## 字符串

1. 下面的连接创建了一个连接用户单击时会提交表单：

<A HREF="javascript:void(document.form.submit())">单此处提交表单</A>

1. isNaN(NaN); //唯一能判断NaN的方法是通过isNaN()函数
2. 大多数情况下，我们都应该用null。undefined仅仅在判断函数参数是否传递的情况下有用。
3. JavaScript的数组可以包括任意数据类型。
4. 要获取一个对象的属性，我们用对象变量.属性名的方式
5. 变量名是大小写英文、数字、$和\_的组合，且不能用数字开头。
6. 这种变量本身类型不固定的语言称之为动态语言
7. 如果一个变量没有通过var申明就被使用，那么该变量就自动被申明为全局变量
8. 启用strict模式的方法是在JavaScript代码的第一行写上：'use strict';
9. 由于多行字符串用\n写起来比较费事，所以最新的ES6标准新增了一种多行字符串的表示方法，用反引号 ` ... ` 表示：
10. var message = `你好, ${name}, 你今年${age}岁了!`;
11. 需要特别注意的是，字符串是不可变的，如果对字符串的某个索引赋值，不会有任何错误，但是，也没有任何效果：
12. toUpperCase()把一个字符串全部变为大写：  
    toLowerCase()把一个字符串全部变为小写：  
    indexOf()会搜索指定字符串出现的位置：(没有找到指定的子串，返回-1)  
    substring()返回指定索引区间的子串：

## 数组

1. 请注意，直接给Array的length赋一个新的值会导致Array大小的变化：
2. 请注意，如果通过索引赋值时，索引超过了范围，同样会引起Array大小的变化：
3. slice()就是对应String的substring()版本，它截取Array的部分元素，然后返回一个新的Array：
4. push()向Array的末尾添加若干元素，pop()则把Array的最后一个元素删除掉：
5. 如果要往Array的头部添加若干元素，使用unshift()方法，shift()方法则把Array的第一个元素删掉：
6. sort()可以对当前Array进行排序，它会直接修改当前Array的元素位置，直接调用时，按照默认顺序
7. reverse()把整个Array的元素给掉个个，也就是反转：
8. splice()方法是修改Array的“万能方法”，它可以从指定的索引开始删除若干元素，然后再从该位置添加若干元素：
9. concat()方法可以接收任意个元素和Array，并且自动把Array拆开，然后全部添加到新的Array里：
10. join()方法是一个非常实用的方法，它把当前Array的每个元素都用指定的字符串连接起来，然后返回连接后的字符串：

## 对象

1. 访问属性是通过.操作符完成的，但这要求属性名必须是一个有效的变量名。如果属性名包含特殊字符，就必须用''括起来
2. xiaohong的属性名middle-school不是一个有效的变量，就需要用''括起来。访问这个属性也无法使用.操作符，必须用['xxx']来访问
3. 实际上JavaScript对象的所有属性都是字符串
4. 如果访问一个不存在的属性会返回什么呢？JavaScript规定，访问不存在的属性不报错，而是返回undefined：
5. 由于JavaScript的对象是动态类型，你可以自由地给一个对象添加或删除属性：
6. 如果我们要检测xiaoming是否拥有某一属性，可以用in操作符：
7. 因为toString定义在object对象中，而所有对象最终都会在原型链上指向object，所以xiaoming也拥有toString属性
8. 要判断一个属性是否是xiaoming自身拥有的，而不是继承得到的，可以用hasOwnProperty()方法：
9. 请注意，if...else...语句的执行特点是二选一，在多个if...else...语句中，如果某个条件成立，则后续就不再继续判断了。  
   eg:

var age = 20;

if (age >= 6) {

alert('teenager');

} else if (age >= 18) {

alert('adult');

} else {

alert('kid');

}

结果显示teenager

1. JavaScript把null、undefined、0、NaN和空字符串''视为false，其他值一概视为true
2. var bmi = weight/Math.pow(height, 2);
3. for循环的一个变体是for ... in循环，它可以把一个对象的所有属性依次循环出来：
4. 请注意，for ... in对Array的循环得到的是String而不是Number

