arguments对象

2020年03月04日, 星期三 14:24

1.JavaScript 函数在定义时有固定数目的命名参数，但当调用这个函数时，传递给它的参数数目却可以是任意的。

```
> function fun(a,b){
      console.log(a);
      console.log(b);
  }
  fun(1);
  1
  undefined
< undefined
> fun(1,2);
  1
  2
< undefined
> fun(1,2,3);
  1
  2
< undefined
```

2.arguments对象

代表**传入函数的实参**

是函数中的局部变量

不能显式创建，只有函数调用时才可用

它是一个类数组对象

类数组对象

与数组一样具有 **length 与 index 属性**

本质确实个 Object

3.

```
> function fun(a,b){
      console.log(a);
      console.log(b);
      console.dir(arguments);
  }
  fun(1,2,3);
  1
  2
  ▼Arguments(3) i
     0: 1
     1: 2
     2: 3
     length: 3
   ▶ callee: ƒ fun(a,b)
   ▶ Symbol(Symbol.iterator): ƒ values()
   ▶ __proto__: Object
< undefined
```

补充：console.log()会在浏览器控制台打印出信息

console.dir()可以显示一个对象的所有属性和方法

4.arguments与形参的“**双向绑定**”特性

```
function fun(a, b, c) {
    console.log(a, b, c);
    console.dir(arguments);
    console.log(a===arguments[0]);  // true
    console.log(b===arguments[1]);  // true
    console.log(c===arguments[2]);  // true
}
var obj = { x: 1, y: 2};
fun(1, obj);
```

```
function fun(a, b, c) {
    console.log(a === arguments[0]);  // true
    a = 2;
    console.log(a === arguments[0]);  // true
    arguments[0] = 3;
    console.log(a === arguments[0]);  // true
}
var obj = { x: 1,y: 2 };
fun(1, obj);
```

```
function fun(a, b, c) {
    console.log(b === arguments[1]);  // true
    b = [1, 2, 3];
    console.log(b === arguments[1]);  // true
    arguments[1] = [2, 3, 4];
    console.log(b === arguments[1]);  // true
}
var obj = { x: 1, y: 2 };
fun(1, obj);
console.log(obj); // [2,3,4]
```

```
function fun(a, b, c) {
    console.log(c === arguments[2]);  // true
    c = 2;
    console.log(arguments[2]);        // undefined
    console.log(c === arguments[2]);  // false
    arguments[2] = 3;
    console.log(c);                   // 2
    console.log(c === arguments[2]);  // false
}
var obj = { x: 1,y: 2 };
fun(1, obj);
```

双向绑定特性：

在调用时 arguments 对象与**实际传递了值的形参**变量发生**双向绑定**

arguments 对象中的对应单元会和命名参数建立关联

5.arguments 的 length 属性

表示函数调用时传入的实参数量

在调用时，实参个数确定， arguments.length 确定， 不会再发生改变。

```
function fun(a, b, c) {
    console.log(arguments.length);  // 2
    arguments[4] = 1;
    console.log(arguments.length);  // 2
    return a + b;
}
var obj = { x: 1, y: 2 };
fun(1, obj);
```

```
function fun(a, b, c) {
        console.log(arguments.length);
        arguments[4] = 1;
        console.log(arguments.length); // arguments只与传入的实参有关系
        console.log(arguments);
        return a + b;
      }
      var obj = { x: 1, y: 2 };
      fun(1, obj);
```

```
2                                                          VM408:2
2                                                          VM408:4
▼Arguments(2) ℹ                                            VM408:5
    0: 1
  ▶ 1: {x: 1, y: 2}
    4: 1
    length: 2
  ▶ callee: ƒ fun(a, b, c)
  ▶ Symbol(Symbol.iterator): ƒ values()
  ▶ __proto__: Object
"1[object Object]"
```