JavaScript高级程序设计笔记

1.3个函数可以把非数值转换为数值：Number(),parseInt()和parseFloat()。

第一个函数，即转型函数Number()可以用于任何数据类型，而另两个函数则专门用于把字符串转换成数值。

1. 用parseInt()转换空字符串会返回NaN.
2. \n 换行，\ 制表，\b 退格，\r 回车，\\斜杠，\’单引号，\”双引号。
3. 要把一个值转换为一个自字符串有两种方法：1.几乎每个值都有的toString()方法，唯一要做的就是返回相应的字符串表现。在不知道要转换的值是不是null或undefined的情况下，还可以使用转型函数String(),这个函数能够将任何类型的值转换为字符串。
4. valueOf():返回对象的字符串，数值或布尔值表示。
5. toLocaleString():返回对象的字符串表示，改字符串与执行环境的地区对应。
6. 乘积超过了ECMAScript数值的表示范围，则返回Infinity或-Infinity.
7. Infinity与0相乘，则结果是NaN.
8. Return语句之后的任何代码都永远不会执行。
9. 在函数体内可以通过arguments对象来访问这个参数数组，从而获取传递给函数的每一个参数。
10. Arguments对象只是与数组类似（它并不是Array的实例），因为可以使用方法括号语句访问他的每一个元素（即第一个元素arguments[0],第二个元素arguments[1]...）。
11. Arguments对象的length属性可以获知有多少个参数传递给了函数。
12. 没有传递值得命名参数将自动被赋予undefined值。（3.7.1）
13. 没有重载。Js中定义了两个名字相同的函数，则改名字只属于后定义的函数。
14. ECMAScript中的基本数据类型包括undefined,null,Boolean,Number和String.
15. ECMAScript中也有一种复杂的数据类型，即Object类型。
16. 对于引用类型的值，我们可以为其添加属性和方法，也可以改变和删除其属性和方法。（4.1.1）
17. 不能给基本类型的值添加类型，尽管这样不会导致任何错误。只能给引用类型值动态地添加属性。
18. 复制变量值，复制操作结束后，两个变量实际上将引用同一个对象，因此，改变其中一个变量，就会影响另一个变量。
19. 对象时按值传递的。
20. 即使在函数内部修改了参数的值，但原来的引用仍然保持未变。
21. Typeof操作符是确定一个变量是字符串，数值，布尔值，还是undefined的最佳工具。
22. 在想知道某个对象是什么类型的，instanceof操作符。
23. 所有全局变量和函数都是作为window对象的属性和方法创建的。某个执行环境中的所有代码执行完毕后，该环境被销毁，保存在其中的所有变量和函数定义也随之销毁。全局执行环境直到应用程序退出，例如关闭网页时才会被销毁。
24. 当代码在一个环境中执行时，会创建变量对象的一个作用域链，其用途是保证对执行环境有权访问的所有变量和函数的有序访问。（4.2）
25. 全局执行环境的变量对象始终都是作用域链中的最后一个对象。
26. Js中，if语句中的变量声明会将变量添加到当前的执行环境中。(4.4.2)
27. Js中没有块级作用域。
28. Var声明的变量会自动被添加到最接近的环境中。
29. 对象时某个特定应用类型的实例。新对象是使用new操作符后跟一个构造函数来创建的。Var person = new Object();
30. 创建object实例的方法有两种：
31. New后面跟构造函数

var person = new Object();

Person.name = “Alice”;

Person.age = 29;

1. 对象自面量

Var person = {

name : “Alice”,

age : 29

};属性名也可以用字符串”name” : “Alice”

1. 使用对象字面量语法时，如果留空其花括号，则可以定义只包含默认属性和方法的对象，var person = {};//与new Object()相同。
2. 访问对象属性时使用的都是点表示法。
3. 在使用方括号语法时应该将要访问的属性以字符串的形式放在方括号中，alert(person[“name”]);//”Alice”alert(person.name);//”Alice”
4. 方括号语法的主要优点是可以通过变量来访问属性，

