

HTML, CSS e Javascript!

Guida pratica per
principianti.



Alberto Reineri

www.albertoreineri.it

Prefazione

Ho deciso di scrivere questo libro perché mi sono reso conto che è molto difficile trovare delle guide semplici per un primo approccio al mondo dello sviluppo web in lingua italiana.

Tutti i corsi, gli ebook e i tutorial italiani che ho visto negli ultimi tempi erano sempre troppo teorici, troppo lunghi e troppo lenti.

Se voglio imparare a sviluppare voglio farlo ora, voglio iniziare a vedere i primi risultati in poco tempo, non dedicare ore a capire perché devo mettere il punto e virgola alla fine della riga. Certo anche questo è molto importante, ma posso impararlo anche successivamente.

Personalmente penso che per imparare occorre emozione, e per emozionarsi occorre mettere in pratica ciò che si impara e vedere che funziona.

Nulla è più gratificante di vedere il proprio codice funzionare così come lo si era pensato nella mente.

Quindi ho deciso di scrivere questo libro, nel quale non ci perderemo in chiacchiere ma andremo dritti al sodo, creando le nostre prime pagine web.

In queste pagine troverai *screenshot* e codici sorgenti per capire a fondo il funzionamento di ciò che faremo e mettere subito in pratica le tue capacità.

Non voglio assolutamente sottovalutare la teoria dello sviluppo web. È molto importante conoscere bene il perché un codice genera un risultato, ma credo che all'inizio sia più importante capire le nozioni fondamentali di pratica e vedere il proprio lavoro funzionare.

Se vedrai i primi risultati allora sicuramente approfondirai anche la teoria per capire meglio cosa stai facendo.

Ora basta con le chiacchiere e iniziamo a creare le nostre prime pagine web!

Indice

HTML, CSS e Javascript! – Guida pratica per principianti.	1
Prefazione	2
Indice	3
Le basi di HTML!	5
Introduzione	7
1. COSA SERVE PER INIZIARE	8
1.1. WEB BROWSER	8
1.2. EDITOR DI TESTO	8
2. CREIAMO IL NOSTRO PRIMO FILE HTML	9
3. STRUTTURA BASE	10
4. I TAG HTML	11
5. ESEMPI DI TAG:	12
6. SPORCHIAMOCI LE MANI	13
7. INSERIAMO UN'IMMAGINE	16
8. I CONTENITORI	17
8.1. DIV	18
8.2. SPAN	18
9. I FORM	19
10. Codice completo	21
Conclusione	25
Le basi di CSS	26
Introduzione	27
1. COME INSERIRE IL CSS IN UNA PAGINA HTML	27
1.1. INLINE CSS	28
1.2. CSS INTERNO	28
1.3. CSS ESTERNO	29
2. SINTASSI CSS	30
2.1. SELETTORE	31
2.2. PROPRIETÀ	31
2.3. VALORE	32
3. I COLORI	33
3.1. NOME DEL COLORE	33
3.2. ESADECIMALE	34

3.3.	RGB	34
4.	FONT	34
5.	CLASSI E ID	38
5.1.	ID	38
5.2.	CLASSI	39
6.	MARGIN E PADDING	40
7.	CONTENITORE	42
8.	IMMAGINE COME SFONDO	43
9.	CODICE COMPLETO:	44
	Le basi di Javascript!	51
	Introduzione	52
1.	INSERIRE JAVASCRIPT NELL'HTML	53
1.1.	JAVASCRIPT INTERNO	53
1.2.	JAVASCRIPT ESTERNO	54
2.	UN PO' DI TEORIA	55
2.1.	ALERT	55
2.2.	CONSOLE.LOG	55
2.3.	COMMENTI	56
2.4.	VARIABILI	56
2.5.	FUNZIONI	58
2.6.	PARAMETRI	59
2.7.	IF ELSE	59
2.8.	EVENTI	60
3.	CREIAMO IL NOSTRO PRIMO EFFETTO	61
4.	ANIMAZIONE DI UN COMPONENTE	63
5.	CODICE COMPLETO:	64

Le basi di HTML!

Introduzione

L'HTML è la base del web, c'è dappertutto! Ogni pagina internet che visualizzi ha del codice HTML al suo interno. Se vuoi diventare uno sviluppatore web quindi la prima cosa da fare è imparare per bene l'HTML!

L'HTML non è proprio un linguaggio di programmazione, ma è un **linguaggio di markup**, infatti HTML è l'acronimo di *HyperText Markup Language*.

Ciò significa che l'HTML non fa operazioni di calcolo, ma sostanzialmente indica al browser **come “montare” la pagina**, cosa posizionare e come posizionarlo.

Insieme vedremo le basi di questo linguaggio e inizieremo a creare la nostra prima pagina web.

Questa è una guida pratica, quindi ti suggerisco di leggerla mentre sei al computer, in modo da mettere in pratica subito ciò di cui parliamo.

Iniziamo!

1. COSA SERVE PER INIZIARE

Per iniziare a scrivere codice HTML **non servono super computer** né programmi pesanti e costosi.

Se vuoi iniziare a sviluppare contenuti per il web inizialmente **ti bastano 2 cose:**

- web browser
- editor di testo

1.1. WEB BROWSER

Sviluppando qualcosa che sarà fruibile attraverso un browser, **il browser è fondamentale**. Ne esistono veramente molti e tutti validi, ma il mio consiglio è di utilizzare [Google Chrome](#).

Se non sei un fan del browser di google eccoti alcune alternative:

- [Microsoft Edge](#) (Windows)
- [Safari](#) (Mac)
- [Mozilla Firefox](#)

1.2. EDITOR DI TESTO

L'altro software **fondamentale** per sviluppare per il web è un editor di testo.

Un editor di testo è un programma che **consente di scrivere il codice**. Si potrebbe utilizzare banalmente il classico editor di testo del sistema operativo (Blocco note o Text edit), ma fortunatamente esistono software dedicati allo sviluppo che rendono la scrittura del codice molto più semplice.

Il mio consiglio è di utilizzare [**VS Code**](#), a mio avviso in questo momento è il migliore in assoluto.

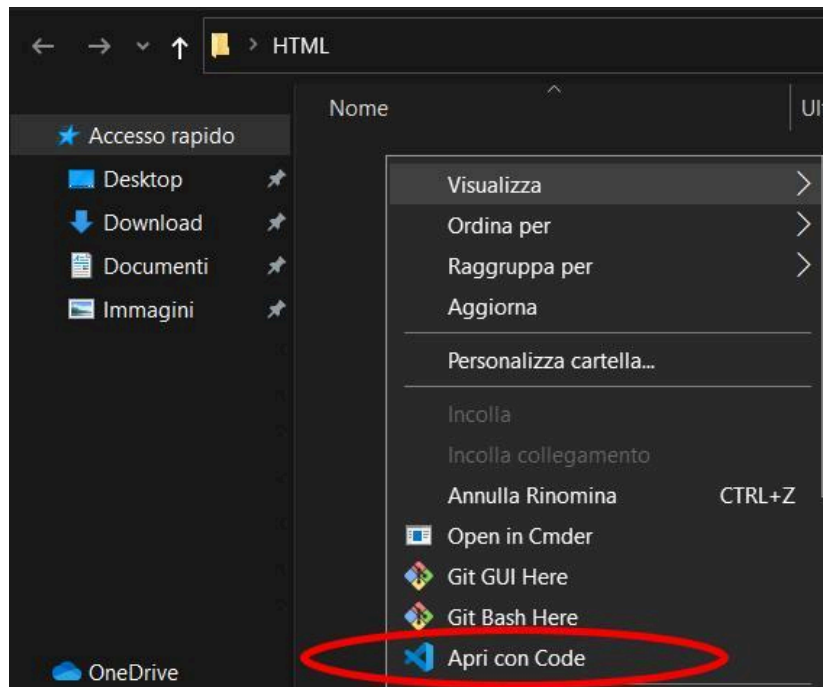
Anche qui hai comunque molta scelta! Ecco alcune delle migliori **alternative** a VS Code:

- [Sublime Text](#)
- [Atom](#)
- [Brackets](#)
- [Notepad++](#)

2. CREIAMO IL NOSTRO PRIMO FILE HTML

Ora che hai scaricato un browser e un editor di testo, possiamo creare il nostro primo file HTML.

Creiamo una **cartella** sul desktop chiamata "**HTML**". Ora apriamo questa cartella con **VS Code**, facendo click con il tasto destro del mouse all'interno della cartella e cliccando "**Apri con Code**":



Ora possiamo creare il nostro file con VS Code.

