun padding per il top, 20pixel a sinistra, in basso e a destra. I valori impostano nell’ordine alto, destra , fondo, sinistra. Così come per margin, si possono anche usare, uno, due o tre valori.
* *border: 5px solid black;* — questo imposta un bordo largo 5 pixel , solido e nero al body.

Posizioniamo e applichiamo lo stile al titolo della nostra pagina

h1 {

margin: 0;

padding: 20px 0;

color: #00539F;

text-shadow: 3px 3px 1px black;

}

Si nota un enorme spazio tra il titolo e l’inizio della pagina. Questo avviene perché il browser applicano uno stile di default agli header , anche se si sono applicati dei css. Per togliere il gap basta mettere *margin: 0;*

Abbiamo poi messo 20 pixel di padding, e il colore dell’intestazione uguale a quello dello sfondo.

Abbiamo poi applicato un’ombreggiatura con text-shadow i cui valori hanno il seguente significato:

1. il primo valore in pixel indica lo scostamento orizzontale dell’ombra dal testo. Se negativo si sposta a sinistra.
2. Il secondo valore indica lo scostamento verticale dell’ombra dal testo: di quanto va giù. Se il valore è negativo va verso l’alto.
3. Il terzo valore indica il fattore di sfocamento dell’ombra. Più è grande più l’ombra è sfocata.
4. Il quarto valore indica il colore dell’ombra.

Centriamo infine l’immagine

Utilizziamo il trucco di prima per il body, con margin: 0 auto;

Ma facciamo qualcos’altro:

L’elemento body è a livello di blocco, che significa che prende spazio nella pagina e può avere margini e altri valori di spaziatura applicati a esso.

Le immagini invece sono elementi inline, ovvero non hanno tali proprietà.

Per applicare margini alle immagini dobbiamo dargli “dignità di blocco”, utilizzando display: block;

**Nota**: Le istruzioni precedent assumono che tu stia utilizzando un’immagine più piccolo della larghezza del body (600 pixels). Se l’immagine utilizzata è più grande , esce fuori dal body e occupa il resto della pagina. Per aggiustare la questione si può procede in due modi 1) ridurre la larghezza dell’immagine con un editor grafico , oppure 2)  dare una dimensione all’immagine impostando nel CSS la proprietà width dell’elemento <img> con un valore più piccolo (ad esempio 400px).

**Nota**: Non ti preoccupare se non è chiaro ancora display:block; e la distinzione tra block e inline. Studiando CSS più approfonditamente risulterà chiara.

JAVASCRIPT

Javascript è a tutti gli effetti un linguaggio di programmazione, a differenza di CSS e HTML.

Quando viene applicato a un documento HTML aggiunge interattività a un sito web.

Primo esempio (file “Hello World”)

Creiamo un file di nome **main.js** e mettiamolo nella cartella **scripts**.

Aggiungiamo nel nostro file html, subito prima del tag </body> la seguente riga

<script src="scripts/main.js"></script>

Questa riga fa sostanzialmente quello che faceva la riga inserita per il file css, aggiunge alla pagina HTML delle nuove caratteristiche, in questo caso del codice javascript.

Scriviamo nel file main.js le seguenti righe di codice

var myHeading = document.querySelector('h1');

myHeading.textContent = 'Hello world!';

Salviamo il file e vediamo il risultato.

**Nota**: La ragione per la quale abbiamo messo l’elemento  [<script>](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/script) vicino alla fine del file HTML è che HTML è caricato dal browser nell’ordine in cui appare nel file. Se il JavaScript è caricato prima e si vuole modificare l’HTML sottostante, si rischia che la modifica non funzioni, visto che il JavaScript sarebbe caricato prima dell’HTML su cui deve operare. Pertanto la migliore strategia è mettere JavaScript vicino alla fine della pagina HTML.

Il testo dell’intestazione della pagina (<h1>) è cambiato, grazie all’utilizzo di JavaScript. Si è ottenuta questa cosa usando una funzione che si chiama **querySelector()** per ottenere un riferimento all’intestazione che è stato salvato nella variabile myHeading.

Questo è molto simile a quanto è avvenuto per i selettori CSS: se vui fare qualcosa con un elemento, per prima cosa bisogna selezionarlo.

Dopo questo, si imposta la proprietà **textContent** della variabile **myHeading**  ( che rappresenta il contenuto dell’intestazione ) a “Hello World”.