Var properyName = “name”;alert(person[propertyName]);//”A

1. ECMAScript数组的每一项可以保存任何类型的数据。
2. 创建数组的基本方式有两种。(1)使用Array构造函数

Var colors = new Array();（5.2）

1. 如果预先知道数组要保存的项目数量，也可以给构造函数传递该数量，而该数量会自动变成length属性的值。

Var colors = new Array(20);

1. 也可以向Array构造函数传递数组中应该包含的项。

Var colors = new Array(“red”,”blue”,”green”);

1. 给构造函数传递一个值也可以创建数组，如果传递的是数值，则会按照该数值创建包含给定项数的数值；而如果传递的是其他类型的函数，则会创建包含那个值得只有一项的数组。Var colors = new Array(3);创建一个包含3箱的数组

Var colors names = new Array(“Alice”);创建一个包含1项，即字符串”Alice”的数组。

1. 在使用Array构造函数是也可以省略new操作符，

Var colors = Array(3);

1. 创建数组的第二种基本方式是使用数字字面量表示法，数组字面量由一堆包含数组项的方括号表示，多个数组项之间以都好隔开，

Var colors = [“red”,”blue”,”green”];//创建一个包含3个字符串的数组。Var names = [];//创建一个空数组

1. 在读取和设置数组的值时，要使用方括号提供相应值的基于0的数字索引。

Var colors = [“red”,”blue”,”green”];

Alert(colors[0]);

Colors[2] = “brown”;

1. 数组的项数保存在其length属性中，这个属性始终会返回0或者更大的值。

Var colors = [“red”,”blue”,”green”];var names = [];

Alert(colors.length);3

Alert(names.length);0

1. 数组的length属性：它不是只读，可以从数组的末尾移除项或向数组中添加新乡。

Var colors = [“red”,”blue”,”green”];var names = [];

Colors.length = 2;

Alert(color[2]);//undefined

Var colors = [“red”,”blue”,”green”];var names = [];

Colors.length = 4;

Alert(color[3]);//undefined

Var colors = [“red”,”blue”,”green”];

Colors[colors.length] = “black”;//在位置3添加一种颜色

46.数组最后一项的索引始终是length-1，因此下一个位置就是length。

47.Var colors = [“red”.”blue”,”green”];

Colors[99] = “black”;

Alert(colors.length);//100

1. 确定某个对象是不是数组，instanceof操作符。If(value instanceof Array){ //对数组执行某些操作}
2. 所有对象都具有toLocaleString(),toString(),valueOf()方法。

调用的toString()方法会返回由数组中每个值的字符串形式拼接而成的一个以逗号分隔的字符串。调用valueOf()返回的还是数组。

1. toLocaleString()方法经常也会返回余toString()和valueOf()方法相同的值，但也不总是如此。当调用数组的toLcaeString()方法是，也会创建一个数组值的以逗号分隔的字符串。而与前两个方法唯一不同之处在于，这一次为了取得每一项的值，调用的是每一项的toLcaleString()方法，而不是toString()方法。
2. 使用join()方法，则可以使用不同的分隔符；来构建这个字符串。Alert(colots.join(“||”));red||green||blue(5.2.2)
3. Push()方法可以接受任意数量的参数，把他们逐个添加到数组末尾，并返回修改后数组的长度。
4. Pop()方法则从数组末尾移除最后一项，减少数组的length值，然后返回移除项。（5.2.3）
5. 模拟队列只需一个从数组前端取得项的方法，shift()方法,t它能移除数组中的第一个项并返回该项，同时将数组长度减1.
6. Unshift()方法，它能在数组前端添加个项并返回新数组的长度。
7. 重排序：resverse()和sort();
8. Sort()方法按升序排列数组项--即最小的值位于最前面，最大的值位于最后面。
9. Sort()方法会调用每个数组项的toString()转型，然后比较得到的字符串，以确定如何排序。Sort()方法比较的也是字符串。
10. valueOf()方法会返回数值类型的对象，function compare(value1,value2){return value2-value1;}
11. Concat()方法可以基于当前数组中的所有项创建一个新数组。这个方法先创建当前数组一个副本，然后接收到的参数添加到这个副本的末尾，最后返回新构建的数组。
12. Slice()方法它能够基于当前数组中的一或多个项创建一个新数组。
13. Slice()方法可以接受一或两个参数，即要返回项的起始和结束位置。如果两个参数，该方法起始和结束位置之间的项--不包括结束位置的项。不会影响原始数组。
14. Splice()方法，用途向数组的中部插入项。
15. 删除：要删除的第一项的位置和要删除的项数，splice(0,2);会删除数组中的前两项。
16. 插入：向指定位置插入任意数量的项。起始位置，0（要删除的项数），和要插入的项。Splice(2,0,”red”,”blue”);从当前位置2插入字符串。
17. 替换：向指定位置插入任意数量的项。且同时删除任意数量的项。起始位置，要删除的项数，要插入的任意数量的项。Splice(2,1,”red”,”green”);
18. 两个位置方法：indexOf()和lastIndexOf().接受两个参数：要查找的项和（可选的）表示查找起点位置的索引。indexOf()方法从数组的开头（位置0）开始向后找，lasIndexOf()从末尾开始找。（5.2.7）
19. 这两个方法都返回要查找的项在数组中的位置，或者在没找到的情况下返回-1.
20. 迭代方法