Clicchiamo “**CTRL+N**” per creare un nuovo file e poi “**CTRL+S**” per salvarlo, con il nome “***index.html***”.

Tutti i file HTML devono avere estensione .html, cioè finire con “.html”, questo farà capire al browser il tipo di file che sta leggendo.

Puoi creare i file anche tramite il menù in alto, cliccando su “File-New File” oppure con l’icona specifica nella barra laterale sulla sinistra.

PERFETTO! Abbiamo creato il nostro primo file HTML!

3. STRUTTURA BASE

Ogni pagina HTML è diversa, ma tutte hanno una **struttura base comune**, uno ***scheletro*** sul quale sono costruite.

VS Code ci permette di creare questo scheletro in maniera **semplicissima e molto veloce**.

Ci basterà aprire il file, inserire un **punto esclamativo** e cliccare il tasto **"tab"**. In questo modo VS Code creerà la struttura base della nostra pagina HTML in automatico.

Se hai fatto questa operazione dovresti vedere comparire questo codice all'interno del file:

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Document</title>

</head>

<body>


</body>

</html>
```

Questo è lo **scheletro** di ogni pagina HTML. Andiamo ad analizzarlo!

4. I TAG HTML

L'HTML è un linguaggio basato sui **tag**.

Ogni tag indica una **tipologia di contenuto**.

Struttura base: <nometag>contenuto</nometag>

Qualsiasi contenuto è sempre inserito all'interno di un **tag**, che **indica al browser come trattare quel tipo di contenuto**.

Generalmente i tag hanno un inizio e una fine, il tag di fine inizia con uno slash(/).

Esistono però alcuni tag senza tag di chiusura, come il tag
, che è utilizzato per andare a capo.

5. ESEMPI DI TAG:

```
<html></html>
```

Questi tag indicano dove inizia e dove finisce la pagina HTML. Tutto il contenuto va inserito fra questi due!

```
<head></head>
```

Questo tag permette di inserire delle informazioni relative alla pagina, come il titolo, gli stili da inserire, gli script etc. (Questo ti sarà più chiaro man mano che andrai avanti)

```
<body></body>
```

All'interno di questi tag c'è il vero e proprio contenuto della pagina

I titoli sono inseriti dentro i tag heading, che vanno dall'1 al 6, in ordine di importanza. Il titolo della pagina deve essere inserito fra i tag `<h1></h1>`, il sottotitolo `<h2></h2>` e così via.

```
<br>
```

Questo tag indica al browser di andare a capo.

```
<!-- Questo è un commento -->
```

In qualsiasi tipo di codice è molto importante inserire i commenti. Questi permettono di inserire delle note all'interno del codice, per poter capire meglio cosa si sta scrivendo o per inserire delle frasi rivolte ai colleghi etc.

```
<ul>

  <li>Questo è un item di un elenco</li>

  <li>Questo è un altro item</li>

</ul>
```

Il tag `` permette di inserire un elenco. Per inserire un elenco numerato c'è il tag `` (Unordered List e Ordered List).

Ogni elemento di un elenco deve essere inserito con il tag `` (List Item)

6. SPORCHIAMOCI LE MANI

Iniziamo ora a **modificare lo scheletro** della nostra pagina HTML.

Iniziamo a modificare la **lingua**, modificando "en" con "it" nella **riga 2** del nostro file.

La riga 2 sarà quindi così:

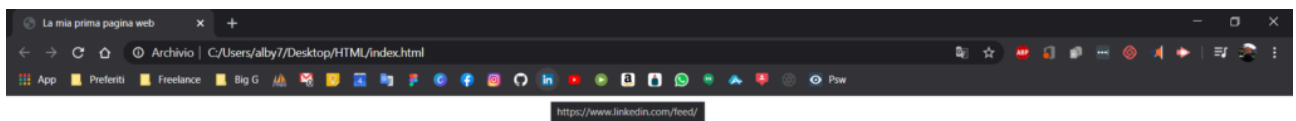
```
<html lang="en">
```

Ora andiamo sulla riga 6 e modifichiamo il titolo, nel tag **<title>**.

Chiamiamo questa pagina "**La mia prima pagina web**"

```
<title>La mia prima pagina web</title>
```

Ora salviamo il file e apriamolo, semplicemente aprendo la cartella "HTML" e facendoci doppio click sopra.



Ci troveremo di fronte una **pagina completamente bianca**, perché non abbiamo ancora inserito nessun codice nel contenuto.

Possiamo vedere però che il nome della scheda in alto è "**La mia prima pagina web**". Questo è il **<title>** della nostra pagina.

Ora inseriamo un po' di contenuto.

Andiamo **fra i tag <body></body>** e inseriamo questo:

```
<h1>La mia prima pagina web</h1>
```

```
<p>Benvenuto nella mia prima pagina web!</p>

<br><!-- questo è un a capo-->

<p>Questo è il secondo paragrafo della mia prima pagina web</p>

<br>

<h2>Elenco</h2>

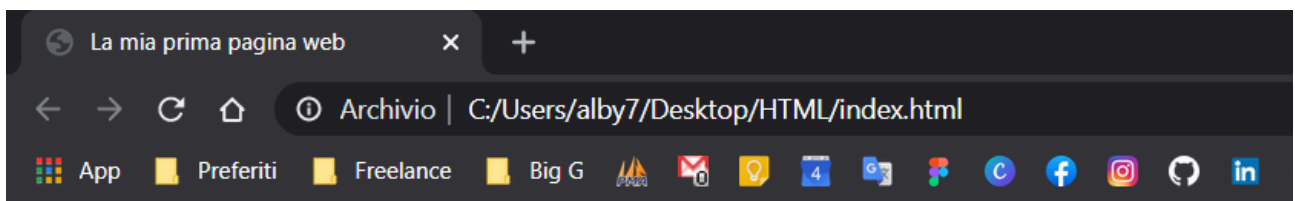
<ul>

  <li>Primo Item</li>

  <li>Secondo Item</li>

</ul>
```

Ora salva la pagina e aggiornala nel browser, vedrai comparire del contenuto!



La mia prima pagina web

Benvenuto nella mia prima pagina web!

Questo è il secondo paragrafo della mia prima pagina web

Elenco

- Primo Item
- Secondo Item

Non è difficile da capire, ogni tag spiega se stesso.

CONGRATULAZIONI!

Hai appena creato la tua prima pagina web!

Ma addentriamoci ancora un po' nell'HTML.

7. INSERIAMO UN'IMMAGINE

Per inserire un'immagine in una pagina HTML bisogna utilizzare il tag ****, con alcuni attributi.

Gli **attributi** forniscono **informazioni aggiuntive** ai tag html. Per esempio il tag **** indicherà al browser di inserire un'immagine, ma quale immagine? A questa domanda rispondiamo con l'attributo **"src"**, cioè la sorgente da cui il browser può attingere per inserire l'immagine.

Esempio:

```

```

In questo esempio il browser inserirà l'immagine foto.jpg presente nella cartella **"immagini"**.

Proviamo ora ad inserire un'immagine nella nostra pagina.

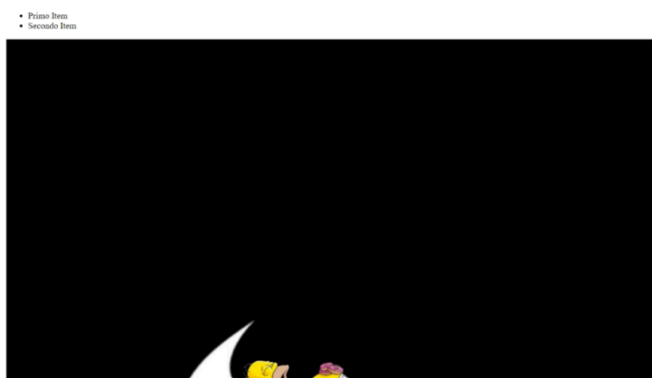
Andiamo nella nostra cartella "**HTML**" sul desktop e creiamo una cartella chiamata "**img**", all'interno di questa cartella inseriamo ora una qualsiasi immagine in formato **JPG**.

Adesso **richiamiamo l'immagine nel nostro file *index.html***, in questo modo:

```

```

Ora salviamo il file e aggiorniamo il browser.



Vediamo che l'immagine viene visualizzata nella nostra pagina html.

Però è un po' **troppo grande!** Almeno nel mio caso, questo dipende dalle dimensioni dell'immagine.

Per visualizzare l'immagine in modo più carino possiamo aggiungere un altro attributo al nostro tag ``: **l'attributo height o width**

