1.1.js的特点

语法简单，易学易用；

解释性语言：非脚本语言编写的程序通常需要进过‘编写-编译-链接-运行’4个步骤。脚本语言js只需要经过‘编写-运行’2个步骤。

跨平台：只依赖浏览器

基于对象和事件驱动：js把html页面中的每个元素都当作一个对象来处理-对象组成文档对象模型。基于事件驱动-js捕捉到用户在浏览器中的操作。

用于客户端：js分为客户端和服务端两种。

1.2.js与java的区别

基于对象和面向对象：js基于对象，本身提供了非常丰富的内部对象。java面向对象，即使是开发简单的程序，也必须设计对象。

强变量和弱变量：js弱变量，即变量使用前不需要声明，而是有解释器在运行时检查其数据类型。java强类型变量检查，即所有变量在编译前必须声明。

代码方式不同

嵌入方式不同

静态联编和动态联编：js静态联编，即js的对象在运行时进行检查。java动态联编，对象引用必须在编译时进行，以使编译器必须能够实现强类型检查。

浏览器执行方式不同

2.js基础

2.1.js的基础语法

执行顺序：

区分大小写：js严格区分大小写(HTML不区分大小写，许多js对象和属性都与其代表的html标签或属性同名。如，在html中单击事件处理器属性通常被声明为onClick或Onclick，而在js中只能使用onclick)

分号与空格：js分号可有可无，并且会忽略多余的空格

对代码行进行折行：用户可以在文本字符串中使用反斜杠(\)对代码进行换行，不是文本字符串不行。

注释： // /\* \*/

(老浏览器不识别js语言，则需再js外加上<!-- -->)

语句和语句块

2.2.js的数据结构

js 脚本语言的数据结构包括标识符、常量、变量、关键字等。

标识符：只能有字母、数字、下划线、美元符号和中文组成，而不能包含空格、标点符号、运算符等其他符号；不能以数字开头；不能使用关键字、保留字。

关键字、保留字：不能用作变量名和函数名。

常量：const声明，常量地址不可改变。

变量：

变量命名：第一个字符必须是一个ASCII字符或一个下划线，但不能是文字(???经验证，文字也可以)

变量声明与赋值：js是一种弱类型的程序设计语言，变量可以不声明直接使用。(js变量如果未初始化/赋值，默认值为undefined)

变量的作用范围：全局变量和局部变量(没有var声明的变量为全局变量)

2.3.看透代码中的数据类型

typeof运算符、undefined类型、null类型、boolean类型(表示逻辑数值)、number类型、string类型、object类型

(null表示一个变量被赋予了空值，undefined表示该变量还未被赋值)

(js的数值类型可以分为4类，即整数、浮点数、内部常量和特殊值。浮点数可以包含一个小数点，也可以包含一个‘e’科学技术法。整数分为整数、0和负数，可以以十进制、八进制和十六进制作为基数来表示。)

(前边介绍的是js的五种原始数据类型，Object是对象类型。对象类型包括Object、Function、String、Number、Boolean、Array、Regexp、Date、Global、Math、Error以及宿主环境提供的object类型)

数值转换：Number(强制转换，可用于任何数据类型???除了完全由数值组成的字符串外的转换完都是NaN)、parseInt和ParseFloat(可以将数值开头的部分截取完成转换)、toString(不能转换null和undefined会报错)和String()、Boolean

2.4运算符

算数运算符：二元五种+ - \* / % ，医院运算符两种 ++和--(结合性自右向左??i++中i的执行顺序再首页轮播时的问题)。

比较运算符：返回布尔类型

位运算符：对数据按二进制位进行运算的运算符。&与、|或、^异或、~取补、<<左移、>>右移。（其中取补运算符为一元运算符，其他均为二元运算符。这些运算均不会溢出，位运算符的操作数为整型或者是转换为整型的任何其他类型。）（异或：两个操作位相同时结果为0，不相同时结果为1。 左移：操作数按位左移，高位被丢弃，低位顺序补0。 右移：操作数按位右移，低位被丢弃，其他各位顺序依次右移。）

逻辑运算符：&&、||、!