传入这些方法的函数会接收三个参数：数组项的值，该项在数组中的位置和数组对象本身。

1. every()：对数组中的每一项运行给定函数，如果该函数对每一项都返回true，则返回true.
2. Filter()：对数组中的每一项运行给定函数，返回该函数会返回true的数组。
3. forEach():对数组中的每一项运行给定函数，这个方法没有返回值。
4. Map()：对数组中的每一项运行给定函数，返回每次函数调用的结果组成的数组。
5. Some()：对数组中的每一项运行给定函数，如果函数对任一项返回true,则返回true。
6. map()方法适合创建包含的项与另一个数组意义对应的数组。
7. forEach()他是对数组中的每一项运行传入的函数。
8. 归并数组的方法：reduce()和reduceRight().这两个方法都会迭代数组的所有项，然后构建一个最终返回的值。（5.2.9）

Reduce()方法从数组的第一项开始，组个遍历到最后。而reduceRight()则数组的最后一项开始，向前遍历到第一项。

这两个方法都接受两个参数：一个在每一项上调用的函数和（可选的）作为归并基础的初始值。

传给reduce()和reduceRight()的函数接受4个参数：前一个值，当前值，项的索引和数组对象。这个函数返回的任何值都会作为第一个参数自动传给下一项。第一次迭代发生在数组的第二项上，因此第一个参数是数组的第一项，第二个参数就是数组的第二项。

