数组方法2/类数组/for in

2017年7月18日 16:53

```
        window.onload = function(){
        // 类数组
        var lis = document.getElementsByTagName("li");
/*          // 类数组转成数组
        var newArr = Array.from(lis);
*/
        var newArr = Array.from(lis);

        alert( Array.isArray(newArr) )  // 转完之后返回的是一个数组

        Array.from(lis).forEach(function (item){
            item.onclick = function (){
                alert(1)
            };
        })


};
```

Array.from()

作用

把类数组转成一个新的数组

语法

array Array.from(arrayLike[, mapFn[, thisArg]])

参数

arrayLike

想要转换成真实数组的类数组

mapFn

生成的数组会经过该函数的加工处理后再返回

```
window.onload = function(){
// 类数组
var lis =
document.getElementsByTagName("li");
// 第二个参数是一个函数，对类数组中的每一项做进一步处理，
// 处理之后把return后面的值放在新数组
```

mapFn
生成的数组会经过该函数的加工处理后再返回
thisArg 可选参数
用来当作callback 函数内this的值。
返回值
新的数组

```
// 第二个参数是一个函数，对类数组中的每一项做进一步处理，
// 处理之后把return后面的值放在新数组中。
var newArr = Array.from(lis,function (item){
    console.log(item);
    console.log(this);
    return item.innerHTML;
},["hello"])

console.log(newArr);
};
```

Array.isArray()

作用
检测值是否是一个数组
语法
boolean Array.isArray(obj)
参数
obj
需要检测的值。
返回值
布尔值
true，是一个数组
false，不是一个数组

循环对象

```
var obj = {
    "miaov":"ketang",
    abc:"123",
    a:1,
```

for循环循环对象：
对象要是连续的数字为key值
手动的添加length
var obj = {

```
        abc:"123",
        a:1,
        b:2,
        c:3
    }
```

对象中的属性名，可以加""也可以不用加

```
        //console.log(obj.miaov);

        // 循环对象 通过属性名那属性值
        /*
            for in  循环
            语法：
                for(变量 in 对象){
                    // 循环体
                }

                每一次循环的时候，都会把对象的key值赋值给变量

                break
                continue

            循环数组 for
            循环对象 for in

            对象中的key类型是字符串
        */

        /*for(var attr in obj){
            console.log(attr+":"+ obj[attr]);
        }*/
```

手动的添加length

```
var obj = {
        0:"a",
        1:"b",
        2:"c",
        length:3
}
// 对象没有length这个属性
console.log(obj.length);
for( var i = 0; i < obj.length; i++ ){
        console.log(obj[i])
}
```

```
// 循环数组

var arr = ["a","b","c","d"];

for(var attr in arr){
    console.log(typeof attr,attr,arr[attr]);
    break;
}
```

多属性运动：

```
function mTween(element,attrObj,duration,fx,callback){
        // 要算出来每一个样式的起始位置和总距离
        // 循环对象attrObj，算出每一个用时的起始位置和总距离

        var beginObj = {}; // 每一个样式的起始位置
        var countObj = {}; // 每一个样式的总距离
        for(var attr in attrObj){
            beginObj[attr] = parseFloat(getComputedStyle(element)
            [attr]);
            // 每一个样式要运动的的总距离
            countObj[attr] = attrObj[attr] - beginObj[attr];
        }

        // 开始运动的时间
        var startTime = Date.now();

        fx = fx || 'linear';

        clearInterval(element.timer);
```

```
    element.timer = setInterval(function (){
        // 已过去时间
        var t = Date.now() - startTime;

        if(t >= duration){
            t = duration;
            clearInterval(element.timer);
        }

        // 循环传过来的对象,要运动的是对象的key值这个样式

        for(var prop in attrObj){

            //判断属性是不是改变透明读的样式名
            if(prop === 'opacity'){
                element.style[prop] = Tween[fx]
                (t,beginObj[prop],countObj[prop],duration);
            }else{
                element.style[prop] = Tween[fx]
                (t,beginObj[prop],countObj[prop],duration) +
                'px';
            }

        }



        if(t === duration){
            typeof callback === 'function' && callback();
        }

    },4)


}
```