## Map和Set

1. 但是JavaScript的对象有个小问题，就是键必须是字符串。
2. Map eg:
   1. var m = new Map([['Michael', 95], ['Bob', 75], ['Tracy', 85]]);
   2. m.set('Adam', 67); // 添加新的key-value
   3. m.get('Michael'); // 95
   4. m.has('Adam'); // 是否存在key 'Adam': true
   5. m.delete('Adam'); // 删除key 'Adam'
3. Set eg:
   1. var s = new Set([1, 2, 3, 3, '3']);
   2. s.add(4);
   3. s.delete(3);

## iterable

1. 遍历Array可以采用下标循环，遍历Map和Set就无法使用下标。为了统一集合类型，ES6标准引入了新的iterable类型，Array、Map和Set都属于iterable类型。
2. 具有iterable类型的集合可以通过新的for ... of循环来遍历。
3. for ... in循环由于历史遗留问题，它遍历的实际上是对象的属性名称。一个Array数组实际上也是一个对象，它的每个元素的索引被视为一个属性。
4. 使用iterable内置的forEach方法，它接收一个函数，每次迭代就自动回调该函数。以Array为例：

var a = ['A', 'B', 'C'];

a.forEach(function (element, index, array) {

// element: 指向当前元素的值

// index: 指向当前索引

// array: 指向Array对象本身

alert(element);

});

1. Set与Array类似，但Set没有索引，因此回调函数的前两个参数都是元素本身：
2. Map方法中的callback回调函数参数依次为value、key和map本身：

## 函数

1. 如果没有return语句，函数执行完毕后也会返回结果，只是结果为undefined。
2. 由于JavaScript的函数也是一个对象，上述定义的abs()函数实际上是一个函数对象，而函数名abs可以视为指向该函数的变量。
3. JavaScript还有一个免费赠送的关键字arguments，S6标准引入了rest参数，上面的函数可以改写为：
4. ES6标准引入了rest参数，eg:

function foo(a, b, ...rest) {

console.log('a = ' + a);

console.log('b = ' + b);

console.log(rest);

}它只在函数内部起作用，并且永远指向当前函数的调用者传入的所有参数。

## 作用域

1. 由于JavaScript的函数可以嵌套，此时，内部函数可以访问外部函数定义的变量，反过来则不行：
2. JavaScript的函数定义有个特点，它会先扫描整个函数体的语句，把所有申明的变量“提升”到函数顶部。

eg：

'use strict';

function foo() {

var x = 'Hello, ' + y;

alert(x);

var y = 'Bob';

}

foo();

虽然是strict模式，但语句var x = 'Hello, ' + y;并不报错，原因是变量y在稍后申明了。

但是alert显示Hello, undefined，说明变量y的值为undefined。这正是因为JavaScript引擎自动提升了变量y的声明，但不会提升变量y的赋值。

1. 实际上，JavaScript默认有一个全局对象window，全局作用域的变量实际上被绑定到window的一个属性
2. 全局变量会绑定到window上，不同的JavaScript文件如果使用了相同的全局变量，或者定义了相同名字的顶层函数，都会造成命名冲突，并且很难被发现。

减少冲突的一个方法是把自己的所有变量和函数全部绑定到一个全局变量中。

1. 为了解决块级作用域，ES6引入了新的关键字let，用let替代var可以申明一个块级作用域的变量：
2. ES6标准引入了新的关键字const来定义常量，const与let都具有块级作用域：

#### 方法

1. 也是不行的！要保证this指向正确，必须用obj.xxx()的形式调用！
2. apply()类似的方法是call()，唯一区别是：

apply()把参数打包成Array再传入；

call()把参数按顺序传入。

eg:调用Math.max(3, 5, 4)，分别用apply()和call()实现如下：

Math.max.apply(null, [3, 5, 4]); // 5

Math.max.call(null, 3, 5, 4); // 5

## 高阶函数

1. JavaScript的函数其实都指向某个变量。既然变量可以指向函数，函数的参数能接收变量，那么一个函数就可以接收另一个函数作为参数，这种函数就称之为高阶函数。
2. 再看reduce的用法。Array的reduce()把一个函数作用在这个Array的[x1, x2, x3...]上，这个函数必须接收两个参数，reduce()把结果继续和序列的下一个元素做累积计算，其效果就是：[x1, x2, x3, x4].reduce(f) = f(f(f(x1, x2), x3), x4)

#### [map/reduce](1.%09http:/www.liaoxuefeng.com/wiki/001434446689867b27157e896e74d51a89c25cc8b43bdb3000/001435119854495d29b9b3d7028477a96ed74db95032675000#0) filter()

sort()

