**JSON: JavaScript Object Notation-** [**https://www.w3schools.com/js/js\_json\_intro.asp**](https://www.w3schools.com/js/js_json_intro.asp)

* JSON è una sintassi per l'archiviazione e lo scambio di dati.
* JSON è un testo, scritto con notazione di oggetti JavaScript.

**Scambio di dati:**

Quando si scambiano dati tra un browser e un server, i dati possono essere solo testo.

JSON è testo e possiamo convertire qualsiasi oggetto JavaScript in JSON e inviare JSON al server. Possiamo inoltre convertire qualsiasi JSON ricevuto dal server in oggetti JavaScript.

**Invio di dati:**

Se si dispone di dati memorizzati in un oggetto JavaScript, è possibile convertire l'oggetto in JSON e inviarlo a un server:

var myObj = {name: "John", age: 31, city: "New York"};  
var myJSON = *JSON.stringify(myObj)*;  
window.location = "demo\_json.php?x=" + myJSON;

<https://www.w3schools.com/js/tryit.asp?filename=tryjson_send>

**Ricevendo dati:**

Se si ricevono dati in formato JSON, è possibile convertirli in un oggetto JavaScript:

var myJSON = '{"name":"John", "age":31, "city":"New York"}';  
var myObj = *JSON.parse(myJSON)*;  
document.getElementById("demo").innerHTML = myObj.name;

<https://www.w3schools.com/js/tryit.asp?filename=tryjson_receive>

**Memorizzazione dei dati:**

Quando si memorizzano i dati, i dati devono essere di un certo formato e, indipendentemente da dove si sceglie di memorizzarli, il ***testo*** è sempre uno dei formati legali. ***JSON*** rende possibile memorizzare ***oggetti JavaScript*** come testo. Memorizzazione dei dati nella memoria locale:

// Storing data:  
myObj = {name: "John", age: 31, city: "New York"};  
myJSON = JSON.stringify(myObj);  
localStorage.setItem("testJSON", myJSON);  
  
// Retrieving data:  
text = localStorage.getItem("testJSON");  
obj = JSON.parse(text);  
document.getElementById("demo").innerHTML = obj.name;

<https://www.w3schools.com/js/tryit.asp?filename=tryjson_store>

**Sintassi -** [**https://www.w3schools.com/js/js\_json\_syntax.asp**](https://www.w3schools.com/js/js_json_syntax.asp)

La sintassi JSON è derivata dalla sintassi di notazione di oggetti JavaScript:

* I dati sono in coppie nome / valore: "name":"John"
* I dati sono separati da virgole
* Le parentesi graffe tengono oggetti: { "name":"John" }
* Le parentesi quadre contengono array

I nomi JSON richiedono doppi apici. I nomi JavaScript non lo fanno.

I ***valori*** devono essere uno dei seguenti tipi di dati:

* una stringa { "name":"John" }
* un numero { "age":30 }
* un oggetto (oggetto JSON) { "employee": "name":"John","age":30 }
* un array { "employees":[ "John", "Anna", "Peter" ] }
* un booleano { "sale":true }
* nullo { "middlename":null }

**JSON vs XML -** [**https://www.w3schools.com/js/js\_json\_xml.asp**](https://www.w3schools.com/js/js_json_xml.asp)

Sia JSON che XML possono essere utilizzati per ricevere dati da un server web.

**JSON**:

{"employees":[  
  { "firstName":"John", "lastName":"Doe" },  
  { "firstName":"Anna", "lastName":"Smith" },  
  { "firstName":"Peter", "lastName":"Jones" }  
]}

**XML**:

<employees>  
  <employee>  
    <firstName>John</firstName> <lastName>Doe</lastName>  
  </employee>  
  <employee>  
    <firstName>Anna</firstName> <lastName>Smith</lastName>  
  </employee>  
  <employee>  
    <firstName>Peter</firstName> <lastName>Jones</lastName>  
  </employee>  
</employees>

* Sia JSON che XML sono "auto descrittivi" (leggibili dall'uomo)
* Sia JSON che XML sono gerarchici (valori all'interno dei valori)
* Sia JSON che XML possono essere analizzati e utilizzati da molti linguaggi di programmazione
* Sia JSON che XML possono essere recuperati con XMLHttpRequest

Unica differenza :

* JSON non usa il tag di fine
* JSON è più breve
* JSON è più veloce da leggere e scrivere
* JSON può usare matrici
* ***XML*** deve essere analizzato con un ***parser*** ***XML***.
* ***JSON*** può essere analizzato da una ***funzione*** ***JavaScript*** standard.