1. Date.parse()方法接受一个表示日期的字符串参数。
2. 一个正则表达式就是一个模式与g(表示全局模式，即模式将被应用于所有字符串，而非在发现第一个匹配项是立即停止），i（不区分大小写模式），m（多行模式，即在到达一行文本末尾是还会继续查找下一行中是否存在于模式匹配的项）。（5.4）
3. 与其他语言中的正则表达式类似，模式中使用的所有元字符都必须转义。正则表达式中的元字符包括：( [ { \ ^ $ | ) ? \* + . ] }
4. RegExp的每个实例都具有下列属性：
5. galobal:布尔值，表示是否设置了g标志。
6. ignoreCase:布尔值，表示是否设置了i标志。
7. lastIndex:整数，表示开始搜索下一个匹配项的字符位置，从0算起。
8. Multiline:布尔值，表示是否设置了m标志。
9. RegExp对象的主要的方法是exec()，该方法是专门为捕获组而设计的。（5.4.2）
10. 在同一个字符串上多次调用exec()将始终返回死一个匹配的信息。而在设置全局标志的情况下，每次调用exec()则都会在字符串中据需查找新的匹配项。全局模式lastIndex的值每次调用exec()后都会增加，而在非全局模式下则始终保持不变。
11. 正则表达式的第二个方法是text()，他接受字符串参数，在只想知道目标字符串与某个模式是否匹配，并不需要知道其文本内容的情况下，当使用这个方法非常方便。
12. 解析器会率先读取函数声明，宁使其在执行任何代码之前可用（可以访问），至于函数表达式，则必须等到解析器执行到它所在的代码行，才会真正被解释执行。
13. 默认情况下，sort()方法会调用每个对象的toString()方法以确定他们的次序。
14. 函数内部，有连个特殊的对象：arguments和this.
15. Arguments的主要用途是保存函数参数，这个对象还有一个名叫callee属性，该属性时一个指针，这项拥有这个arguments对象的函数。
16. 函数对象属性：caller。这个属性中保存着调用当前函数的函数的引用。
17. 不能为函数的caller属性赋值。
18. Length属性表示函数希望接受的命令参数的个数。
19. Prototype是保存它们所有实例方法的真正所在。Prototype是不可枚举的。
20. 每个函数都包含两个非继承而来的方法：apply()和cal()。这两个方法的用途都是在特定的作用域中调用函数，实际上等于设置函数体内this对象的值。
21. 在使用call()方法时，传递给函数的参数必须逐个列举出来。
22. Call()和apply()真正强大的地方是能扩充函数赖以运行的作用域。
23. Bind()，这个方法会创建一个函数的实例。其this值会被绑定到传给bind()函数的值。
24. 3个特殊的引用类型：Boolean,Number,String.（5.6）
25. 引用类型与基本包装类型的主要区别就是对象的生存期。
26. 自动创建的基本包装类型的对象，则只存在于一行代码的执行瞬间，然后立即被销毁。
27. 对基本包装类型的实例用typeof回返回“object”,而且所有基本包装类型的对象都会被转换为布尔值true。
28. Typeof确定一个变量是字符串，数值，布尔值，还是undefined，检测基本类型。
29. 布尔表达式中的所有对象都会被转换为true。
30. Instanceof 检测引用类型。
31. toFixed()方法会按照指定的小数位返回数值的字符串表示。
32. valueOf(),toLocaleString(),toString()方法，都返回对象所表示的基本字符串值。
33. 两个用于访问字符串中特定字符串是：charAt()和charCodeAt()。这两个方法都接受一个参数，即基于0的字符位置。

Var stringValue = “hello world”;

Alert(stringValue.charAt(1));//e

1. 如果想得到的是字符编码，使用charCodeAt().

Var stringValue = “hello world”;

Alert(stringValue.charCodeAt(1));//101

1. 使用方括号加数字索引来访问字符串中的特定字符。

Var stringValue = “hello world”;