```

```

In questo modo sto **impostando la larghezza dell'immagine a 200px**. Ed ecco che si vede tutto decisamente meglio.

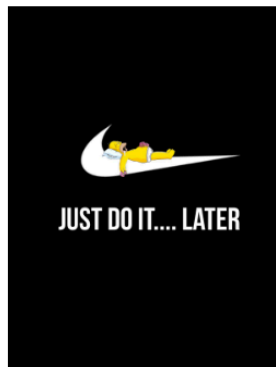
La mia prima pagina web

Benvenuto nella mia prima pagina web!

Questo è il secondo paragrafo della mia prima pagina web

Elenco

- Primo Item
- Secondo Item



In questo modo posso **ridimensionare** l'immagine.

Ora **andiamo ancora più a fondo** nell'html.

8. I CONTENITORI

Nelle nostre pagine html possiamo inserire dei **contenitori**, nei quali inserire del contenuto. Questi sono molto utili per **suddividere le pagine e gestire i contenuti al meglio**.

8.1. DIV

Un primo tipo di contenitore è il tag **<div>**. Questo crea una **sezione** nella pagina. È un **block element**, cioè il contenuto dopo questo tag è inserito **a capo**.

8.2. SPAN

Lo **** è un contenitore ma **inline**, cioè **non va a capo** dopo di esso.

Se voglio creare un quadrato verde nel sito dovrò utilizzare un `<div>`, se invece voglio colorare una parola di rosso allora userò il tag ``.

Eccoti un esempio:

Aggiungi questo codice a ***index.html***

```
<div style="background-color: green;">
```

Questo è un contenitore con sfondo verde

```
</div>
```

```
<p>
```

Questo è un paragrafo con del testo inserito a caso. In questo testo voglio `colorare` una parola di rosso

```
</p>
```

ed ecco il **risultato**

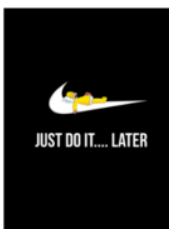
La mia prima pagina web

Benvenuto nella mia prima pagina web!

Questo è il secondo paragrafo della mia prima pagina web

Elenco

- Primo Item
- Secondo Item



Questo è un contenitore con sfondo verde

Questo è un paragrafo con del testo inserito a caso. In questo testo voglio **colorare** una parola di rosso

Per inserire i colori ho utilizzato l'attributo "style", che permette di inserire del codice CSS all'interno dell'HTML, ma lo vedremo meglio nella [guida al CSS](#).

9. I FORM

Un altro elemento molto importante di una pagina web è il **form**.

Navigando online avrai compilato moltissime volte dei **moduli**, che siano di contatto, di prenotazione etc.

Per inserire un form occorre utilizzare il tag **<form></form>** e al suo interno inserire **le tipologia di input** richieste.

Esempio di form:

```
<form>

  <input type="text" placeholder="Nome">

  <br><br>

  <input type="text" placeholder="Cognome">

  <br><br>

  <select name="select" id="">

    <option value="0">Opzione 1</option>
```

```
<option value="1">Opzione 2</option>

<option value="2">Opzione 3</option>

</select>

<br><br>

<textarea name="" id="" cols="30" rows="10" placeholder="Inserisci il
testo qui."></textarea>

<br><br>

<input type="checkbox" name="privacy" value="0">Accetto la Privacy
Policy

<br><br>

<button>Invia</button>

</form>
```

Prova a inserire questo codice in ***index.html***, salvare e aggiornare.

Vedrai comparire dei campi compilabili.

Questi campi sono:

- input di tipo text nel caso del nome e del cognome.
- select nel caso del menù a tendina
- textarea nel caso dell'area di testo
- input di tipo checkbox per accettare la privacy
- button per il bottone di invio

Premendo sul tasto “**Invia**” non succederà nulla. Per far svolgere un’azione alla nostra pagina html occorre integrarla con altri linguaggi. Ricordi che **I’HTML è solamente un linguaggio di markup**, non di programmazione.

10. Codice completo

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>La mia prima pagina web</title><!-- Il titolo della pagina che appare
nella scheda del browser -->

</head>

<body>

    <!-- Titolo -->

    <h1>La mia prima pagina web</h1>

    <!-- Paragrafo -->

    <p>Benvenuto nella mia prima pagina web!</p>

    <br><!-- questo è un a capo-->
```

```
<!-- Sottotitolo -->
```

```
<h2>Sottotitolo</h2>
```

```
<p>Questo è il secondo paragrafo della mia prima pagina web</p>
```

```
<br>
```

```
<h2>Elenco</h2>
```

```
<!-- Elenco -->
```

```
<ul>
```

```
  <li>Primo Item</li> <!-- Item di un elenco -->
```

```
  <li>Secondo Item</li>
```

```
</ul>
```

```
<!-- Immagine -->
```

```

```

```
<!-- DIV: block element -->
```

```
<div style="background-color: green;">
```

```
  Questo è un contenitore con sfondo verde
```

```
</div>
```

```
<!-- SPAN: inline element -->
```

```
<p>
```

Questo è un paragrafo con del testo inserito a caso. In questo testo
voglio

```
<span style="color:red">colorare</span> una parola di rosso
```

```
</p>
```

```
<!-- FORM -->
```

```
<form>
```

```
<!-- Casella di testo -->
```

```
<input type="text" placeholder="Nome">
```

```
<br><br>
```

```
<input type="text" placeholder="Cognome">
```

```
<br><br>
```

```
<!-- Menù a tendina -->
```

```
<select name="select" id="">
```

```
<option value="0">Opzione 1</option>
```

```
<option value="1">Opzione 2</option>
```

```
<option value="2">Opzione 3</option>
```

```
</select>
```

```
<br><br>

<!-- Area di testo -->

    <textarea name="" id="" cols="30" rows="10" placeholder="Inserisci il
testo qui."></textarea>

<br><br>

<!-- Checkbox-->

    <input type="checkbox" name="privacy" value="0">Accetto la Privacy
Policy

<br><br>

<!-- Bottone -->

<button>Invia</button>

</form>

</body>

</html>
```

Ora puoi iniziare a smanettare un po' con i tag che hai imparato, provando a **creare e modificare qualche pagina HTML!**

E ricorda, se hai bisogno di aiuto scrivimi a help@albertoreineri.it, insieme possiamo vedere come risolvere i tuoi bug.

Conclusione

Per questa prima parte è tutto. Hai imparato le basi del linguaggio HTML, ora non ti resta che metterle in pratica.

Nella programmazione il modo migliore di imparare è provare provare e provare. La pratica e la costanza sono le componenti fondamentali per poter diventare un buon sviluppatore, perciò mettiti al pc ed inizia a creare qualche pagina HTML, utilizzando i vari tag che abbiamo visto nei capitoli precedenti, provando a mischiare l'ordine e creare strutture e layout differenti.

Ti lascio ancora **il codice per intero della nostra *index.html*** con i **commenti** che spiegano cosa fa ogni cosa. Puoi copiarlo e incollarlo dove vuoi!

Le basi di CSS

Introduzione

Il CSS è il linguaggio di **formattazione** del web. Sta per *Cascading Style Sheets* ed è utilizzato per **assegnare uno stile alle pagine html**.

Ha una sintassi specifica e permette di separare l'html dal suo stile, mantenendo così il **codice pulito ed ordinato**.

Come l'HTML, anche il css **non è un linguaggio di programmazione**, è un linguaggio utilizzato per creare i layout delle pagine web. Consente di gestire gli spazi, modificare i colori, creare i layout e tutto ciò che ha a che fare con la parte grafica di un contenuto web.

Questo articolo è una continuazione del [corso intensivo per HTML](#), che puoi trovare [qui](#).

Se ti perdi durante l'articolo sul [fondo di questo](#) articolo potrai trovare il [codice](#) di tutto ciò che andremo a creare.

1. COME INSERIRE IL CSS IN UNA PAGINA HTML

Il CSS da solo quindi non serve a nulla, ma **deve essere inserito in una pagina html**.

Esistono **3 modi** per inserire del codice CSS in una pagina HTML

- **Inline CSS**
- **CSS Interno**
- **CSS Esterno**

1.1. INLINE CSS

Consente di inserire del codice CSS **direttamente all'interno del codice HTML**.

Con questo metodo **i linguaggi HTML e CSS restano mischiati insieme**. Un esempio di questa tipologia di CSS è quello che abbiamo inserito nel corso intensivo di HTML, quando abbiamo impostato lo sfondo verde al div, o il rosso alla parola nello span.