1. Array的sort()方法默认把所有元素先转换为String再排序，结果'10'排在了'2'的前面，因为字符'1'比字符'2'的ASCII码小
2. 按数字大小排序

var arr = [10, 20, 1, 2];

arr.sort(function (x, y) {

if (x < y) {

return -1;

}

if (x > y) {

return 1;

}

return 0;

}); // [1, 2, 10, 20]

1. 倒序排序

var arr = [10, 20, 1, 2];

arr.sort(function (x, y) {

if (x < y) {

return 1;

}

if (x > y) {

return -1;

}

return 0;

}); // [20, 10, 2, 1]

1. 忽略大小写来比较两个字符串

var arr = ['Google', 'apple', 'Microsoft'];

arr.sort(function (s1, s2) {

x1 = s1.toUpperCase();

x2 = s2.toUpperCase();

if (x1 < x2) {

return -1;

}

if (x1 > x2) {

return 1;

}

return 0;

}); // ['apple', 'Google', 'Microsoft']

1. 最后友情提示，sort()方法会直接对Array进行修改，它返回的结果仍是当前Array：

## 闭包

1. 相关参数和变量都保存在返回的函数中，这种称为“闭包（Closure）”的程序结构拥有极大的威力
2. [脑洞大开那块不懂](http://www.liaoxuefeng.com/wiki/001434446689867b27157e896e74d51a89c25cc8b43bdb3000/00143449934543461c9d5dfeeb848f5b72bd012e1113d15000#0)

## 箭头函数

1. 箭头函数相当于匿名函数由于
2. this在箭头函数中已经按照词法作用域绑定了，所以，用call()或者apply()调用箭头函数时，无法对this进行绑定，即传入的第一个参数被忽略：

## generator（生成器）

generator跟函数很像，定义如下：

function\* foo(x) {

yield x + 1;

yield x + 2;

return x + 3;

}

## 标准对象

typeof 123; // 'number'

typeof NaN; // 'number'

typeof 'str'; // 'string'

typeof true; // 'boolean'

typeof undefined; // 'undefined'

typeof Math.abs; // 'function'

typeof null; // 'object'

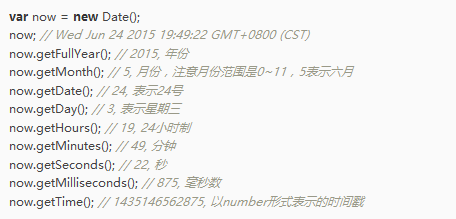
typeof []; // 'object'

typeof {}; // 'object'

1. 虽然包装对象看上去和原来的值一模一样，显示出来也是一模一样，但他们的类型已经变为object了！所以，包装对象和原始值用===比较会返回false：
2. 
3. 

## Date

1. 要获取系统当前时间



1. 你可能观察到了一个非常非常坑爹的地方，就是JavaScript的月份范围用整数表示是0~11，0表示一月，1表示二月……，所以要表示6月，我们传入的是5！
2. 创建日期：
   1. var d = new Date(2015, 5, 19, 20, 15, 30, 123);
   2. 第二种创建一个指定日期和时间的方法是解析一个符合ISO 8601格式的字符串

var d = Date.parse('2015-06-24T19:49:22.875+08:00');

d; // 1435146562875

但它返回的不是Date对象，而是一个时间戳。不过有时间戳就可以很容易地把它转换为一个Date：

var d = new Date(1435146562875);

d; // Wed Jun 24 2015 19:49:22 GMT+0800 (CST)

1. 要获取当前时间戳，可以用：

if (Date.now) {

alert(Date.now()); // 老版本IE没有now()方法

} else {

alert(new Date().getTime());

