GABRIELE ROMANATO

Introduzione a JSON: storia ed uso nello sviluppo web

JSON, acronimo per JavaScript Object Notation, è divenuto nel corso degli anni un popolare formato per lo scambio di dati sul web. JSON non è stato creato nel senso letterale del termine, poichè la notazione letterale degli oggetti è una delle caratteristiche già presenti nel linguaggio JavaScript. Si può dire che JSON sia stato riscoperto come formato per lo scambio dati. In questo articolo forniremo un'introduzione a questo formato.

Origine di JSON

JSON è stato proposto come formato per lo scambio dati da Douglas Crockford, il quale ne ha anche definito le specifiche. Egli lo definisce così:

> JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. These properties make JSON an ideal data-interchange language.
>
> JSON (JavaScript Object Notation) è un formato leggero per lo scambio di dati, facile da leggere e scrivere per gli esseri umani e facile da generare e analizzare da parte delle macchine. Si basa su un sottoinsieme del linguaggio di programmazione JavaScript, definito nelle specifiche ECMA-262, 3a edizione, del dicembre 1999. JSON è un formato di testo completamente indipendente dal linguaggio ma che usa convenzioni già familiari ai programmatori dei linguaggi derivati dal C, tra cui C, C++, C#, Java, JavaScript, Perl, Python e molti altri. Queste caratteristiche rendono JSON un linguaggio ideale per lo scambio di dati.

Crockford ha inoltre definito, oltre alla sintassi ed ai tipi di dati JSON, anche il tipo di contenuto con cui JSON deve essere servito nell'RFC 4627.

Sintassi di JSON

JSON prende origine dalla sintassi degli oggetti letterali in JavaScript. Un oggetto letterale può essere definito così:

```
var JSON = {
  proprieta1: "Valore",
  proprieta2: "Valore",
  proprietaN: "Valore"
}
```

Si tratta di coppie di proprietà/valori separate dalla virgola ad eccezione dell'ultima. L'intero oggetto viene racchiuso tra parentesi graffe. A differenza di questa notazione JavaScript, che può contenere anche funzioni e valori complessi, JSON ammette solo valori semplici ed atomici, tra cui:

1. stringhe
2. numeri
3. array
4. oggetti letterali

5. `true`
6. `false`
7. `null`

Un esempio:

```
{
  "home": "test.it",
  "link": "http://www.test.it",
  "argomento": "Lorem Ipsum",
  "aree": [

    {

      "area": "CSS",
      "url": "http://css.test.it"

    },

    {

      "area": "HTML",
      "url": "http://html.test.it"


    }



  ]
}
```

JSON inizia con una coppia di parentesi graffe che racchiudono il corpo dell'intera struttura. Seguono poi le coppie di proprietà e valori che, come in questo esempio, possono contenere i valori atomici elencati sopra. A differenza degli oggetti letterali, JSON richiede che i nomi delle proprietà e i valori stringa siano racchiusi tra doppie virgolette. Per accedere a ciascuno dei membri di un oggetto JSON utilizziamo la tradizionale notazione JavaScript.

JSON viene usato da JavaScript tramite AJAX. Quando la richiesta ha luogo, viene restituito l'oggetto mostrato sopra, cui possiamo accedere con la sintassi precedentemente indicata. Se si utilizza jQuery, possiamo scrivere:

```
"use strict";
$.ajax({
  url: "json.php",
  type: "GET",
  dataType: "json",
  success: function( oggetto ) {

    var sito = oggetto.home;
    var url =  oggetto.link;
    var aree = oggetto.aree;

    //...


  }
});
```

Potete vedere più esempi qui.

Servire JSON

JSON deve essere servito con il content type impostato su `application/json`:

```
<?php
header('Content-Type: application/json');


?>
```

Applicazioni Correlate

- ### JavaScript Password Mask

  Un esempio in JavaScript di mascheramento di una password con l'aggiunta della funzionalità di copia negli appunti.
  DockerDocker ComposeJavaScript

- **Python Placeholder Image**

  Applicazione sviluppata in Python con Flask per la creazione di immagini segnaposto.
  DockerDocker ComposePythonFlaskJavaScript

-

  Go Placeholder Image

  Applicazione in Go per la creazione di immagini segnaposto.
  DockerDocker ComposeGoJavaScript