Alert(stringValue[1]);//e

1. cancat()用于将一或多个字符串拼接起来，返回拼接得到的心字符串。
2. 三个基于字符串创新新字符串的方法：slice(),substr(),substring()，这三个方法都会返回被操作字符串的一个子字符串，而且也都接受一或两个参数。第一个参数指定子字符串的开始位置，第二个参数表示子字符串的位置。而substr()的第二个参数指定的则是返回的字符个数。
3. 在传递给这些方法的参数是负数的情况下，他们的行为就不尽相同了。Slice()方法会将传入的负值与字符串的长度相加，substr()方法将负的第一个参数加上字符串的长度，而将负的第二个参数转换为0.substring()方法会把所有负值参数都转换为0.
4. Substring()方法会将较小的数作为开始位置，将较大的数作为结束位置。Substr()方法没有这个功能。
5. 从字符串中查找子字符串的方法：indexOf()和lastIndexOf()。返回子字符串的位置，如果没有找到该子字符串，则返回-1.
6. indexOf()方法凶字符串的开头向后搜索子字符串，而lastindexOf()方法是从字符串的末尾向前搜索子字符串。
7. 这两个方法都可以接收两个参数，表示从哪个位置开始搜索，indexOf()从该参数指定的位置向后搜索，lastIndexOf()则会从指定的位置向前搜索。
8. Trim()方法,这个方法会创建一个字符串的副本，删除前置及后缀的所有空格，然后返回结果。
9. TrimLeft()和trimRight()方法，分别删除字符串开头和结尾的空格。
10. 字符串大小写转换方法：toLowerCase()和toUpperCase()是经典方法，toLocaleLowerCase()和toLocaleUpperCase()方法则是针对特定地区的实现。
11. String类型定义几个用于在字符串中匹配模式的方法。(1).match()方法只接受一个参数，要么是一个正则表达式，要么是一个RegExp对象。
12. 用于查找模式的方法是search()。这个方法的唯一参数与match()相同，search()方法始终是从字符串开头向后查找模式。
13. Replace()方法，接受两个参数，一个参数可以是一个RegExp对象或者一个字符串，第二个线数可以是一个字符串或者一个函数。如果第一个参数是字符串，那么只会替换第一个子字符串。，要想替换所有子字符串，唯一办法就是提供一个正则表达式，指定全局（g）。
14. Replace()方法的第二个参数也可以是一个函数，在只有一个匹配项情况下，会向这个函数传递3个参数：模式的匹配项，模式匹配项在字符串的位置和原始字符串。
15. Split()，这个方法基于指定的分隔符将一个字符串分隔成多个子字符串，并将结果放在一个数组中。
16. localeCompare()，这个方法比较两个字符串，并返回下列值中的一个：
17. .如果字符串在字母表中应该在字符串之前，则返回负数。
18. 如果等于字符串参数，则返回0；
19. 如果字符串在字母表中应该排在字符串参数值后，则返回一个正数；
20. fromCharCode()方法，接收以或多个字符编码，人后将它们转换一个字符串。

Alert(String.fromCharCode(104,101,108,108,111);//hello

1. eval()方法，就像一个完整的ECMAScript解析器，即要执行的ECMAScript。
2. Eval(“alert(‘hi’)”);相当于alert(‘hi’);
3. Math.random()方法返回大于等于0小于1的随机数。
4. 值 = Math.floor(Math.random() \*可能值的总数 + 第一个可能的值)
5. 多数情况下，可以通过一个函数来计算可能值的总数和第一个可能的值。selectFrom()接受两个参数，一个最小值，一个最大值。最大值减最小值加1，就是总数。
6. 定义函数有两种方法：一种是函数声明，另一种是函数表达式。
7. 函数声明语法：function functionName(arg0,arg1,arg2){..}
8. 函数表达式：

var functionName = function(arg0,arg1,arg2){...函数体...};

1. 闭包是指有权访问另一个函数作用域中的变量的函数。创建闭包的常见方式，就是在一个函数内部创建另一个函数。
2. 闭包只能取得包含函数中任何变量的最后一个值。闭包所保存的是整个变量对象，而不是某个特殊的变量。（7.2.1）
3. This对象时在运行时基于函数的执行环境绑定的；在全局函数中，this等于window，而当函数被作为某个对象的方法调用时，this等于那个对象。
4. 文档节点是每个文档的根节点。**（10章）**

130.在HTML页面中，文档元素永远是<html>元素。

131.Node类型，每个节点都有一个nodeType属性，用于表明节点的类型。

132.节点类型有在Node类型中定义的下列12个数值常量来表示：任何节点类型必居其一：

(1).Node.ELEMENT\_NODE(1);

(2).Node.ATTRIBUTE\_NODE(2);

(3).Node.TEXT\_NODE(3);

(4).Node.CDATA\_SECTION\_NODE(4);

(5).Node.ENTITY\_REFERENCE\_NODE(5);

(6).Node.ENTITY\_NODE(6);

(7).Node.PROCESSING\_INSTRUCTION\_NODE(7);

(8).Node.COMMENT\_NODE(8);

(9).Node.DOCUMENT\_NODE(9);

(10).Node.DOCUMENT\_TYPE\_NODE(10);

(11).Node.DOCUMENT\_FRAGMENT\_NODE(11);

(12).Node.NOTATION\_NODE(12);

133.通过比较上面这些常量，可以很容易地确定节点的类型，例如：if (someNode.nodeType == 1){

Alert(“Node is an element.”);

}