JSON Validator and Formatter

Code

```
1  myObj = {
2   "name":"John",
3   "age":30,
4   "cars": [
5    { "name":"Ford", "models":[ "Fiesta", "Focus", "Mustang" ] },
6    { "name":"BMW", "models":[ "320", "X3", "X5" ] },
7    { "name":"Fiat", "models":[ "500", "Panda" ] }
8   ]
9  }
```

Validate JSON Clear

Valid JSON

```
for (i in myObj.cars) {
  x += "<h1>" + myObj.cars[i].name + "</h1>";
  for (j in myObj.cars[i].models) {
    x += myObj.cars[i].models[j];
  }
}
```

Invalid JSON

```
{
  "errors": [{
    "status": "400",
    "detail": "JSON parse error -7)."
  }]
}
```

Result

Ford

Fiesta
Focus
Mustang

Know more about JSON

Json

More Developer Tools

About JSONLint?

JSONLint is a validator and reformatter for JSON, a lightweight data-interchange format. Copy and paste,directly type, or input a URL in the editor above and let JSONLint tidy and validate your messy JSON code. JSONLint is a validator and reformatter for JSON, a lightweight data-interchange format. Copy and paste,directly type, or input a URL in the editor above and let JSONLint tidy and validate your messy JSON code.

Tips & Tricks

1

You can directly input a URL into the editor and JSONLint will scrape it for JSON and parse it.

2

You can provide JSON to lint in the URL if you link to JSONLint with the "json" parameter. Here's an example URL to test.

3

JSONLint can also be used as a JSON compressor if you add ?reformat=compress to the URL.

Common Errors

- Expecting 'STRING' - You probably have an extra comma at the end of your collection. Something like { "a": "b", }
- Expecting 'STRING', 'NUMBER', 'NULL', 'TRUE', 'FALSE', '{', '[' - You probably have an extra comma at the end of your list Something like: ["a", "b",]
- Enclosing your collection keys in quotes. Proper format for a collection is { "key": "value" }
- Make sure you follow JSON's syntax properly. For example, always use double quotes, always quotify your keys, and remove all callback functions.

Different Results

If you use a Windows computer you may end up with different results. This is possibly due to the way Windows handles newlines. Essentially, if you have just newline characters (\n) in your JSON and paste it into JSONLint from a Windows computer, it may validate it as valid erroneously since Windows may need a carriage return (\r) as well to detect newlines properly. As a solution, either use direct URL input, or make sure your content's newlines match the architecture your system expects!

Maintained by CircleCell. Thanks to Douglas Crockford of JSON and JS Lint, and Zach Carter, who built a pure JavaScript implementation. You can download the JSONLint source code on GitHub.