XMLHttpRequest

XMLHttpRequest è un oggetto del linguaggio di programmazione Javascript, ad oggi è il principale metodo per effettuare le richieste HTTP e HTTPS in modo asincrono tra dispositivi.

Questo oggetto una volta dichiarato permette di richiedere e inviare dati da e verso un server esterno senza il bisogno di ricaricare l'intera pagina web. L'oggetto XMLHttpRequest è stato introdotto come componente della tecnologia AJAX (Asynchronous JavaScript and XML) che ha permesso di velocizzare e rendere efficiente l'aggiornamento di dati all'interno di una pagina web.

L'Oggetto XMLHttpRequest è in grado di gestire vari formati di dati oltre all'XML [Formato Nativo] come JSON e CSV.

Effettuare richieste asincrone al server permette al browser di continuare a funzionare senza essere bloccato dalla richiesta in corso. Il browser, infatti, può continuare ad interpretare il codice JavaScript e quando il file richiesto viene completamente scaricato viene annesso. Questa tecnologia è particolarmente importante se si tratta di file di dimensioni elevate che potrebbero portare a un errore di time out o problemi di sicurezza informatica.

L'asimmetria tra la richiesta e la risposta permette di aggiornare il contenuto di una pagina in modo dinamico, riducendo i tempi di attesa e migliorando l'esperienza di navigazione evitando schermate vuote. Inoltre, l'uso asincrono delle richieste riduce il carico sui server e rende le applicazioni web molto più reattive.

Per inizializzare una richiesta occorre fornire alcuni parametri al metodo *.open()* tra cui:

- Il metodo utilizzato per trasferire le informazioni; si può scegliere se utilizzare il metodo POST o GET.
- Il server o l'url del server a cui effettuare la richiesta del servizio
- Il modo di effettuare la richiesta: in questo caso può essere sincrono (false) o asincrono (true) [è preferibile l'uso dell'asincrono]

```
xhttp.open('GET', 'https://…', true);
```

Successivamente è possibile salvare i dati ricevuti tramite la funzione *.response*.

```
const xhttp = new XMLHttpRequest();

xhttp.open('GET', '', true)

xhttp.onreadystatechange = function() {

    if(xhttp.readyState === 4 && xhttp.status === 200) {

        var str = xhttp.response;
    }
}

xhttp.send();
```

Un modo rapido ed efficiente di inviare i dati è con il formato dato JSON.

JSON (JavaScript Object Notation) è un formato dati basato su una struttura costituita da chiave e valore; i tipi di valore che un JSON può supportare sono molteplici tra cui Array, Stringhe, ecc. JSON offre anche la possibilità di poter essere utilizzato su linguaggio di programmazione differenti, tra cui PHP, Java, JavaScript, Ruby e Python.

JSON è nato nel 2000 per semplificare e rendere più agevole la scrittura e la comprensione rispetto i file XML, allora standard molto usato nello sviluppo web.

Per quanto riguarda la sintassi l'oggetto viene dichiarato come segue:

var oggetto = {nome: "Marco", cognome: "Rossi", residenza: "Roma", anni: 23}

In un oggetto JSON le chiavi sono separate dai valori dai due punti mentre le diverse coppie di chiave e valore sono separate tra di loro con le virgole.

Il valore può assumere tipi dato diversi nello stesso oggetto come ad esempio stringhe, interi o array.

Tramite la funzione *JSON.stringify(oggetto)* è possibile convertire l'oggetto creato in una stringa facilmente trasferibile a un'altra pagina HTML tramite il Local Storage o un XMLHttpRequest.

```
<script>

    function invia() {

        var elenco = {"nome": "Marco", "cognome": "Rossi", "anni": 23};

        var elencoStr = JSON.stringify(elenco);

        localStorage.setItem("Elenco", elencoStr);

        location.href = "JSON2.html";

    }

</script>
```

La funzione inversa è la *JSON.parse(stringa)* che da una stringa crea un variabile JSON utilizzabile come oggetto in JavaScript.

```
<script>

    var stringaOG = localStorage.getItem("Elenco");

    var oggetto = JSON.parse(stringaOG);

    document.getElementById("testo").innerHTML = oggetto.nome;

</script>
```