**Nota**: Entrambe le caratteristiche utilizzate sopra sono parte delle API [Document Object Model (DOM)](https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model) , che permettono di manipolare documenti

Basi del linguaggio in breve

Definizioni variabili

Var MiaVariabile;

Il punto-e-virgola alla fine è necessario solo se ci sono più istruzioni sulla stessa riga.

E’ consigliato utilizzarlo sempre per chiarezza.

Il nome della variabile è CASE SENSITIVE ( si fa distinzione tra lettere maiuscole e minuscole)

Attenzione che le variabili possono contenere valori di diverso tipo.

|  |  |  |
| --- | --- | --- |
| **Variabile** | **Spiegazione** | **Esempio** |
| String | Una sequenza di testo. Per specificare che il valore è una stringa si mette tra apici | var myVariable = 'Bob'; |
| Number | Un numero, non ha apici che lo circondano | var myVariable = 10; |
| Boolean | Un valore True/False le parole true e false sono parole chiave in javascript e non vanno tra apici | var myVariable = true; |
| Array | Una struttura che permette di salvare più valori con un solo reference | var myVariable = [1,'Bob','Steve',10]; Si fa riferimento a ciascun elemento dell’array nel seguente modo: myVariable[0], myVariable[1], etc. |
| Object | Praticamente tutto. Qualsiasi cosa in javascript è un oggetto e può essere salvato in una variabile |  |

**Commenti:**

/\*

Everything in between is a comment.

\*/

// This is a comment

**Operatori:**

**(Da testare su console del browser)**

**Un operatore è un simbolo matematico che produce un risultato in base a due valori (o variabili).**

Addizione:

6 + 9;  
"Hello " + "world!";

Sottrazione, moltiplicazione e divisione

9 - 3;  
8 \* 2;   
9 / 3;

Assegnamento:

var myVariable = 'Bob';

Uguaglianza:

var myVariable = 3;  
myVariable === 4; (test che in questo caso ritorna false)

Not, Disuguaglianza

var myVariable = 3;  
!(myVariable === 3);

Per quanto riguarda la disuguaglianza il risultato è sostanzialmente lo stesso, cambia solo la sintassi.

Vediamo se myVariable non è uguale a 3. Il test ritorna false perché myVariable è uguale e tre:

var myVariable = 3;  
myVariable !== 3;

**Condizioni**

var iceCream = 'chocolate';

if(iceCream === 'chocolate') {

alert('Yay, I love chocolate ice cream!');

} else {

alert('Awwww, but chocolate is my favorite...');

}

**Funzioni**

Modalità per concentrare le funzionalità in un unico punto, richiamabile al bisogno.

Esempi

var myVariable = document.querySelector('h1');

alert('hello!');

Le funzioni document.querySelector() e alert() fanno parte di Javascript e sono pronte all’uso all’occorrenza.

Si possono definire le proprie funzioni:

function multiply(num1,num2) {

var result = num1 \* num2;

return result;

}

E richiamarle altrove:

multiply(4, 7);

multiply(20, 20);

multiply(0.5, 3);

proviamo le chiamate a console.

**Eventi**

L’interattività nelle pagine web si ottiene utilizzando gli eventi.

Gli eventi sono strutture di codice che sono in ascolto che qualcosa accada nel browser e reagiscono con del codice quando questo qualcosa accade.

L’evento più ovvio è l’evento click che viene emesso (fired) dal browser quando si fa click su qualche elemento della pagina.

Per dimostrare l’evento click scriviamo su console il seguente codice e poi facciamo click sulla pagina web corrente.

document.querySelector('html').onclick = function() {

alert('Ouch! Stop poking me!');

}

Esistono molti modi per legare un evento a un elemento. In questo caso abbiamo selezionato l’elemento html (ovvero tutta la pagina) e abbiamo associato la proprietà onclick dell’elemento a un gestore, che altro non è che una funzione senza nome che contiene il codice che deve essere eseguito al click

Da notare che il codice

document.querySelector('html').onclick = function() {}

equivale a

var myHTML = document.querySelector('html');

myHTML.onclick = function() {};

è soltanto abbreviato.

Applichiamo quanto visto alla nostra pagina html di esempio.

Cerchiamo su internet un’altra immagine che abbia dimensioni, se non uguali, almeno molto simili a quelle dell’immagine scelta inizialmente per la nostra pagina web.

