Introduzione a JSON

JSON sta per **J**ava**S**cript **O**bject **N**otation, è un modo leggero di trasmettere e memorizzare i dati ed è facile da utilizzare. Come con XML, che vedremo, è autoesplicativo e usa il linguaggio JS ad oggetti.

JSON permette di aggregare dati (stringhe, numeri, etc.) per creare informazioni di più alto livello, tipicamente qualcosa che nel nostro programma rappresenta un elemento del mondo reale.

Per creare un oggetto JSON è sufficiente:

- disporre una coppia di parentesi graffe, quelle che notiamo all'inizio e alla fine della struttura, il cui scopo sarà contenere l'oggetto intero;
- inserire all'interno delle parentesi graffe una sequenza di proprietà il cui nome e valore siano separate dal simbolo di due punti. Ad esempio, "nome":"Edoardo" significa che l'oggetto possiede la proprietà "nome" il cui valore è impostato a "Edoardo";
- le proprietà di un oggetto sono separate da virgole.

es.

```
var miOggetto = {nome: "Edoardo", età: 15, città: "Rho"};
var myJSON = JSON.stringify(miOggetto);
window.location = "prova.php?x=" + myJSON;
```

In questo modo invio ad una pagina php un formato standard autoesplicativo di tipo JSON.

Per ricevere:

```
var miElenco = '{"nome":"Andrea", "età":22, "città":"Codogno"}';
var miOggetto = JSON.parse(miElenco);
document.getElementById("prova").innerHTML = miOggetto.nome;
```

Se volessimo registrare dei dati:

```
var miElenco, miOggetto, testo, oggetto;

//REGISTRO

miElenco = {"nome":"Andrea", "età":22, "città":"Codogno"};
miOggetto = JSON.stringify(miElenco);
localStorage.setItem("provaJSON", miOggetto);

//STAMPO

testo = localStorage.getItem("provaJSON");
oggetto = JSON.parse(testo);
document.getElementById("prova").innerHTML = oggetto.nome + " " + oggetto.età;
```

Notare sempre che in **strumenti per gli sviluppatori** di **Chrome**, è possibile visualizzare in "**Application/Local Storage/File**" l'oggetto registrato

I valori accettati possono essere:

- una stringa (sono obbligatori, a differenza di JS, i doppi apici)
- un numero
- un oggetto (oggetto JSON - quindi multidimensionale!!!)
- un array
- un boolean
- null

lato server, in **php**, la sintassi prevede:

```
<? php
  $ miElenco-> nome = "Andrea" ;
  $ miElenco-> atà = 22 ;
  $ miElenco-> città = "Codogno" ;

  $ miOggetto = json_encode ($ miElenco);

  echo $ miOggetto;
?>
```

Il risultato:

```
{"nome": "Andrea", "età": 22, "città": "Codogno"}
```

Anche lato **client** sarà possibile leggere un file di tipo **json** che è stato creato lato server (da provare con xammp):

```
<p id="demo"></p>

<script>
var xmlhttp = new XMLHttpRequest();
xmlhttp.onreadystatechange = function() {
if (this.readyState == 4 && this.status == 200) {
var myObj = JSON.parse(this.responseText);
document.getElementById("demo").innerHTML = myObj.name;
}
};
xmlhttp.open("GET", "prova.json", true);
xmlhttp.send();
</script>

</body>
</html>
```

Il file di prova.json sarà di questo tipo:

```
{
"nome":"Andrea",
"età":22,
"auto":[
{ "tipo":"suv", "nome":"bmw" },
{ "tipo":"coupè", "nome":"Fiat" },
{ "tipo":"berlina", "nome":"Lancia" }
]
}
```

per verificare la corretta sintassi del file json si può utilizzare il seguente software online: http://json.parser.online.fr/

Provare a fare un esercizio di organizzazione di semplice magazzino di dischi mettendo insieme lato client e server