Vediamo come generare un output sulla pagina, usando il codice JS.

**Output**

Un’azione di output prevede di far comparire qualcosa (ad esempio un messaggio o un oggetto) all’interno della pagina web.

JavaScript è in grado di produrre un output in vari modi:

* la generazione di elementi HTML sulla pagina;
* la generazione di un messaggio in una finestra di dialogo;
* la modifica di un elemento (a livello di nodo) della pagina web;
* l’invio di un messaggio alla console di debug del browser.

**document.write()**

Il primo output che vediamo consiste nell’invocare il metodo **write()** sull’oggetto document.

Quando una pagina web (un file HTML) viene caricata in un browser, viene generato un oggetto **document**.

L’oggetto document ha diverse proprietà e metodi a sua disposizione.

Tra i suoi metodi troviamo **write()** che permette di “riempire” il documento con un contenuto di tipo HTML, passato come argomento (all’interno delle parentesi).

Pensiamo ad un codice di questo tipo:

<script>

 document.write("Testo semplice");

</script>

Tale codice, all’interno di una pagina web, inserisce un contenuto HTML nel punto in cui viene collocato lo script.

**alert()**

Un altro modo per generare un output è quello di sfruttare una **finestra di dialogo**.

Una finestra di dialogo è una finestrella che appare per avvisare l’utente della pagina o per chiedergli qualcosa.

Un esempio di questa finestra è dato da **alert()**:

<script>

 alert("CIAO CIAO");

</script>

Alert genera una finestra molto semplice, costituita da un messaggio e da un pulsante che permette di chiuderla.

Il messaggio corrisponde a ciò che voi indicherete nelle parentesi tonde, sempre tra doppi apici.

Esempio di risultato finale:



**innerHTML**

Un modo “intelligente” per inserire un contenuto HTML in un punto preciso della pagina è dato da innerHTML.

Secondo il DOM (Document Object Model), la pagina web viene vista come una struttura ad albero, caratterizzata da elementi genitori e sotto-elementi figli.

Ciascuno di questi elementi è detto **nodo** e rappresenta un oggetto della pagina web.

L’oggetto di livello superiore, quello in cima alla gerarchia è **document**: tutti gli altri nodi sono suoi figli.

Con JS è possibile manipolare i singoli nodi figli del document ma, per farlo, è necessario prima selezionarli.

Per selezionare i nodi possiamo usare diversi metodi (occhio alle maiuscole!):

document.getElementById()

Tale metodo è in grado di selezionare un determinato nodo in base al valore del suo attributo Id.

Esempio, se la pagina contiene questo paragrafo:

 <body>

 <p id="speciale">Paragrafo speciale.</p>

 </body>

Il paragrafo è stato etichettato con l’id “speciale”, e solo lui può avere questo nome, in modo da poterlo individuare tra altri paragrafi.

Per poter scrivere del testo all’interno di quel determinato paragrafo, possiamo usare:

document.getElementById("speciale")

Ma selezionarlo e basta non serve a nulla, dobbiamo dire al codice JS cosa deve fare con quel nodo, per questo usiamo il metodo innerHTML:

<body>

 <p id="speciale">Paragrafo speciale.</p>

 <script>

 document.getElementById("speciale").innerHTML = "Nuovo contenuto";

 </script>

</body>

In questo esempio, abbiamo individuato l’unico paragrafo speciale della pagina e gli abbiamo modificato il contenuto, sovrascrivendolo.

**console.log()**

Un ultimo metodo di output si usa nelle operazioni di testing e debug dei programmi JS, e prevede di inviare messaggi alla console del browser.

Questa console è un ambiente di controllo che solitamente viene visualizzato sul vostro browser premendo il tasto F12.

Con il comando JS:

console.log("messaggio");

potete mostrare un messaggio direttamente nella finestra di output della console per lo sviluppo.

Si tratta di una operazione utile prevalentemente all’analisi dei vostri programmi, al testing ed al debug.