}

## RegExp

1. 用\d可以匹配一个数字，\w可以匹配一个字母或数字，.可以匹配任意字符A|B可以匹配A或B，所以(J|j)ava(S|s)cript可以匹配'JavaScript'、'Javascript'、'javaScript'或者'javascript'。
2. 用\*表示任意个字符（包括0个），用+表示至少一个字符，用?表示0个或1个字符，用{n}表示n个字符，用{n,m}表示n-m个字符：
3. ^表示行的开头，^\d表示必须以数字开头。
4. $表示行的结束，\d$表示必须以数字结束。
5. 要匹配变长的字符，在正则表达式中，用\*表示任意个字符（包括0个），用+表示至少一个字符，用?表示0个或1个字符，用{n}表示n个字符，用{n,m}表示n-m个字符：
6. var re = /^\d{3}\-\d{3,8}$/;

re.test('010-12345'); // true test()方法用于测试给定的字符串是否符合条件

1. 'a b c'.split(/\s+/); // ['a', 'b', 'c'] //无论多少个空格都可以正常分割。
2. 除了简单地判断是否匹配之外，正则表达式还有提取子串的强大功能。用()表示的就是要提取的分组（Group）。
3. JavaScript的正则表达式还有几个特殊的标志，最常用的是g，表示全局匹配：(全局匹配类似搜索)

eg: var r1 = /test/g;

// 等价于:

var r2 = new RegExp('test', 'g');

1. 正则表达式还可以指定i标志，表示忽略大小写，m标志，表示执行多行匹配。
2. 正则表达式中需要转义的特殊字符（15个）： ^ $ | ? . \* + { } [ ] ( ) / \
3. 正则表达式可以用.\*匹配任意除换行之外的字符。但如果内容中包含任意多个换行怎么匹配？用.\*是匹配失败的。但可以用[\s\S]\*?。\s是匹配所有空白字符，\S是匹配所有非空白字符，那么[\s\S]这个组合就可以匹配所有字符了。

## JSON

1. JSON还定死了字符集必须是UTF-8，表示多语言就没有问题了。为了统一解析，JSON的字符串规定必须用双引号""，Object的键也必须用双引号""。
2. 要输出得好看一些，可以加上参数，按缩进输出：

JSON.stringify(xiaoming, null, ' ');

1. 第二个参数用于控制如何筛选对象的键值，如果我们只想输出指定的属性，可以传入Array：

JSON.stringify(xiaoming, ['name', 'skills'], ' ');

1. 拿到一个JSON格式的字符串，我们直接用JSON.parse()把它变成一个JavaScript对象：

# 面向对象编程

1. JavaScript的原型链和Java的Class区别就在，它没有“Class”的概念，所有对象都是实例，所谓继承关系不过是把一个对象的原型指向另一个对象而已。

eg: xiaoming.\_\_proto\_\_ = Bird;

## 浏览器

1. 如果引入的第三方的JavaScript中存在恶意代码，则第三方网站将直接获取到当前网站的用户登录信息。

为了解决这个问题，服务器在设置Cookie时可以使用httpOnly，设定了httpOnly的Cookie将不能被JavaScript读取。这个行为由浏览器实现，主流浏览器均支持httpOnly选项，IE从IE6 SP1开始支持。

为了确保安全，服务器端在设置Cookie时，应该始终坚持使用httpOnly。

## 提交表单

1. 两种方式：方式一是通过<form>元素的submit()方法提交一个表单，例如，响应一个<button>的click事件，在JavaScript代码中提交表单

第二种方式是响应<form>本身的onsubmit事件，在提交form时作修改：

1. 在检查和修改<input>时，要充分利用<input type="hidden">来传递数据。
2. 没有name属性的<input>的数据不会被提交。

## Ajax

1. 在现代浏览器上写AJAX主要依靠XMLHttpRequest对象：
2. 对于低版本的IE，需要换一个ActiveXObject对象：
3. XMLHttpRequest对象的open()方法有3个参数，第一个参数指定是GET还是POST，第二个参数指定URL地址，第三个参数指定是否使用异步，默认是true，所以不用写。
4. 默认情况下，JavaScript在发送AJAX请求时，URL的域名必须和当前页面完全一致。
5. 完全一致的意思是，域名要相同（www.example.com和example.com不同），协议要相同（http和https不同），端口号要相同（默认是:80端口，它和:8080就不同）。

## Canvas

1. Canvas是HTML5新增的组件，它就像一块幕布，可以用JavaScript在上面绘制各种图表、动画等。没有Canvas的年代，绘图只能借助Flash插件实现，页面不得不用JavaScript和Flash进行交互。有了Canvas，我们就再也不需要Flash了，直接使用JavaScript完成绘制。
2. 由于浏览器对HTML5标准支持不一致，所以，通常在<canvas>内部添加一些说明性HTML代码，如果浏览器支持Canvas，它将忽略<canvas>内部的HTML，如果浏览器不支持Canvas，它将显示<canvas>内部的HTML：