# console

## 记时

console.time("Array initialize");var array= new Array(1000000);for (var i = array.length - 1; i >= 0; i--) {

array[i] = new Object();

};console.timeEnd("Array initialize");

## 计数

会计数一个变量值调用的次数

var a = ‘gogo’

function(){

console.count(a) gogo: 1

}

## 表格输出

console.table([{a:1, b:2, c:3}, {a:"foo", b:false, c:undefined}]);

console.table([[1,2,3], [2,3,4]]);

## 打印调用栈

[console.trace()](http://www.css88.com/doc/chrome-devtools/console/console-reference" \l "consoletraceobject" \t "http://www.css88.com/doc/chrome-devtools/console/track-exceptions/_blank)

## 断言输出

console.assert(condition, msg)

console.assert(a > b, {"message":"a is not greater than b","a":a,"b":b});

try catch（e）

e.stack --- > 每个Error对象都一有一个名为stack的属性，包含堆栈跟踪信息。

# window.onerror

如果错误没有被try - catch 捕捉，这个函数就会被调用

window.onerror = function(msg, url, line){

}

# 对象上的事件

## 监视对象事件触发

monitorEvents(document.body, "click");

或监视所有事件

monitorEvents(document.body);

unmonitorEvents(document.body);

## 返回对象上绑定的事件监听器

getEventListeners(document);

## 直接在Element面板中查看

在element面板上点击相关元素，右侧显示所有事件