条件运算符

赋值运算符：=、+=、-=、\*=、/=、%=、&=(a&b相当于a=a&b)、|=、^=

运算符的优先级：只有自增、自减、赋值运算符和快捷运算符是从右到左，其他都是从左到右。

2.5表达式

赋值表达式、算术表达式、布尔表达式、字符串表达式

类型转换

2.6局部变量和全局变量

3.循环控制结构和语句

if要是用小写的，如果使用大写则会报错

break语句用于退出包含在最内层的循环或者退出一个switch语句。break语句通常用在for、while、do-while或switch语句中。

continue终止本次循环，并进行下一次循环。只能用在while、for、do-while和switch语句中。

4函数

4.2定义函数

4.2.3函数参数的使用

在未调用函数时，形参并不占用存储单元。只有在发生方法调用时，才会给函数中的形参分配内存单元。在调用结束后，形参所占的内存单元也自动释放。

实参可以是变量、常量或表达式；形参必须是声明的变量，由于js是弱类型语言，所有不需要指定类型。

在函数调用中，实参列表中的参数的数量、类型、和顺序必须与参数列表中的参数一致。如果形参个数大于实参个数，那么多出的行参值为undefined，反之，多出的实参将忽略。

实参对形参的数据传递是单向传递，即只能由实参传给形参，而不能由形参传给实参。

4.2.4函数的返回值

使用return返回，不使用则默认返回undefined

4.3函数的调用

直接调用；在表达式中调用；在事件响应中调用；通过链接调用

4.4 js中常用的函数

4.4.1嵌套函数：

可以使用内部函数轻松获得外部函数的参数以及函数的全局变量。(功能枪法，可读性低)

4.4.2递归函数：

递归函数处理不当，会使程序进入死循环。为了防止死循环，可以设计一个做自加运算的变量，用于记录函数自身调用的次数，如果次数太多，就使它自动退出。定义递归的连个必要条件，一个结束递归的条件，一个递归调用语句。

4.4.3 内置函数：

可提高编程效率，常用的内置函数有6种。

eval函数： eval可以把一个字符串当成一个表达式去执行。接受一个字符串型的参数，将这个字符创作为代码在上下文环境中执行，并返回执行的结果。(有返回值，如果参数字符串是一个表达式，则返回表达式的值，没有值则返回undefined；参数字符串作为代码执行时，是与调用eval函数的上下文相关的，即其中出现的变量或函数调用，必须在调用eval的上下文环境中可用)

isFinite(number)函数：确定参数是否是一个有限值。（参数为非数值、正无穷数或负无穷数，返回false，否则返回true；如果是字符串类型的数值，则会自动转为数值型）

isNaN(num)

parseInt和parseFloat：parseFloat(str)函数返回由字符串转换得到的浮点数，其中，字符串参数是包含浮点数的字符串；即如果str值为11，那么计算结果就是11，而不是3或B。

Number和String

escape和unescape函数：escape(str)对String对象编码，以便它们能在所有计算机上可读。str为必选项，表示要编码的String对象或文字，返回一个包含了str内容的字符串值(Unicode格式)。除了个别如\*@之类的符号外，其余所有空格、标点、重音符号以及其他非ASCII字符均可用%xx编码代替，其中xx等于表示该字符的十六进制数；unescape用于返回指定值的ASCII字符串。

疑难解惑：1.如果浏览器不支持js，如何做能够不影响网页的美观呢？可以使用html注释语句<!-- -->将其注释，这样便不会在网页中输出这些代码了。2.函数Number和parseInt都可以将字符串转成整数，有何区别？函数Number不但可以将数字字符串转换成整数，还可以转换成浮点数，它是将字符串转换成数值。而parseInt只能转成整数；Number在转换时，如果字符串中包含非数字字符，则转换回失败，而parseInt函数只要第一个是数字即可成功。

5.对象与数组

对象：复合数据类型

数组时js中唯一用来存储和操作有序数据集的数据结构。

5.1对象

对象组成：一组各种类型数据的属性；允许对属性中的数据进行的操作，即相关方法。属性、方法、事件

js支持的对象分为4种：

js核心对象：基本数据类型的相关对象(String、Boolean、Number)、允许创建用户自定义和组合类型的对象(如Object、Array)、其他能简化js操作的对象(Math、Date、RegExp、Function)