Esempio:

```
<div style="background-color:green>Ciao Mondo</div>
```

Sebbene sia molto veloce da applicare, **è il modo peggiore** per inserire del codice CSS.

Mischiare i linguaggi di programmazione non è mai un bene, è meglio imparare fin da subito che l'ordine è una caratteristica fondamentale per un buon sviluppatore.

Vediamo quindi gli altri metodi.

1.2. CSS INTERNO

Questo metodo consiste nell'inserire il codice CSS **all'interno dell'head** della pagina HTML.

In questo modo il CSS è all'interno della pagina HTML ma **non in mezzo al contenuto HTML**. È una scelta sicuramente migliore rispetto all'Inline CSS ma non ancora ottimale.

Per inserire del CSS interno occorre andare **fra i tag <head></head>** e indicare che stiamo per scrivere del codice CSS, in questo modo:

```
<style type="text/css">
```

```
</style>
```

All'interno del tag `<style>` possiamo inserire il codice CSS.

Esempio:

```
<style type="text/css">
```

```
  h1{
```

```
    color:red
```

```
  }
```

```
</style>
```

In questo caso **tutti gli h1 della pagina saranno rossi**.

1.3. CSS ESTERNO

Questo è **il modo migliore** e più efficiente di gestire i file CSS.

Consiste nel **creare un file esterno**, che deve avere estensione `.css`, e **richiamarlo nell'head della pagina HTML**.

In questo modo **i linguaggi sono ben separati**, ognuno nel suo file.

Esempio:

Andiamo nella nostra cartella **"HTML"** sul desktop e apriamola con **VS Code** (se non sai di cosa sto parlando dai un'occhiata al [corso intensivo di HTML](#))

Ora **creiamo un nuovo file** (CTRL+N) e salviamolo (CTRL+S) con il nome **"style.css"**.

In questo file possiamo inserire questo:

```
h1{  
  
  color:red;  
  
}
```

Ora **salviamo "style.css"** e apriamo **"index.html"**.

Andiamo **nell'head** di "index.html" e inseriamo questo sotto al title:

```
<link rel="stylesheet" href="style.css">
```

Se apri il file **"index.html"** nel tuo browser noterai che **l'h1 ora è rosso**.

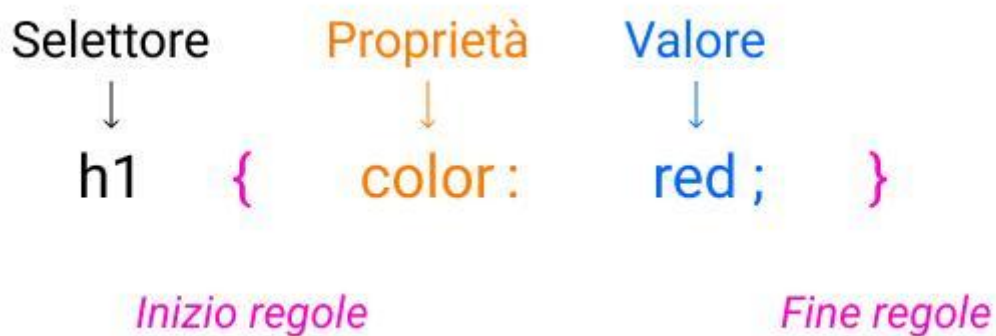
Questi sono **i tre metodi** per inserire del codice CSS in una pagina HTML. Come avrai potuto capire **il metodo migliore è il terzo**, ma ci sono alcune occasioni nelle quali può essere utile applicare uno dei primi due metodi.

Ora che abbiamo imparato a inserire il CSS in una pagina HTML iniziamo a parlare proprio di CSS!

2. SINTASSI CSS

Il CSS è un linguaggio con una **sintassi specifica**. Se si commette un **errore** nella scrittura della sintassi il codice **non funzionerà**.

Ecco uno **schema** che raccoglie gli **elementi** del linguaggio CSS:



2.1. SELETTORE

Indica l'**oggetto a cui applicare lo stile**. Questo può essere un tag html, come <h1>, <p> etc., oppure una classe o un id (approfondiremo classi e id fra poco).

2.2. PROPRIETÀ

Una proprietà è una **regola che si applica al selettore**. Il CSS ha moltissime proprietà, man mano che imparerai questo linguaggio ne scoprirai sempre di più. Fortunatamente VS Code offre una lista di tutte le proprietà, visibile premendo "CTRL+Space Bar".

Nell'esempio di prima la proprietà è "**color**", altre fra le più utilizzate sono "background-color", "margin", "padding", "border" etc.

2.3. VALORE

È il **valore da assegnare alla proprietà**. Può essere un colore, un numero in pixel o altro ancora. Imparerai ad utilizzare i valori corretti man mano che utilizzerai il CSS.

Proprietà e Valore devono essere racchiusi in una **parentesi graffa**.

Al termine di ogni Valore occorre inserire un **"punto e virgola"**.

Esempio:

```
h1{  
  
  color:red;  
  
}
```

- **h1** è il **selettore**
- dopo il selettore apro la **graffa**
- **color** è la **Proprietà**
- **red** è il **Valore**
- alla fine del valore inserisco il **punto e virgola**
- chiudo la **graffa**.

Dopo questa piccola introduzione teorica **iniziamo a fare sul serio!**

INIZIAMO A SPORCARCI LE MANI

Andiamo nel nostro file "style.css" e iniziamo a scrivere del codice CSS!

Iniziamo con il dare uno stile generale alla nostra pagina, utilizzando il selettore "body", in questo modo:

```
body{  
  
    color:#444;  
  
    background-color:#f2f2f2  
  
}
```

In questo modo abbiamo inserito un colore grigio scuro al testo e un bianco leggermente sporco allo sfondo.

3. I COLORI

Nel codice CSS ci sono **vari modi con i quali inserire i colori**:

- Nome del colore
- Esadecimale
- RGB

3.1. NOME DEL COLORE

Questo modo consente di indicare il colore semplicemente scrivendo il **nome del colore in inglese**.

Per esempio se voglio un colore bianco basterà scrivere "white", e così via.

Esempio:

```
color:blue;
```

3.2. ESADECIMALE

Il colore è indicato utilizzando un codice esadecimale, chiamato anche **Hex code**. Per utilizzare questo metodo occorre inserire un # seguito dal codice a 6 cifre. È di gran lunga il metodo più utilizzato.

Esempio:

```
color:#f4f4f4;
```

3.3. RGB

Consiste nell'indicare il colore utilizzando il **metodo RGB**. È possibile anche utilizzare delle trasparenze con l'RGBA.

Esempi:

RGB:

```
rgb(243, 163, 44)
```

RGBA:

```
rgba(243, 163, 44,.7)
```

In questo caso il ".7" indica che la trasparenza sarà al 70%.

4. FONT

Una delle prime cose da fare quando si personalizza un layout è **scegliere un bel font**.

Per inserire un font in un sito web occorre **importarlo**, per essere certi che qualsiasi utente su qualsiasi computer visualizzi il font corretto.

Fortunatamente **google** mette a disposizione moltissimi **font** in maniera **gratuita** e semplicissima da utilizzare. Vediamo come.

Per prima cosa andiamo su **google fonts**, a questo link: <https://fonts.google.com/>

Qua possiamo cercare il font che più ci piace. In questa guida utilizzeremo il **"Source Sans Pro"**.

