**How To Use JSON.parse() and JSON.stringify()**

The [JSON object](https://www.digitalocean.com/community/tutorials/how-to-work-with-json-in-javascript), available in all modern browsers, has two useful methods to deal with JSON-formatted content: parse and stringify.

[**JSON.parse()**](https://www.digitalocean.com/community/tutorials/js-json-parse-stringify#json-parse)

JSON.parse() takes a JSON string and transforms it into a JavaScript object.

let userStr = '{"name":"Sammy","email":"sammy@example.com","plan":"Pro"}';

let userObj = JSON.parse(userStr);

console.log(userObj);

Executing this code will produce the following output:

Output

{name: 'Sammy', email: 'sammy@example.com', plan: 'Pro'}

email: "sammy@example.com"

name: "Sammy"

plan: "Pro"

Trailing commas are not valid in JSON, so JSON.parse() throws an error if the string passed to it has trailing commas.

JSON.parse() can take a function as a second argument that can transform the object values before they are returned.

Here the object’s values are transformed to uppercase in the returned object of the parse method:

let userStr = '{"name":"Sammy","email":"sammy@example.com","plan":"Pro"}';

let userObj = JSON.parse(userStr, (key, value) => {

if (typeof value === 'string') {

return value.toUpperCase();

}

return value;

});

console.log(userObj);

Copy

Executing this code will produce the following output:

Output

{name: 'SAMMY', email: 'SAMMY@EXAMPLE.COM', plan: 'PRO'}

email: "SAMMY@EXAMPLE.COM"

name: "SAMMY"

plan: "PRO"

The values have been transformed to uppercase characters.

[**JSON.stringify()**](https://www.digitalocean.com/community/tutorials/js-json-parse-stringify#json-stringify)

JSON.stringify() takes a JavaScript object and transforms it into a JSON string.

let userObj = {

name: "Sammy",

email: "sammy@example.com",

plan: "Pro"

};

let userStr = JSON.stringify(userObj);

console.log(userStr);

Copy

Executing this code will produce the following output:

Output

{"name":"Sammy","email":"sammy@example.com","plan":"Pro"}

JSON.stringify() can take two additional arguments. The first one is a replacer function. The second is a String or Number value to use as a space in the returned string.

The replacer function can be used to filter out values, as any value returned as undefined will be out of the returned string:

let userObj = {

name: "Sammy",

email: "sammy@example.com",

plan: "Pro"

};

function replacer(key, value) {

console.log(typeof value);

if (key === 'email') {

return undefined;

}

return value;

}

let userStrReplacer = JSON.stringify(userObj, replacer);

console.log(userStrReplacer);

Copy

Executing this code will produce the following output:

Output

{"name":"Sammy","plan":"Pro"}

The email key-value pair has been removed from the object.

And an example with a space argument passed-in:

let userObj = {

name: "Sammy",

email: "sammy@example.com",

plan: "Pro"

};

let userStrSpace = JSON.stringify(user, null, '...');

console.log(userStrSpace);

Copy

Executing this code will produce the following output:

Output

{

..."name": "Sammy",

..."email": "sammy@example.com",

..."plan": "Pro"

}