Ovviamente salviamo la nuova immagine nella cartella **images** del nostro progetto

Scriviamo il seguente codice nel file main.js ( cancellando ciò che eventualmente è rimasto dall’esempio precedente)

var myImage = document.querySelector('img');

myImage.onclick = function() {

var mySrc = myImage.getAttribute('src');

if(mySrc === 'images/img1.xxx') {

myImage.setAttribute ('src','images/img2.xxx');

} else {

myImage.setAttribute ('src','images/img1.xxx');

}

}

Dove img1.xxx e img2.xxx sono le due immagini presenti nella cartella images (xxx è l’estensione dell’immagine )

Si salva il reference all’immagine presente nella pagina, nella variabile myImage.

Poi si associa la proprietà **onclick** dell’elemento immagine a una funzione anonima che “spiega” cosa dovrà accadere quando l’utente clicca sull’immagine.

Se l’immagine è attuale è la prima, al click si vede la seconda. Altrimenti si vede la prima.

**Aggiungere un messaggio di benvenuto personalizzato**

Aggiungiamo un altro pezzo di codice per personalizzare il messaggio di benvenuto nel titolo della pagina HTML. Il messaggio verrà acquisito dall’utente la prima volta che entra nel sito , e sarà poi persistente.

Per la persistenza utilizzeremo le API Web Storage. Aggiungeremo la possibilità di cambiare il nome dell’utente quando necessario.

Nella pagina HTML aggiungiamo un bottone subito prima del tag <script>

<button>Cambia utente</button>

Nel file main.js mettiamo il seguente codice proprio alla fine del file.

var myButton = document.querySelector('button');

var myHeading = document.querySelector('h1');

Abbiamo adesso il riferimento all’elemento <button> e all’elemento <h1> salvato in due variabili, myButton e myHeading rispettivamente.

Aggiungiamo nel file javascript la seguente funzione:

function setUserName() {

var myName = prompt('Inserisci il tuo nome.');

localStorage.setItem('name', myName);

myHeading.textContent = 'Mozilla is cool, ' + myName;

}

Adesso la funzione non fa ancora nulla, ma la aggiustiamo subito.

La funzione è composta da una funzione **prompt()** che è una funzione javascript che mostra una finestra che chiede dei dati all’utente. Il dato inserito dall’utente viene memorizzato in una variabile ( nel nostro caso *myName*) quando l’utente preme il taso OK

La successiva operazione da fare è quella di chiamare le API Web Storage , tramite localStorage, che permettono di salvare il nome assegnato alla variabile myName.

Chiamiamo la funzione setItem() di localStorage per effettuare questa operazione. Si associa all’etichietta name il contenuto della variabile myName.

Infine si inserisce nell’intestazione il messaggio personalizzato, tramite la proprietà textContent dell’elemento myHeading.

Non ci resta che attivare la funzione setUserName con il seguente frammento di codice

if(!localStorage.getItem('name')) {

setUserName();

} else {

var storedName = localStorage.getItem('name');

myHeading.textContent = 'Mozilla is cool, ' + storedName;

}

Se il localStorage non contiene l’oggetto name lo si inserisce chiamando setUserName. Altrimenti si prende il valore associato all’etichetta name e con esso si popola l’intestazione.

Infine, gestiamo l’evento onclick sul bottone button, quando l’utente vuole modificare il nome da far apparire nell’intestazione.

myButton.onclick = function() {

setUserName();

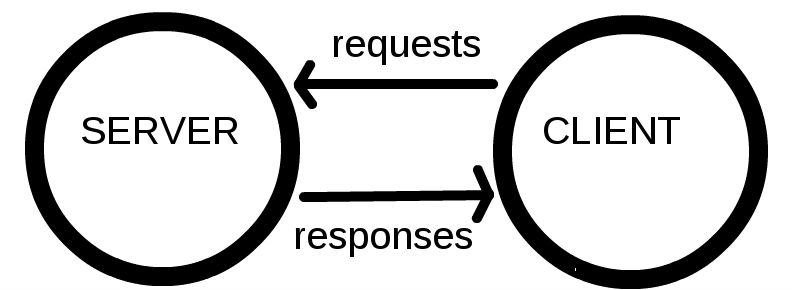
}

**Come funziona il web**

Client e server

I computer collegati alla rete sono chiamati client e server.

Vediamo un diagramma semplificato.