浏览器对象：包括不属于js语言本身但被绝大数浏览器支持的对象，如控制浏览器窗口和用户交互界面的window对象、提供客户端浏览器配置信息的Navigator对象。

用户自定义对象：web应用程序开发者用于完成特定任务而创建的自定义对象，可自由设计对象的属性、方法和事件处理程序，编程灵活性较大。

文本对象：由文本域构成的对象，在DOM中定义，同时赋予很多特定的处理方法，如insertData()、appendData()等

面向对象编程：js脚本是基于对象的脚本编程语言，而不是面向对象的编程语言。其原因在于：js是以DOM和BOM中定义的对象模型及操作方法为基础的，但又不具备面向对象编程语言所具备的显著特征，如分类、继承、封装、多态、重载等。js还支持DOM和BOM提供的对象模型，用于根据其对象模型层次结构来访问目标对象的属性并对对象施加相应的操作。(在js中，之所以任何类型的对象都可以赋予任意类型的数值，是因为js为弱类型的脚本语言，即变量在使用前不许声明，在浏览器解释运行其代码时，才检查目标变量的数据类型)

js内部对象:静态对象和动态对象。(动态对象使用new创建实例：对象实例名.成员；静态对象:对象名.成员)？？？

静态对象：Event(用来表示js的事件)、File(获取服务器端指定文件的相关属性)、Folder(获取服务器端指定文件的相关属性)。

动态对象：Object(可以在程序运行时为js对象随意添加属性)、String(处理或格式化文本字符串以及确定和定位字符串中的子字符串)、Date、FileSystemObject(实现文件操作功能)、Drive(用于收集系统中的物理或逻辑驱动器资源中的内容)。

5.2 对象访问语句

for-in：遍历对象的每一个属性(包括继承的属性)。

with语句：(在with语句中，凡是js不识别的属性和方法都和该语句块指定的对象有关。即参数为对象，块中成员为该对象的成员，不需要使用对象名.成员/对象实例.成员的方式访问)

5.3 js中的数组

有序的，数组元素允许属性为不同数据类型。参数可为空、数组长度、数组元素逗号隔开(一个参数为一个元素)

问题：创建固定长度的数组，还可再增加多余的元素？(使用固定长度创建的数组，还可以再push新元素，如果创建时没指定元素只声明了长度，则声明长度部分的元素不能是用push，只能使用arr[i]赋值)

5.5创建和使用自定义对象

通过定义对象的构造函数的方法：先定义对象构造函数，然后使用new操作符生成该对象的实例。

通过对象直接初始化的方法：使用{}直接定义对象

(创建对象的三种方式：对象字面量；new操作符；Object.create()方法)

js删除对象实例的属性之后，该属性变为undefined，同任何不存在的对象属性一样为未定义类型，但并不能删除对象实例本身，否则将返回错误。(js脚本中的delete运算符用于删除对象实例的属性，而在C++中，delete运算符不能删除对象的实例)

通过原型为对象添加新属性和新方法：js中，对象prototype属性是用来返回对象类型原型引用的。使用prototype属性提供对象的类的一组基本功能。并且对象的新实例会‘继承’赋予该对象原型的操作。所有js对象都有只读的prototype属性。可以向其原型中动态添加功能(属性和方法)，但该对象不能被赋予不同的原型。然而，用户定义的对象可以被赋予新的原型。

内存的分配和释放：js是基于对象的编程语言，而不是面向对象的编程语言，因此缺少指针的概念。面向对象的编程语言在动态分配和释放内存的方面起着非常重要的作用，那么js中的内存如何管理呢？在创建对象的同时，浏览器自动为创建的对象分配内存空间，js将新对象的引用传递给调用的构造函数；而在对象清除时其占据的内存将被自动回收，其实整个过程都是浏览器的功劳，js只是创建该对象。 浏览器中的这种内存管理机制称为‘内存回收’，它动态分析程序中每个占据内存空间的数据(变量、对象等)。如果该数据对于程序标记为不可再用时，浏览器将调用内部函数将其占据的内存空间释放，实现内存的动态管理。在自定义的对象使用过后，可以通过给其赋空值null的方法来标记对象占据的空间可以释放。浏览器将根据此标记动态释放其占据的内存，否则将保存该对象，直至当前程序再次使用它为止。

