

Reference Types

fanzhongkai@baidu.com

*Object

- {} equals new Object()
- use named arguments for those that are required and an object literal to encompass multiple optional arguments
- a.b equals a['b']

Array(1)

- [] equals new Array()
- array.length (not read-only)
- detecting
 - instanceof
 - isArray
 - Object.prototype.toString.call(array);

Array(2)

- conversion: toString, toLocaleString, valueOf
- stack method: pop, push
- queue method: shift, unshift
- reordering method: reverse, sort
- manipulation method: concat, slice, splice
- location method: indexOf, lastIndexOf
- iterative method: every, filter, forEach, map, some
- reduction method: reduce, reduceRight

Date & RegExp

Bypass

*Function

• Function Declarations vs Function Expressions

```
alert(sum(10,10));  
function sum(num1, num2){  
    return num1 + num2;  
}
```

```
alert(sum(10,10));  
var sum = function(num1, num2){  
    return num1 + num2;  
};
```

function declaration hoisting

• Functions as values

• Function internals: callee, caller

```
function factorial(num){  
    if (num <= 1) {  
        return 1;  
    } else {  
        return num * factorial(num-1)  
    }  
}
```

• Function properties and methods:

• length, prototype

• call, apply, **bind**

*Call & Apply

```
function sum(num1, num2){ return num1 + num2;
}
function callSum(num1, num2){
    return sum.call(this, num1, num2);
}
alert(callSum(10,10)); //20
```

```
window.color = "red";
var o = { color: "blue" };
function sayColor(){ alert(this.color);
}
sayColor(); //red
sayColor.call(this); //red
sayColor.call(window); //red
sayColor.call(o); //blue
```

```
function sum(num1, num2){ return num1 + num2;
}
function callSum1(num1, num2){
    return sum.apply(this, arguments);
}
//passing in arguments object
function callSum2(num1, num2){
    return sum.apply(this, [num1, num2]); //passing in array
}
alert(callSum1(10,10)); //20 alert(callSum2(10,10)); //20
```


Bind


```
window.color = "red";  
var o = { color: "blue" };  
function sayColor(){ alert(this.color);  
}  
var objectSayColor = sayColor.bind(o);  
objectSayColor(); //blue
```


Singleton Built-in Object

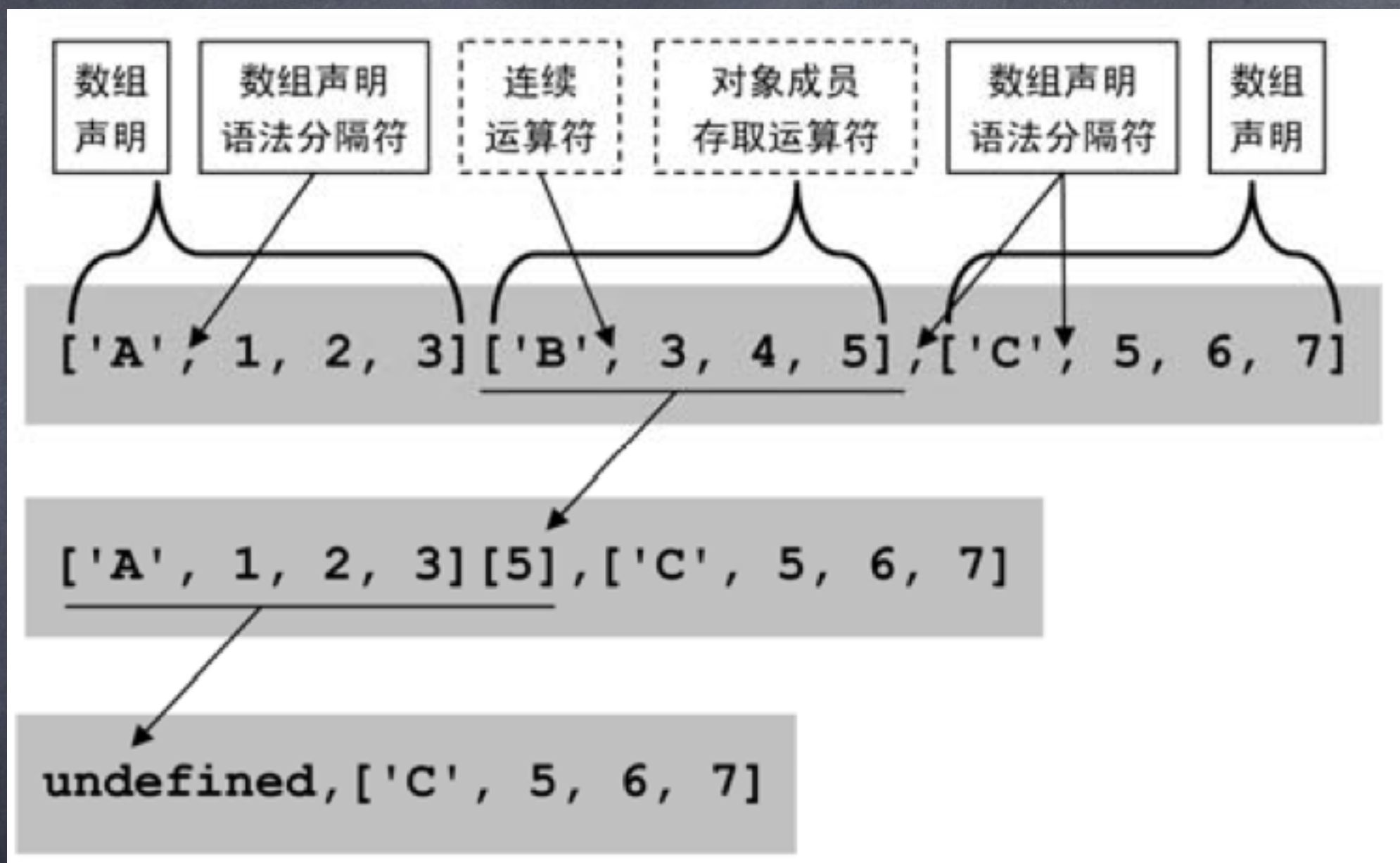
- Global (window)
- Math

二义性

	运算符/符号	运算符含义	其他含义
具有二义性的运算符	,	连续运算符	参数分隔符 对象/数组声明分隔符
	+	增值运算符 正值运算符 连接运算符	数字直接量声明 (正、负或指数形式)
	()	函数调用运算符	强制运算(优先级) 形式参数表
	?:	条件运算符	: 号有声明标签的含义 : 号有声明 switch 分支的含义 : 号有声明对象成员的含义
	[]	数组下标 对象成员存取	数组直接量声明
其他	{ }		函数直接量(代码部分的)声明 对象直接量声明 复合语句
	;		空语句 语句分隔符



```
var table = [  
  ['A', 1, 2, 3]  
  ['B', 3, 4, 5],  
  ['C', 5, 6, 7]  
];
```

Web Resources

- bower.io
- [github](https://github.com)
- [stackoverflow](http://stackoverflow.com)
- fex.baidu.com
- source code

Books

- JavaScript语言精髓与编程实践（第二版）
- JavaScript模式
- JavaScript设计模式
- 高性能JavaScript
- ECMAScript 6入门