JS中数据类型分2种

原始类型：boolen/number/string/null/undefined。原始类型的变量直接保存原始值，也就是说当你将原始值赋值给一个变量时，该值被复制到变量中（在内存中新用一个位置），每一个变量都有它自己的一份数据拷贝。

|  |
| --- |
| 栈内存 |
| Color1 red |
| color2 blue |
|  |
|  |
|  |
|  |

var color1 = ‘red’;

var color2 = color1;

console.log(color1); //red

console.log(color2); //red

color2 = ‘blue’; //对color1没有影响

console.log(color1); //red

console.log(color2); //blue

引用类型：function/object/数组。不在变量中保存对象，object变量实际上并不包含对象的实例，而是一个指向内存中实际对象所在位置的指针（引用），这就意味着将一个对象赋值给变量时，实际上赋值给这个变量的是一个指针，这两个变量指向内存中的同一个对象

var object = new Object();

|  |
| --- |
| 栈内存 |
| S2堆地址 |
| S1堆地址 |
| func |
|  |
|  |
|  |

堆内存

function func(){

made in china

var S1 = [“made in china”];

var S2 = S1;

function func(){

var S1 = [“made in china”];

var S2 = S1;

console.log(S2[0]);

S1[0] = “made in USA”;

console.log(S2[0]);

}

console.log(S2[0]);

S1[0] = “made in USA”;

console.log(S2[0]);

}

func();

var person1 = {name:’pschen’,age:40,};

堆内存

var person2 = { name:’ellen’,age:40,};

|  |
| --- |
| 栈内存 |
| person1 |
| person2 |
| Color1 red |
| color2 blue |
| obj1 |
| obj |

var obj = new Object();

object

var obj1 = obj;

|  |
| --- |
| name |
| age |
| addr |
| love |
| … |

|  |
| --- |
| name |
| age |
| addr |
| love |
| … |

当查询引用类型的变量时， 先从栈中读取内存地址， 然后再通过地址找到堆中的值。对于这种，我们把它叫做按引用访问