6.日期与字符串对象

6.1创建日期对象：

创建日期的四种方法：new Date()、new Date(日期字符串)、new Date(年,月,日,时,分,秒,[毫秒]) 、new Date(毫秒)

new Date()：创建了一个包含当前系统事件的日期对象。

new Date(日期字符串)：可以将一个字符串转成日期对象，这个字符串可以是只包含日期的字符串，也可以是既包含日期又包含时间的字符串。

(js使用的日期格式通常有以下两种：日期字符串表示为‘月 日,年 小时:分钟:秒钟’其中月份必须使用英文单词，而其他部分可以使用数字表示，日和年之间一定要有逗号；日期字符串可以表示为‘年/月/日 小时:分钟:秒钟’左右部分都要求使用数字，年份要求使用4位，月份用1-12整数)

new Date(年,月,日,时,分,秒,[毫秒])：指定年月日时分秒创建日期对象，时分秒都可以省略。月份用0-11的整数。

new Date(毫秒)：使用毫秒来创建日期对象。将1970年1月1日0时0分0秒0毫秒为基数，参数代表距离这个基数的毫秒数。

Date对象属性：Date日期对象只包含两个属性，分别是constructor和prototype，这两个属性在每个内部对象中都有。

日期对象的常用方法：setXX方法用于设置时间和日期值；getXX方法用于获取时间和日期值；toXX方法主要是将日期转换成指定格式。

|  |  |  |
| --- | --- | --- |
| 1 | Date() | 返回当日的日期和时间 |
| 2 | getDate() | 从Date对象返回一个月中的某一天(1-31) |
| 3 | getDay() | 从Date对象返回一周中的某一天(0-6) |
| 4 | getMonth() | 从Date对象返回月份(0-11) |
| 5 | getFullYear() | 从Date对象以4位数字返回年份 |
| 6 | getYear() | 请使用getFullYear()方法代替 |
| 7 | getHours() | 返回Date对象小时(0-23) |
| 8 | getMinutes() | 分钟(0-59) |
| 9 | getSeconds() | 秒数(0-59) |
| 10 | getMilliseconds() | 毫秒(0-999) |
| 11 | getTime() | 返回1970年1月1日至今的毫秒数 |
| 12 | getTimezoneOffset() | 返回本地时间与格林尼治标准时间(GMT)的分钟差 |
| 13 | getUTCDate() | 根据世界时从Date对象返回月中的一天(1-31) |
| 14 | getUTCDay() | 周中的一天(0-6) |
| 15 | getUTCMonth() | 返回月份(0-11) |
| 16 | getUTCFullYear() | 4位数年份 |
| 17 | getUTCHours() | 小时 |
| 18 | getUTCMinutes() | 分钟 |
| 19 | getUTCSeconds() | 秒钟 |
| 20 | getUTCMilliseconds() | 毫秒 |
| 21 | parse() | 返回1970年1月1日午夜到指定日期(字符串)的毫秒数 |
| 22 | setDate() | 设置Date对象中月份的某一天(1-31) |
| 23 | setMonth() | (0-11) |
| 24 | setFullYear() |  |
| 25 | setYear() | 请使用setFullYear()代替 |
| 26 | setHours() |  |
| 27 | setMinutes() |  |
| 28 | setSeconds() |  |
| 29 | setMilliseconds() |  |
| 30 | setTime() | 以毫秒设置Date对象 |
| 31 | setUTCDate() | 根据世界时设置Date对象中月份的一天(1-31) |
| 32 | setUTCMonth() | (0-11) |
| 33 | setUTCFullYear() |  |
| 34 | setUTCHours() |  |
| 35 | setUTCMinutes() |  |
| 36 | setUTCSeconds() |  |
| 37 | setUTCMilliseconds() |  |
| 38 | toSource() | 返回该对象的源代码 |
| 39 | toString() | 把Date对象转换为字符串 |
| 40 | toTimeString() | 把Date对象的时间部分转换为字符串 |
| 41 | toDateString() | 把Date对象的日期部分转换为字符串 |
| 42 | toGMTString() | 请使用toUTCString()代替 |
| 43 | toUTCString() | 根据世界时，把Date对象转换为字符串 |
| 44 | toLocaleString() | 根据本地时间格式，把Date对象转换为字符串 |
| 45 | toLocaleTimeString() | 根据本地时间格式，把Date对象的时间部分转换为字符串 |
| 46 | toLocaleDateString() | 根据本地时间格式，把Date对象的日期部分转换为字符串 |
| 47 | UTC() | 根据世界时返回1970年1月1日到指定日期的毫秒数 |
| 48 | valueOf() | 返回Date对象的原始值 |