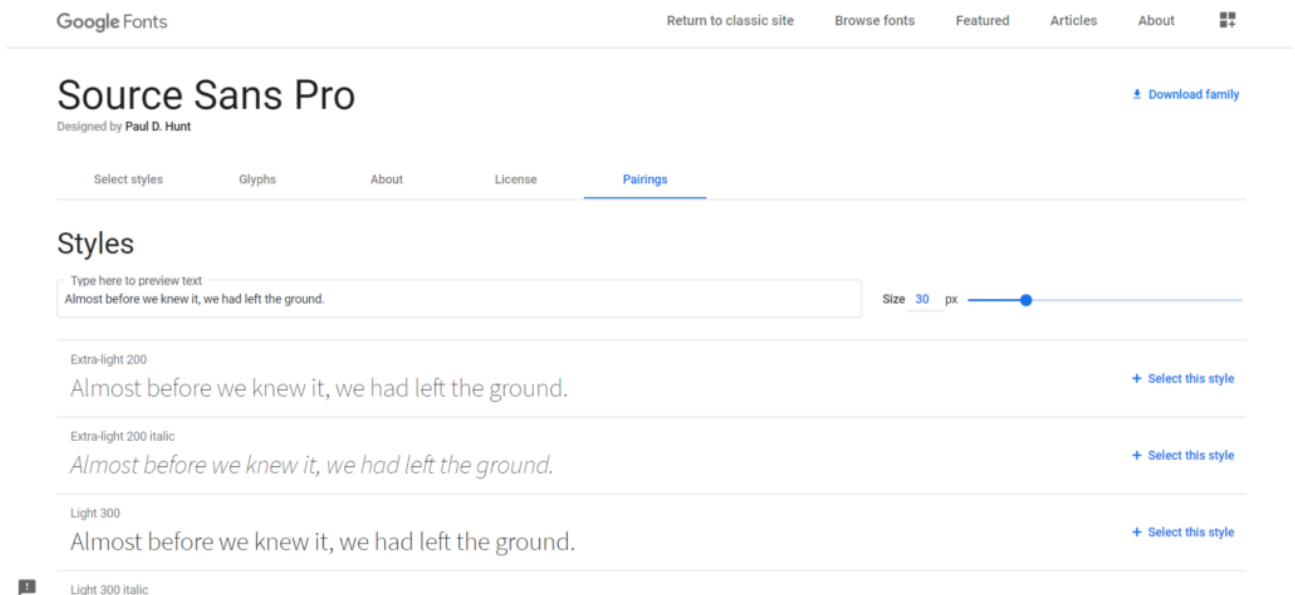
Inseriamo quindi **"Source Sans Pro"** nella barra di ricerca di Google Fonts

Google Fonts

source sans

e lo **selezioniamo**.

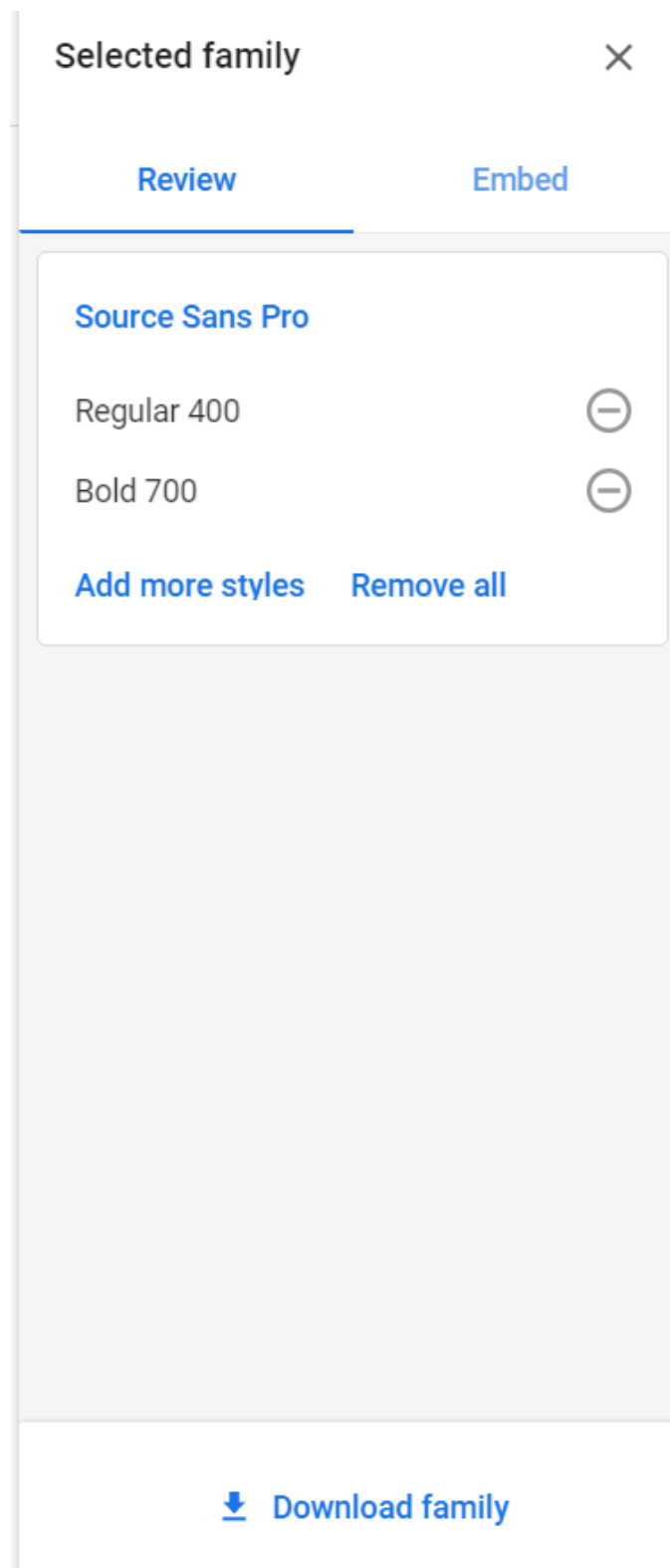
Ora ci troveremo di fronte ad una schermata come questa:



Sulla destra possiamo cliccare su **" + Select this style"** in corrispondenza del carattere che vogliamo. Possiamo selezionarli tutti per avere tutte le variabili possibili del font, ma per ottimizzare il tempo di caricamento della pagina è meglio selezionare solo l'essenziale.

In questa guida selezioniamo solo il “**regular 400**” e il “**bold 700**”.

Ora si aprirà sulla destra una finestra come questa:



Qua clicchiamo su “**Embed**” e successivamente su “**@import**”

`<link>` `@import`

```
<style>
@import url('https://fonts.goog
leapis.com/css2?family=Source+S
ans+Pro:wght@400;700&display=sw
ap');
</style>
```

Adesso possiamo copiare il contenuto fra `<style>` e `</style>` e incollarlo nel nostro **"style.css"**, cancellando tutto il resto.

Ora aggiungiamo questo codice:

```
body{

    font-family: 'Source Sans Pro', sans-serif;

    font-size: 22px;

    font-weight: 400;

    font-style: normal;

    line-height: 35px;

}
```

In questo modo abbiamo impostato **"Source Sans Pro"** come **font primario** del sito.

Ecco cos'altro abbiamo impostato:

- **Font-size** indica la **dimensione** del font, che abbiamo settato a 22 pixel.

- **Font-weight** indica lo **spessore** del font, in questo caso è settato come regolare. In questo campo possiamo utilizzare sia i numeri da 100 a 900, sia il nome, da "lighter" a "bolder". Logicamente occorrerà importare queste dimensioni da google fonts, per il momento abbiamo importato solo il 400 e il 700.
- **Font-style** indica lo **stile** del font, in questo caso è normale. Puoi inserire per esempio "italiac" per avere un font in corsivo.
- **Line-height** indica l'**altezza** del font, lo spazio fra le righe, in questo caso impostato a 35 pixel.

Prova a **salvare** il foglio di stile e **aggiornare** la pagina, vedrai che **il testo sarà cambiato!**

5. CLASSI E ID

Come abbiamo già accennato poco fa, è possibile impostare delle classi e degli id ai tag html, in modo da poterli **raggruppare** alcune regole di css.

Classi e id sono **attributi** che possiamo **aggiungere ai tag html** per distinguerli fra loro.

5.1. ID

Un **id** è un **attributo univoco**, va utilizzato nel caso ci sia un elemento particolare che **non si ripeterà** mai. Se per esempio voglio che un titolo sia giallo, solo quel titolo, posso dargli un id particolare.

Per indicare un id nel CSS occorre farlo precedere da un **hashtag**

Esempio:

style.css

```
#giallo{  
  
color:yellow
```

```
}
```

index.html

```
<h2 id="giallo">Questo titolo è giallo</h2>
```

5.2. CLASSI

Una classe è un **elemento che ritorna spesso**, e che quindi posso **riutilizzare**. Per esempio se voglio inserire una serie di bottoni con la stessa formattazione, posso dare loro la classe "**bottone**", impostarla una sola volta nel CSS e questa verrà applicata a tutti gli elementi con la classe "bottone"

Per indicare una classe nel CSS occorre farla precedere da un **punto**.