XML è molto più difficile da analizzare rispetto a JSON e viene analizzato in un oggetto JavaScript pronto per l'uso.

**Oggetti JSON -** [**https://www.w3schools.com/js/js\_json\_objects.asp**](https://www.w3schools.com/js/js_json_objects.asp)

Gli oggetti JSON sono circondati da parentesi graffe {}, coppie nome / valore, i nomi devono essere stringhe e i valori devono essere un tipo di dati JSON valido: { "name":"John", "age":30, "car":null }

**Accesso ai valori degli oggetti:**

myObj = { "name":"John", "age":30, "car":null };  
x = myObj.name; <https://www.w3schools.com/js/tryit.asp?filename=tryjson_object_dot> *or* x = myObj["name"];

**Loop di un oggetto:**

È possibile scorrere le ***proprietà*** dell'oggetto utilizzando il ciclo for-in:

myObj = { "name":"John", "age":30, "car":null };  
for (x in myObj) {  
  document.getElementById("demo").innerHTML += x;  
} <https://www.w3schools.com/js/tryit.asp?filename=tryjson_object_loop>

In un ciclo for-in, utilizzare la notazione della parentesi ***per accedere ai valori*** delle proprietà:

myObj = { "name":"John", "age":30, "car":null };  
for (x in myObj) {  
  document.getElementById("demo").innerHTML += myObj[x];  
} <https://www.w3schools.com/js/tryit.asp?filename=tryjson_object_loop_bracket>

**Oggetti JSON annidati:**

I valori in un oggetto JSON possono essere un altro oggetto JSON:

myObj = {  
  "name":"John",  
  "cars": {  
    "car1":"Ford",  
    "car2":"BMW"  
  }  
 }

È possibile accedere agli oggetti JSON nidificati usando la notazione dot o la notazione parentesi:

x = myObj.cars.car2;  
// or:  
x = myObj.cars["car2"]; <https://www.w3schools.com/js/tryit.asp?filename=tryjson_object_nested>

**Modifica valori:**

Puoi usare la notazione a punti per modificare qualsiasi valore in un oggetto JSON: myObj.cars.car2="Mercedes"; <https://www.w3schools.com/js/tryit.asp?filename=tryjson_object_modify>

**Elimina proprietà oggetto:**

Usa la parola ***delete*** chiave per cancellare le proprietà da un oggetto JSON:

delete myObj.cars.car2; <https://www.w3schools.com/js/tryit.asp?filename=tryjson_object_delete>

**JSON.parse() -** [**https://www.w3schools.com/js/js\_json\_parse.asp**](https://www.w3schools.com/js/js_json_parse.asp)

Quando si ricevono dati da un server Web, i dati sono sempre una stringa. Analizza i dati con ***JSON.parse()*** e i dati diventano oggetti JavaScript.

*Immagina di aver ricevuto questo testo da un server web:*

'{"name":"John","age":30,"city":"New York"}'

*Usa la funzione JavaScript* ***JSON.parse()*** *per convertire il testo in un oggetto JavaScript:*

var obj = JSON.parse('{ "name":"John", "age":30, "city":"New York"}');

*Usa l'oggetto JavaScript nella tua pagina:*

<p id="demo"></p>   
<script>  
document.getElementById("demo").innerHTML = obj.name + ", " + obj.age;   
</script> <https://www.w3schools.com/js/tryit.asp?filename=tryjson_parse>

**JSON dal server:**

È possibile richiedere JSON dal server utilizzando una richiesta AJAX. Finché la risposta dal server è scritta in formato JSON, è possibile analizzare la stringa in un oggetto JavaScript.

Utilizzare ***XMLHttpRequest*** per ottenere dati dal server:

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", "json\_demo.txt", true);  
xmlhttp.send(); <https://www.w3schools.com/js/tryit.asp?filename=tryjson_ajax>

**Eccezioni:**

Gli oggetti data non sono consentiti in JSON. Se è necessario includere una data, scriverla come una stringa.