从js1.6开始，添加了一个toLocaleFormat()方法，可以有选择地将日期对象中的某个或某些部分转换成字符串，也可以指定转换的字符串格式。

日期间的运算：

日期对象与整数年、月、或日相加：需将它们相加的结果，通过setXX函数设置新的日期对象。

日期相减：js中允许两个日期相减，相减之后将会返回这两个日期之间的毫秒数。通常会转换成秒、分、时、天等。

6.3字符串对象

js中的基本数据类型之一。在js中，可以将字符串直接看成字符串对象，不需要任何转换。

创建字符串对象的两种方法：直接声明字符串变量；使用new创建。

字符串对象的常用属性：constructor、length、prototype。(测试字符串长度时，空格占一个字符位。一个汉字占一个字符位，即一个汉字的长度为1。)

字符串对象的常用方法：

anchor()：创建html锚。

big()：用大号字体显示字符串。

blink()：显示闪动字符串。

bold()：使用粗体显示字符串。

charAt()

charCodeAt()

concat()：stringObject.concat()将所有参数转换成字符串，并按顺序连接。（stringObject本身没有改变，与Array.concat()相似。不过使用‘+’连接会更简便一些）

fixed()：以打字机文本显示字符串。

fontcolor()：使用指定的颜色来显示字符串。

fontsize()：使用指定的尺寸来显示字符串。

fontCharCode()：从字符编码创建一个字符串。

indexOf()

italics()：使用斜体显示字符串。

lastIndexOf()

link()：将字符串显示为链接

localeCompare()：用本地特定的顺序来比较两个字符串。(stringObject.localeCompare(target),其中target参数是要以本地特定的顺序与stringObject进行比较的字符串，返回值是比较结果数字。如果string小于target，则返回小于0 的数。如果string大于target，则返回大于0 的数。相等，则返回等于0 的数。)

match()：找到一个或多个正则表达式的匹配

replace()

search()：检索与正则表达式相匹配的值

slice()

small()：使用小字号来显示字符串

split()

strike()：使用删除线来显示字符串

sub()：把字符串显示为下标

substr()

substring()

sup()：显示为上标

toLocaleLowerCase()

toLocaleUpperCase()

(toLocaleLowerCase这两种方法按照本地方式把字符串转换为大小写。只有几种语言具有地方特有的大小写映射)

toLowerCase()

toUpperCase()

toSource()：代表对象的源代码

toString()

valueOf()：返回某个字符串对象的原始值

问题：

1.在实际开发中，会经常使用指定范围内的随机整数。借助数学方法，总结出以下两种指定范围内的随机整数产生方法：

产生0-n之间的随机数：Math.floor(Math.random()\*(n+1)) (包含0也包含n)

产生n1-n2之间的随机数：Math.floor(Math.random()\*(n2-n1))+n1 (包含n1不包含n2)

7.数值与数学对象

7.1Number对象

Number对象是原始数值的包装对象，它代表数值数据类型和提供数值的对象。

创建Number对象：可以不与new运算符一起使用，而直接作为转换函数来使用。以这种凡是调用Number对象时，它会把自己的参数转化成一个数值，然后返回转换后的原始数值。(numObj = new Number(value))