Esempio:

style.css

```
.bottone{  
  
    background-color:coral;  
  
    border-radius: 15px;  
  
    color:white;  
  
}
```

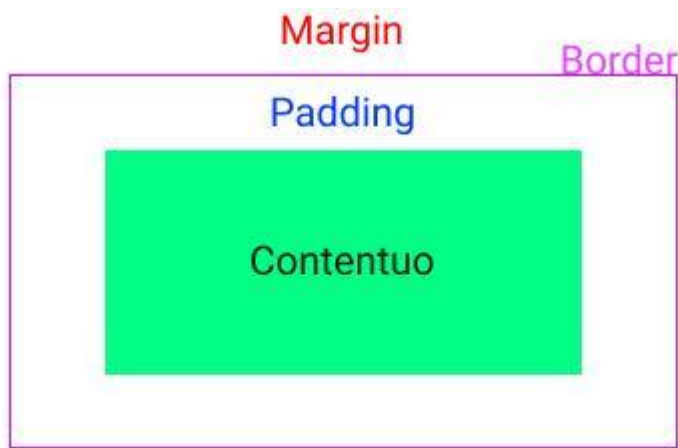
index.html

```
<div class="bottone">Premi qui!</div>
```

6. MARGIN E PADDING

Per gestire gli spazi fra gli elementi si possono utilizzare “**margin**” e “**padding**”.

Ecco uno **schema** per spiegarti che differenza c’è fra i due:



Il **margin** indica lo spazio **all'esterno** del contenuto, il **padding** lo spazio **all'interno**.

È possibile indicare la **direzione** dello spazio sia per il margin che per il padding, per esempio se si vuole inserire un margine superiore occorre utilizzare “**margin-top**”.

Ecco alcuni **esempi**:

style.css

```
.box-margin{  
    background-color: coral;  
  
    margin:50px  
}  
  
.box-padding{  
  
    background-color: coral;
```



```
padding:50px  
}  
  
.box-margin-top{  
  
margin-top: 50px;  
  
background-color: aquamarine;  
}
```

index.html

```
<h2>Margin e Padding</h2>  
  
<h3>Margin:</h3>  
  
<div class="box-margin">  
  
    Questo è un box con del margine  
  
</div>  
  
  
<div class="box-margin-top">  
  
    Questo box ha solo il margine superiore  
  
</div>  
  
  
<h3>Padding</h3>  
  
<div class="box-padding">  
  
    Questo è un box con del padding
```

```
</div>
```

7. CONTENITORE

Gli elementi del CSS possono essere **uno dentro l'altro**, in questo modo permettono di creare layout più elaborati.

Proviamo a rendere la nostra pagina HTML un po' più carina inserendola in un **contenitore**.

Andiamo **sotto il tag body** e inseriamo un **div** con classe **"container"**, in questo modo:

```
<div class="container">
```

Ora andiamo prima del `</body>` e **chiudiamo questo div**, inserendo:

```
</div>
```

Ora aggiungiamo questo codice nel nostro **"style.css"**:

```
.container{  
  
    max-width: 800px;  
  
    margin: 0 auto;  
  
}
```

In questo modo abbiamo impostato una **larghezza massima del contenuto della nostra pagina a 800 pixel**, e impostato il margine del contenuto a 0 pixel dall'alto e dal basso e **automaticamente** da destra e sinistra.

Ora **salviamo** e **aggiorniamo** e vedremo il contenuto inserito a centro pagina, più carino no?

8. IMMAGINE COME SFONDO

Vediamo ancora un'ultima cosa prima di terminare questa prima carrellata generale di CSS: come inserire **un'immagine come sfondo** di un elemento.

Per poter inserire un'immagine come sfondo occorre utilizzare la proprietà **"background-image"**.

Creiamo un **div** che conterrà la nostra immagine nel file **html**:

```
<div class="immagine-sfondo">  
  
    Questo div ha un'immagine di sfondo!  
  
</div>
```

E inseriamo l'url all'immagine tramite il **CSS** nel nostro "style.css":

```
.immagine-sfondo{  
  
    background-image: url(img/immagine.jpg);  
  
    height:500px;  
  
    text-align: center;  
  
    padding-top: 250px;  
  
    color:white;  
  
}
```

In questo modo abbiamo impostato **la nostra immagine come sfondo**. Abbiamo anche impostato **un'altezza** in modo da far vedere bene l'immagine.

Prova a **salvare** e **aggiornare** e vedrai cosa succede.

Ora prova a smanettare un po' con queste classi e con queste regole, modificando dimensioni, font, colori, immagini e tutto ciò che hai in mente!

Ricorda che **il modo migliore per imparare è dedicare tanto tempo** alla pratica, quindi **inizia a darci dentro con il CSS!**

9. CODICE COMPLETO:

Qua puoi trovare il **codice completo** dei file index.html e style.css

index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>La mia prima pagina web</title><!-- Il titolo della pagina che appare
nella scheda del browser -->

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

</head>
```

```
<body>

<div class="container">

  <!-- Titolo -->

  <h1>La mia prima pagina web</h1>

  <!-- Paragrafo -->

  <p>Benvenuto nella mia prima pagina web!</p>


  <br><!-- questo è un a capo-->


  <!-- Sottotitolo -->

  <h2>Sottotitolo</h2>


  <p>Questo è il secondo paragrafo della mia prima pagina web</p>


  <br>


  <h2>Elenco</h2>

  <!-- Elenco -->

  <ul>

    <li>Primo Item</li><!-- Item di un elenco -->

    <li>Secondo Item</li>
```

```
</ul>
```

```
<!-- Immagine -->
```

```

```

```
<!-- DIV: block element -->
```

```
<div style="background-color: green;">
```

Questo è un contenitore con sfondo verde

```
</div>
```

```
<!-- SPAN: inline element -->
```

```
<p>
```

Questo è un paragrafo con del testo inserito a caso. In questo testo
voglio

colorare una parola di rosso

```
</p>
```

```
<!-- FORM -->
```

```
<form>
```

```
<!-- Casella di testo -->
```

```
<input type="text" placeholder="Nome">
```

```
<br><br>
```

```
<input type="text" placeholder="Cognome">
```

```
<br><br>
```

```
<!-- Menù a tendina -->
```

```
<select name="select" id="">
```

```
  <option value="0">Opzione 1</option>
```

```
  <option value="1">Opzione 2</option>
```

```
  <option value="2">Opzione 3</option>
```

```
</select>
```

```
<br><br>
```

```
<!-- Area di testo -->
```

```
  <textarea name="" id="" cols="30" rows="10" placeholder="Inserisci il  
testo qui."></textarea>
```

```
<br><br>
```

```
<!-- Checkbox-->
```

```
  <input type="checkbox" name="privacy" value="0">Accetto la Privacy  
Policy
```

```
<br><br>
```

```
<!-- Bottone -->
```

```
<button>Invia</button>
```

```
</form>
```

```
<h2>Margin e Padding</h2>
```

```
<h3>Margin:</h3>
```

```
<div class="box-margin">
```

```
    Questo è un box con del margine
```

```
</div>
```

```
<div class="box-margin-top">
```

```
    Questo box ha solo il margine superiore
```

```
</div>
```

```
<h3>Padding</h3>
```

```
<div class="box-padding">
```

```
    Questo è un box con del padding
```

```
</div>
```

```
<div class="immagine-sfondo">
```

```
    Questo div ha un'immagine di sfondo!
```

```
</div>
```

```
</div>
```

```
</body>
```



```
</html>
```

style.css

```
@import
url('https://fonts.googleapis.com/css2?family=Source+Sans+Pro:wght@400;700&display=swap');

body{

    font-family: 'Source Sans Pro', sans-serif;

    font-size: 22px;

    font-weight: lighter;

    line-height: 35px;

    font-style: normal;