*Convertire una stringa in una data:*

var text = '{ "name":"John", "birth":"1986-12-14", "city":"New York"}';  
var obj = JSON.parse(text);  
obj.birth = new Date(obj.birth);  
document.getElementById("demo").innerHTML = obj.name + ", " + obj.birth;

<https://www.w3schools.com/js/tryit.asp?filename=tryjson_parse_date>

**Funzioni**:

Le funzioni non sono consentite in JSON. Se è necessario includere una funzione, scriverla come una stringa.

*Converti una stringa in una funzione:*

var text = '{ "name":"John", "age":"function () {return 30;}", "city":"New York"}';  
var obj = JSON.parse(text);  
obj.age = eval("(" + obj.age + ")");  
document.getElementById("demo").innerHTML = obj.name + ", " + obj.age();

<https://www.w3schools.com/js/tryit.asp?filename=tryjson_parse_function>

**JSON.stringify() -** [**https://www.w3schools.com/js/js\_json\_stringify.asp**](https://www.w3schools.com/js/js_json_stringify.asp)

Quando si inviano dati a un server Web, i dati devono essere una stringa. Convertire un oggetto JavaScript in una stringa con ***JSON.stringify().***

*Immagina di avere questo oggetto in JavaScript:*

var obj = { name: "John", age: 30, city: "New York" };

*Usa la funzione JavaScript* ***JSON.stringify()*** *per convertirla in una stringa.*

var myJSON = JSON.stringify(obj);

***myJSON*** *è ora una stringa e pronta per essere inviata a un server:*

document.getElementById("demo").innerHTML = myJSON;

<https://www.w3schools.com/js/tryit.asp?filename=tryjson_stringify>

**Eccezioni:**

In JSON, gli oggetti data non sono consentiti. La funzione ***JSON.stringify()*** convertirà qualsiasi data in stringhe.

var obj = { name: "John", today: new Date(), city : "New York" };  
var myJSON = JSON.stringify(obj);  
document.getElementById("demo").innerHTML = myJSON;

<https://www.w3schools.com/js/tryit.asp?filename=tryjson_stringify_date>

**Funzioni**:

La funzione ***JSON.stringify()*** rimuoverà qualsiasi funzione da un oggetto JavaScript, sia la chiave che il valore:

var obj = { name: "John", age: function () {return 30;}, city: "New York"};  
var myJSON = JSON.stringify(obj);  
document.getElementById("demo").innerHTML = myJSON;

<https://www.w3schools.com/js/tryit.asp?filename=tryjson_stringify_function>

Questo può essere omesso se converti le tue funzioni in stringhe prima di eseguire la funzione ***JSON.stringify().***

var obj = { name: "John", age: function () {return 30;}, city: "New York"};  
obj.age = obj.age.toString();  
var myJSON = JSON.stringify(obj);  
document.getElementById("demo").innerHTML = myJSON;

**HTML JSON -** [**https://www.w3schools.com/js/js\_json\_html.asp**](https://www.w3schools.com/js/js_json_html.asp)

JSON può essere facilmente tradotto in JavaScript, usato per creare HTML nelle tue pagine web.

**Tabella HTML dinamica:**

Crea la tabella HTML in base al valore di un menu a discesa:

<select id="myselect" onchange="change\_myselect(this.value)">  
  <option value="">Choose an option:</option>  
  <option value="customers">Customers</option>  
  <option value="products">Products</option>  
  <option value="suppliers">Suppliers</option>  
</select>  
<script>  
function change\_myselect(sel) {  
  var obj, dbParam, xmlhttp, myObj, x, txt = "";  
  obj = { table: sel, limit: 20 };  
  dbParam = JSON.stringify(obj);  
  xmlhttp = new XMLHttpRequest();  
  xmlhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
      myObj = JSON.parse(this.responseText);  
      txt += "<table border='1'>"  
      for (x in myObj) {  
        txt += "<tr><td>" + myObj[x].name + "</td></tr>";  
      }  
      txt += "</table>"   
      document.getElementById("demo").innerHTML = txt;  
    }  
  };  
  xmlhttp.open("POST", "json\_demo\_db\_post.php", true);  
  xmlhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");  
  xmlhttp.send("x=" + dbParam);  
}  
</script>

<https://www.w3schools.com/js/tryit.asp?filename=tryjson_html_table_dynamic>

