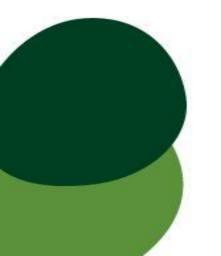


JavaScript进阶

---JSON对象





内容提纲

- **➢ JSON简介**
- > JSON对象方法
- **➢ JSON案例**



JSON简介

•JSON(JavaScript Object Notation, JS 对象标记)

- JSON是一种轻量级的数据交换格式(其他数据格式,如XML等,也可作为数据的载体)
- JSON的使用完全独立于编程语言的文本格式来存储和传输数据
- 简洁和清晰的层次结构使得 JSON 成为理想的数据交换格式
- 易于读写和解析,并可有效地提升网络传输效率(轻量)

•JSON与JS对象的关系

- JS中一切都是对象(如何理解这句话),任何支持的类型都可以通过对象来表示
- JSON 是 JS 对象的字符串形式的表示法,使用文本表示 JS 对象的信息,本质是一个字符串

```
var obj = {a: 'Hello', b: 'World'}; //这是一个对象,注意键名也是可以使用引号包裹的 var json = '{"a": "Hello", "b": "World"}'; //这是一个 JSON 字符串,本质是一个字符串
```



JSON简介

•JSON实例(JSON练习: http://www.json.cn/)

```
var obj1 = "xxx";
var obj2 = 23;
var obj3 = false;
var obj4 = { x: 1, y: 2, a: [1, 3, 5], b: "xyz" };
var obj5 = [123, 345];
var obj6 = [{ z: 3 }, [1, 2]];
var obj7 = {x:true};
var json1 = '"xxx"';
var json2 = '23';
var json3 = 'false';
var json4 = '{"x":1,"y":2,"a":[1,3,5],"b":"xyz"}';
var json5 = '[123,345]';//注意区别于: '["123","345"]'
var json6 = '[{"z":3},[1,2]]';
var json7 = '{"x":true}';//注意区别于: '{"x":"true"}'
```

内容提纲

- **➢ JSON简介**
- > JSON对象方法
- **➢ JSON案例**



JSON对象方法(基本用法)

JSON.stringify(value [,replacer [,space]])

```
var a1 = [1,"x",true,{y:2,z:3}];
var jsonStrArr1 = JSON.stringify(a1);
console.log(jsonStrArr1);//'[1,"x",true,{"y":2,"z":3}]'
console.log(a1);//[1, "x", true, Object]
```

JSON.parse(text [,reviver])

```
var jsonStr3 = '{"a":[1,2],"b":true,"c":[3,4,"x",{"y":34,"z":56}]}';
var jsonStr4 = '[1,"x",true,{"y":2,"z":3}]';
var o3 = JSON.parse(jsonStr3);//{a: Array(2), b: true, c: Array(4)}
var o4 = JSON.parse(jsonStr4);//[1, "x", true, Object]
```



JSON对象方法(进阶)

•JSON.stringify(value [,replacer [,space]])

```
function replacer(key, value) {
    if (typeof value === "string") {
        return undefined;
    return value;
var foo = {model: "box",
           week: 45,
           transport: "car",
           month: 7 };
var jsonString1 = JSON.stringify(foo, replacer);
console.log(jsonString1);
```



JSON对象方法(进阶)

JSON.parse(text [,reviver])

```
var o7 = JSON.parse('{"p": 5,"x":1}', function (k, v) {
   if(k === 'p') return 2*v; //
   if(k === 'x') return 3*v;
   if(k === '') return v; //最终到达顶层
});
console.log(o7);
var o8 = JSON.parse('{"1": 1, "2": 2,"3": {"4": 4, "5": {"6": 6}}}',
   function (k, v) {
       console.log(k); // 输出当前的属性名,从而得知遍历顺序是从内向外的,
      // 最后一个属性名会是个空字符串。
       return v; // 返回原始属性值,相当于没有传递 reviver 参数。
   });
console.log(o8);
```

内容提纲

- **➢ JSON简介**
- > JSON对象方法
- **➢ JSON案例**



JSON案例

·JSON案例-前后端之间的数据传输

```
var fs = require("fs");
var http = require("http");
var url = require("url");
http.createServer(function (req, res) {
    var getDataObj = url.parse(req.url,true).query;
    var arrayIndex = getDataObj.id-1;
    console.log(typeof arrayIndex,arrayIndex);
    fs.readFile("./NodeJsonTest.json", function readData(err, data) {
        res.writeHead(200, {"Content-Type": "application/json",
            "Access-Control-Allow-Origin": "*",
            "Access-Control-Allow-Methods": "GET, POST"
        });
        res.end(JSON.stringify(JSON.parse(data)[arrayIndex]));
    });
```

作业

- •复习本章内容
- 查看深入理解JavaScript中的JSON章节