}

.box-margin{

    background-color: coral;

    margin:50px

}

.box-padding{

    background-color: coral;

    padding:50px

}

.box-margin-top{
```

```
margin-top: 50px;

background-color: aquamarine;
}

.container{

    max-width: 800px;

    margin: 0 auto;
}

.immagine-sfondo{

    background-image: url(img/immagine.jpg);

    height:500px;

    text-align: center;

    padding-top: 250px;

    color:white;
}
```

Le basi di Javascript!

Introduzione

Javascript è il linguaggio che permette di **creare animazioni** nei contenuti web. Tutte le gallery, gli slider, i pop up, le transizioni di pagina e ogni effetto animato che vedi navigando online è realizzato con **Javascript**.

Questo linguaggio **si è sviluppato moltissimo**, passando dall'essere una cosa in più, un modo per creare effetti divertenti e simpatici, ad essere oggi uno dei più utilizzati al mondo, non solo per animazioni. ma per creare **vere e proprie strutture software** in grado di far funzionare applicativi potentissimi.

In questo articolo **tratteremo le basi**, partiremo **da zero** e vedremo come funziona questo linguaggio di programmazione, e creeremo insieme qualcosa di **semplice** ma utile per capire come utilizzare questo linguaggio.

Faremo prima un po' di teoria e poi passeremo a creare qualcosa di utilizzabile sulla nostra pagina web.

Se ti perdi durante l'articolo sul [fondo di questo](#) articolo potrai trovare il [codice](#) di tutto ciò che andremo a creare.

1. INSERIRE JAVASCRIPT NELL'HTML

Iniziamo ad **inserire** del codice Javascript nel nostro file html (se non sai di cosa sto parlando dai un'occhiata ai nostri articoli su [HTML](#) e [CSS](#))

Come per il codice CSS, anche il Javascript può essere inserito in **modi diversi**:

- Javascript interno
- Javascript esterno

1.1. JAVASCRIPT INTERNO

In questo caso il codice Javascript è inserito **direttamente nel file html** prima del fine body (`</body>`), fra i tag `<script>` e `</script>`.

Esempio:

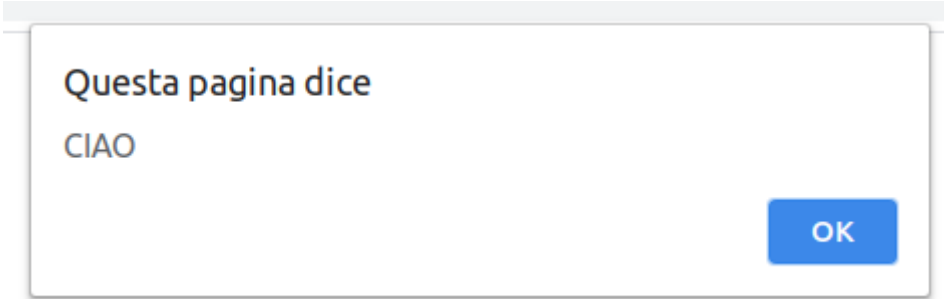
Apriamo il nostro **"index.html"**, andiamo sul fondo e inseriamo questo codice appena prima del tag `</body>`:

```
<script>

    alert("CIAO");

</script>
```

Ora **salviamo** e **apriamo** la pagina **"index.html"**. Ecco che apparirà un **popup** con scritto **"CIAO!"**.



Questa pagina dice
CIAO

OK

1.2. JAVASCRIPT ESTERNO

Per inserire il javascript esterno occorre **creare un file .js** e **importarlo** nell'html. Come per il CSS questo è **il metodo migliore**, quasi sempre.

Esempio:

Andiamo nella nostra cartella "**HTML**" sul desktop, la apriamo con code (guadra il [corso intensivo di HTML](#)) e creiamo un file (**CTRL+N**) e lo salviamo (**CTRL+S**) con il nome "**scripts.js**".

All'interno di questo file scriviamo:

```
alert("CIAO");
```

Ora **salviamo** "**scripts.js**" e apriamo "**index.html**".

Qua, sempre prima del `</body>`, inseriamo questo:

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

Ora **salviamo** e **aggiorniamo** il browser.

Il risultato è uguale a prima: il popup, solo che il modo con cui l'abbiamo fatto apparire è diverso.

2. UN PO' DI TEORIA

2.1. ALERT

Abbiamo appena utilizzato la funziona **"alert"**, che permette di far apparire un **popup** nella pagina web con del testo al suo interno. Raramente questo è utilizzato per i popup, perché **l'estetica è un po' bruttina** ed esistono metodi molto migliori per far comparire dei popup nelle pagine web, ma può essere molto utile **in fase di debugging**.

Se sto creando una funzione e non riesco a trovare l'errore, posso inserire un **"alert"** a metà funzione e vedere così se l'errore si verifica prima o dopo.

2.2. CONSOLE.LOG

Un altro modo per visualizzare errori in Javascript è la funzione **"console.log"**. Questa permette di **inserire del testo nella console di Javascript**. Anche questa è molto utile in fase di debug e sviluppo.

Ecco un esempio:

```
console.log("CIAO!");
```

Se inserisci questa in **"scripts.js"**, salvi e aggiorni, non vedrai accadere niente. Questo perché il **"CIAO!"** che abbiamo scritto è **inserito nella console di Javascript**, non nel body della pagina. Per vedere la console Javascript premi il tasto **"f12"**, oppure fai click con il tasto destro e clicca su **"ispeziona"** (Su google chrome, ma è molto simile su tutti i browser)



2.3. COMMENTI

I **commenti** sono **importantissimi** in ogni linguaggio di programmazione.

Per inserire i commenti in Javascript esistono **due modi**:

Se il commento è su **una sola riga** puoi inserire un **doppio slash** prima della riga. In questo modo **tutta la riga sarà commentata**.

Esempio

```
// Questo è un commento su una riga Javascript
```

Se invece **il commento è più lungo**, puoi usare la stessa sintassi del CSS: **/*
Commento */**

Esempio:

```
/*  
  
Questo è un commento  
Javascript su più righe  
  
*/
```

2.4. VARIABILI

A differenza di HTML e CSS, **Javascript è un vero e proprio linguaggio di programmazione**, e non dei più semplici.

Come ogni linguaggio di programmazione è possibile **utilizzare delle variabili per memorizzare i dati e fare calcoli**.

Inserire una variabile è molto semplice, basta inserire **"var"** prima della variabile, in questo modo:


```
var anni = 30;
```

È fondamentale **inserire il punto e virgola** alla fine della variabile, per indicare che la regola finisce in quel punto. Senza il punto e virgola verranno generati degli errori.

Ora possiamo **richiamare la variabile dentro il console.log** oppure in un **alert**, in questo modo:

```
console.log(anni);
```

oppure:

```
alert(anni);
```

Salviamo e aggiorniamo e vedremo il numero **"30"** apparire nel popup oppure nella cosole.

Ma possiamo fare di più!

Impostiamo **una serie di variabili**:

```
var nome = "Marco";  
  
var altezza = "1.83";  
  
var anni = 30;
```

Adesso possiamo **creare una frase ed inserire al suo interno le nostre variabili**, oltre che utilizzarle per fare dei calcoli.

I **valori testuali** devono essere inseriti fra **virgolette**, mentre i **valori numerici** **senza**. In questo modo possiamo fare anche delle **operazioni aritmetiche**.

Ecco un esempio:

```
console.log('Ciao, mi chiamo ' + nome + ' e sono alto ' + altezza + ' metri. In questo momento ho ' + anni + ' anni. Fra 5 anni avrò ' + (anni + 5) + ' anni.');
```

Per inserire una **variabile in un testo** occorre **concatenarla**, metterla insieme.

Per fare questo abbiamo utilizzato il segno **'+'**.

2.5. FUNZIONI

Le funzioni sono delle **parti di codice che svolgono una determinata azione**.

Isolando una parte di codice in una funzione, questa potrà essere **richiamata** più volte all'interno del progetto.

Le funzioni possono avere dei **parametri**, che ne personalizzano l'azione.

Esempio:

Andiamo sul file ***scripts.js*** e scriviamo:

```
function ciao() {  
  
    alert ("CIAO");  
  
}
```

Ora andiamo sul file ***index.html*** e aggiungiamo un bottone, in questo modo:

```
<button onclick="ciao()">Salutami</button>
```

Salviamo e aggiorniamo e vedremo che **clickando sul nuovo bottone apparirà il popup con scritto "CIAO"**.

2.6. PARAMETRI

Adesso aggiungiamo un **parametro**. Andiamo nella funzione a la modifichiamo così:

```
function ciao(nome) {  
    alert ("CIAO "+nome);  
}
```

e sul file ***index.html*** modifichiamo così il bottone:

```
<button onclick="ciao('Marco')">Salutami</button>
```

Salviamo e aggiorniamo e possiamo vedere che ora il saluto è rivolto al nome inserito nel parametro della funzione!

"Marco" è il nostro parametro

2.7. IF ELSE

Gli "If" sono **alla base di tutta la programmazione**. Ogni azione è fatta come conseguenza di un'altra. Vediamo cosa significa.

Utilizziamo sempre il bottone del saluto. Possiamo prevedere che se il nome è "Marco" allora il popup dirà "Ciao Marco", se invece il nome è "Mark", possiamo far apparire "Hello Mark".

Vediamo come fare:

Sostituiamo la funzione con questa:

```
function ciao(nome) {  
  
    if(nome=="Marco"){  
  
        alert ("CIAO "+nome);  
  
    }else if(nome=="Mark"){  
  
        alert ("Hello "+nome);  
  
    }else{  
  
        alert ("Buongiorno "+nome);  
  