* I client sono i tipici dispositivi collegati alla rete (ad esempio il tuo computer collegato via WI-FI, il tuo telefonino collegato alla rete mobile) provvisti di software adatti a quei dispostivi (di solito browser web come Chrome e Firefox )
* I server sono computer che contengono pagine web, siti, app. Quando un client vuole accedere a una di queste risorse , una copia della pagina web è scaricata dal server sulla macchina client e visualizzata attraverso il web browser.

Ovviamente quanto detto non esaurisce l’argomento, ci sono molte altre parti in gioco e ora le vediamo.

Immaginiamo per ora che il web sia una strada. Da un lato della strada si trova il client , che è pensabile come casa tua. Dall’altro lato c’è il server, che può essere visto come il tuo negozio preferito in cui vuoi comprare qualcosa.

In aggiunta al client e al server, facciamo la conoscenza dei seguenti altri elementi:

1. **La tua connessione internet**: permette di spedire e ricevere dati dal web. Sostanzialmente è la strada che collega il client con il server.
2. **TCP/IP ( Transmission Control Protocolo / Internet Protocol)** : sono protocolli di comunicazione che stabiliscono come i dati debbano viaggiare in rete. E’ come il meccanismo di trasporto che permette di fare un ordine, andare al negozio e comprare i tuoi beni o servizi. Nel nostro esempio è paragonabile all’automobile o alla bicicletta.
3. **DNS** (Domain Name Server): sono come le rubriche dei siti web. Quando si digita un indirizzo internet nel proprio browser( <http://www.google.com>) il browser va a cercare nel DNS l’indirizzo reale del sito web. Il DNS deve sapere in quale server risiede il sito così da poter inviare messaggi HTTP (vedi dopo) nel luogo giusto. E’ come cercare l’indirizzo del negozio per accedervi
4. **HTTP (HyperText Transfer Protocol)**: è un **protocollo**(\*) applicativo che definisce un linguaggio per far parlare tra loro client e server. Questo è come il linguaggio utilizzato per fare ordini al negozio.
5. **File**: Un sito web è formato da file che rappresentano le varie parti di cui è composto in bene che si vuole acquistare. Tali file sono di due tipi principali

* **File di codice**: I siti web sono costituiti principalmente da file HTML, CSS, JavaScript e altri
* **Risorse**: nome collettivo per tutte le altre cose presenti su un sito web, come immagini, musica, video, documenti word o pdf

(\*Un **protocollo è un Sistema di regole che definiscono come scambiare dati all’interno o tre computer.**  Le comunicazioni tra dispositivi richiedono che i dispositivi accettino il formato dei dati che vengono scambiati. L’insieme delle regole che definiscono un formato è detto protocollo).

Cosa succede esattamente?

Quando si digita l’indirizzo di un sito web nel browser accadono le seguenti cose:

1. Il browser va nel server DNS e cerca l’indirizzo reale del sito
2. Il browser invia un messaggio di richiesta HTTP chiedendo una copia del sito al server. Questo messaggio e tutti gli altri dati inviati tra client e server viaggiano attraverso la tua connessione internet utilizzando TCP/IP
3. Se il server approva la richiesta del client , il server invia al client un messaggio “200 OK”, che significa “Certo , puoi vedere il sito, eccolo” e inizia a spedire il contenuto del sitoal client in piccole parti, detti pacchetti.
4. Il browser assembla i pacchetti inviati dal server e ti mostra il sito

Spiegazione DNS

Gli indirizzi web reali non sono carini da vedere, Sono numeri speciali del tipo 63.245.215.20

Questo è chiamato un indirizzo IP e rappresenta un luogo univoco nel web. Ma, non è semplice da ricordare per gli esseri umani. Questo è il motivo per il quale sono stati inventati i server DNS. Sono dei server speciali che associano il sito web che si digita nel browser con il corrispondente indirizzo IP.

I siti web possono essere raggiunti digitando il loro indirizzo IP, senza passare dal DNS.

Spiegazione pacchetti

Prima abbiamo utilizzato il termine “pacchetti” per descrivere il formato nel quale i dati sono spediti dal server al client. Cosa vogliamo dire qui? In sostanza, quando i dati sono spediti attraverso il web , vengono spediti in migliaia di pezzi, così che diversi utenti del web possano scaricare lo stesso sito allo stesso momento. Se il server inviasse tutto il blocco del sito in una volta sola , solo un utente alla volta potrebbe scaricarlo, rendendo il meccanismo del tutto inefficiente e poco divertente.