    }  
  
}
```

Ora possiamo andare a **modificare il parametro nel bottone**, da "Marco" a "Mark", oppure inserire un nome totalmente diverso.

Se il nome è Marco allora il popup sarà "Ciao Marco", se Mark allora "Hello Mark", se altro sarà "Buongiorno + nome".

Da notare come **ci siano due segni uguale**, questo perché nel Javascript un uguale assegna il valore, per confrontarli invece occorre usarne 2 o 3 a seconda dei casi. Per il momento ci basta sapere che quando dobbiamo confrontare più valori bisogna inserire 2 segni uguale.

2.8. EVENTI

Il Javascript può essere richiamato all'interno dell'HTML all'accadere di determinati eventi, per esempio al click, al passaggio con il mouse etc.

Oggi esistono anche molti altri modi, ma per iniziare questi sono i più semplici ed immediati.

Esempio:

```
<button onclick="alert('CIAO!')">Salutami</button>
```

In questo caso al click del bottone apparirà il **popup di saluto**.

3. CREIAMO IL NOSTRO PRIMO EFFETTO

Al di là della teoria, **a noi interessa soprattutto vedere come possiamo utilizzare Javascript per creare gli effetti per le nostre pagine web.**

Quindi ora andremo a **creare un semplice effetto che cambierà lo sfondo del body al click di un bottone.**

Vediamo come fare:

Iniziamo con il creare un bottone nella nostra ***index.html***:

```
<button onclick="cambiaSfondo()"> Cambia sfondo! </button>
```

Ora andiamo nel nostro ***scripts.js*** e creiamo la funzione **cambiaSfondo()**:

```
function cambiaSfondo(){  
    document.body.style.backgroundColor='#000';  
}
```

Se **salviamo e aggiorniamo** vedremo che ora **al click del bottone lo sfondo diventerà nero.**

Abbiamo detto al browser che al click del bottone deve selezionare il colore di sfondo del body (body.style.backgroundColor) e impostarlo a nero.

Notiamo però che **è un'unica azione**, una volta che lo sfondo è nero non possiamo più tornare al bianco...

Andiamo ad aggiungere ancora qualche linea di codice:

```
function cambiaSfondo(){  
  
    var sfondo = document.body.style.backgroundColor;  
  
    if(sfondo=="rgb(0, 0, 0)")  
    {  
        document.body.style.backgroundColor='#fff';  
    }else{  
        document.body.style.backgroundColor='#000';  
    }  
}
```

In questo modo facciamo un **controllo del colore di sfondo**. Se è nero lo impostiamo bianco, altrimenti sarà nero.

Così facendo possiamo **cambiare colore di sfondo ogni volta che clicchiamo sul bottone**.

Congratulazioni! Hai appena creato il tuo primo effetto Javascript!!!

4. ANIMAZIONE DI UN COMPONENTE

Se al posto dello sfondo intero volessimo modificare solamente un **componente**, possiamo farlo utilizzando gli **id** ([corso intensivo di CSS](#)).

Iniziamo con il creare un **div con id="box"**

```
<div id="box" onmouseover="cambiaBoxOver()"
onmouseout="cambiaBoxOut()">

    Questo contenitore cambierà colore al passaggio del mouse

</div>
```

Ora **creiamo le due funzioni in scripts.js**:

```
function cambiaBoxOver(){

    document.getElementById('box').style.backgroundColor="coral";

    document.getElementById('box').style.color="blue";

}

function cambiaBoxOut(){

    document.getElementById('box').style.backgroundColor="white";

    document.getElementById('box').style.color="black";

}
```

Salviamo e aggiorniamo e vediamo che **il contenitore cambia colore e sfondo al passaggio del mouse**.

Molto bene, **queste sono le basi per iniziare a smanettare un po' con il Javascript.**

Cercando online potrai trovare moltissime guide e tutorial sull'argomento. Ricorda che qua siamo **Specialisti WP**, quindi le nostre skills più approfondite saranno riservate a WordPress ed al suo linguaggio: il PHP.

È però bene conoscere anche un po' di Javascript e soprattutto JQuery, per poter **evitare di installare un'infinità di plugin e crearci da soli tutti gli effetti e le funzioni che vogliamo!**

Se ti perdi durante l'articolo sul [fondo di questo](#) articolo potrai trovare il [codice](#) di tutto ciò che andremo a creare.

5. CODICE COMPLETO:

index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>La mia prima pagina web</title><!-- Il titolo della pagina che appare
nella scheda del browser -->

  <link rel="stylesheet" href="style.css">
```



```
<br><!-- questo è un a capo-->
```

```
<!-- Sottotitolo -->
```

```
<h2>Sottotitolo</h2>
```

```
<p>Questo è il secondo paragrafo della mia prima pagina web</p>
```

```
<br>
```

```
<h2>Elenco</h2>
```

```
<!-- Elenco -->
```

```
<ul>
```

```
  <li>Primo Item</li><!-- Item di un elenco -->
```

```
  <li>Secondo Item</li>
```

```
</ul>
```

```
<!-- Immagine
```

```
 -->
```

```
<!-- DIV: block element -->
```

```
<div style="background-color: green;">
```

Questo è un contenitore con sfondo verde

</div>

<!-- SPAN: inline element -->

<p>

Questo è un paragrafo con del testo inserito a caso. In questo testo
voglio

colorare una parola di rosso

</p>

<!-- FORM -->

<form>

<!-- Casella di testo -->

<input type="text" placeholder="Nome">

<input type="text" placeholder="Cognome">

<!-- Menù a tendina -->

<select name="select" id="">

<option value="0">Opzione 1</option>

<option value="1">Opzione 2</option>

<option value="2">Opzione 3</option>

```
</select>
```

```
<br><br>
```

```
<!-- Area di testo -->
```

```
    <textarea name="" id="" cols="30" rows="10" placeholder="Inserisci  
il testo qui."></textarea>
```

```
<br><br>
```

```
<!-- Checkbox-->
```

```
    <input type="checkbox" name="privacy" value="0">Accetto la  
Privacy Policy
```

```
<br><br>
```

```
<!-- Bottone -->
```

```
<button>Invia</button>
```

```
</form>
```

```
<h2>Margin e Padding</h2>
```

```
<h3>Margin:</h3>
```

```
<div class="box-margin">
```

```
    Questo è un box con del margine
```

```
</div>
```

```
<div class="box-margin-top">
```

```
    Questo box ha solo il margine superiore
```

```
</div>

<h3>Padding</h3>

<div class="box-padding">

    Questo è un box con del padding

</div>

<!--
<div class="immagine-sfondo">

    Questo div ha un'immagine di sfondo!

</div>
-->

</div>

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

</body>

</html>
```

scripts.js

```
function cambiaSfondo(){  
  
    var sfondo = document.body.style.backgroundColor;  
  
    if(sfondo=="rgb(0, 0, 0)"){  
  
        document.body.style.backgroundColor='#fff';  
  
    }else{  
  
        document.body.style.backgroundColor='#000';  
  
    }  
}
```

```
function ciao(nome) {  
  
    if(nome=="Marco"){  
  
        alert ("CIAO "+nome);  
  
    }else if(nome=="Mark"){  
  
        alert ("Hello "+nome);  
  
    }else{  
  
        alert ("Buongiorno "+nome);  
  
    }  
}
```

```
function cambiaBoxOver(){  
  
    document.getElementById('box').style.backgroundColor="coral";  
  
    document.getElementById('box').style.color="blue";  
  
}
```

```
function cambiaBoxOut(){  
  
    document.getElementById('box').style.backgroundColor="white";  
  
    document.getElementById('box').style.color="black";  
  
}
```

Conclusione

Se sei arrivato fino a questo punto allora hai imparato le basi dei 3 principali linguaggi per diventare uno sviluppatore web.

La strada da fare è ancora molto lunga, il web si evolve di continuo così come i suoi linguaggi, ma HTML, CSS e Javascript ci saranno sempre, sono le fondamenta del web.

Continua a sperimentare e a provare. Il miglior modo per imparare e fare, puoi leggere tutti i libri di programmazione del mondo ma se non scrivi codice non imparerai mai veramente a sviluppare, perciò prenditi del tempo ogni giorno e prova, modifica i layout delle tue pagine web, crea formattazioni ed effetti e vedrai che nel giro di poco tempo riuscirai a creare i tuoi primi siti web.

Come sempre se hai bisogno di aiuto puoi scrivermi a help@albertoreineri.it

Spero che questo libro ti abbia aiutato nel creare le tue fondamenta da sviluppatore web.

Buono sviluppo!

Alberto Reineri

www.albertoreineri.it