Number对象包括7个属性：

constructor：返回对创建此对象的Number函数的引用。

MAX\_VALUE：可表示的最大的数。

MIN\_VALUE：可表示的最小的数。

NaN：非数字值。

NEGATIVE\_INFINITY：负无穷大，溢出时返回该值。

POSITIVE\_INFINITY：正无穷大，溢出时返回该值。

prototype：使我们有能力向对象添加属性和方法。

7.2Number对象的方法

toString(radix)：把数值转换为字符串，使用指定的基数。(参数radix为可选项。规定表示数值的基数，为2-36之间的整数。若省略该参数，则使用基数10。但是要注意，如果该参数是10以外的其他值，则ECMAScript标准允许实现返回任意值。)

toLocaleString()：把数值转换为字符串，使用本地数字格式顺序。(根据本地规范进行格式化，可能影响到小数点或千分位分隔符采用的标点符号。)

toFixed(num)：把数值转换为字符串，结果的小数点后有指定位数的数字。(四舍五入为指定小数位数的数值。num必选项。规定小数的位数，是0-20之间的值，包括0和20，有些实现可以支持更大的数值范围。如果省略了该参数，将用0代替。)

toExponential(num)：把对象的值转换为指数计数法。(num必选项，规定指数计数法中的小数位数，在0-20之间。)

toPrecision(num)：把数值格式化为指定的长度。(可在对象的值超出指定位数时将其转换为指数计数法。num必选项，规定必须被转换为指数计数法的最小位数，该参数是1-21之前，包含1和21。有效实现允许有选择地支持更大或更小的num。如果省略了该参数，则调用方法toString()，而不是把数转换成十进制的值。)

valueOf()：返回一个Number对象的基本数值。

7.3 Math对象

问题：

1.Math对象与Date和String对象有哪几点不同？

Math对象并不像Date和String那样是对象的类，因此没有构造函数Math()，像Math.sin()这样的函数只是函数，不是某个对象的方法。无需创建它，通过把Math作为对象使用就可以调用其所有属性和方法。

Math不存储数据，String和Date对象存储数据。

2.如何表示对象的源代码？

使用toSource()方法可以表示对象的源代码。该方法通常由JavaScript在后台自动调用，并不显示地出现在代码中。(目前，只有Firefox支持该方法。)

8.文档对象模型与事件驱动

8.2访问节点

表：DOM定义的html文档节点类型

|  |  |  |  |
| --- | --- | --- | --- |
| 节点类型数值 | 节点类型 | 附加说明 | 示例 |
| 1 | 元素(Element) | Html标记元素 | <h1>...</h1> |
| 2 | 属性(Attribute) | Html标记元素的属性 | Color:'red' |
| 3 | 文本(text) | 被html标记括起来的文本段 | Hello world |
| 8 | 注释(Comment) | Html注释段 | <!-- comment --> |
| 9 | 文档(Doctype) | Html文档和文本对象 | <html> |
| 10 | 文档类型(DoctypeType) | 文档类型 | <!DOCTYPE HTML PUBLIC "..."> |

IE内核的浏览器中，属性类型在IE6版本中才获得支持。所有支持W3C DOM的浏览器(IE5+、Mozl和Safari等)都实现了前3种常见的类型，其中Mozl实现了所有的类型。

表：DOM中的节点处理方法

1.document.createElement(tagName):创建有tagName指定类型的标记。

2.document.createTextNode(string)：创建包含字符串string的文本节点。

3.document.createAttribute(name)：针对节点创建由name指定的属性，不常用

(返回Attr 对象属性 eg: var att=document.createAttribute("class");att.value="democlass";

document.getElementsByTagName("H1")[0].setAttributeNode(att);)。

4.document.createComment(string)：创建由字符串string指定的文本注释。

5.appendChild(newChild,targetChild)：添加子节点到目标节点上。

6.parentNode.insertBefore(newChild,targetChild)：将新节点newChild插入目标节点targetChild前。

7.cloneNode(bool)：复制节点自身，由逻辑量bool确定是否复制子节点。

8.parentNode.removeChild(childName)：删除由childName(获取的节点node)指定的节点。

9.parentNode.replaceChild(newChild,oldChild)：用新节点newChild替换旧节点oldChild。

(提示：由于Opera和Mozilla系列的浏览器的DOM树种包含空白字符，所以，上面的程序在这些浏览器中显示时，要删除一个节点，会比在IE浏览器中多单机几次删除按钮)

表：Node对象的方法

1.hasChildNodes()：判断节点是否包含子节点。

2.setAttrbuteNode(attibute)：设置节点的属性节点。

3.parentNode.firstChild/lastChild：获取节点的第一个或最后一个子节点

4.appendChild(newChild)：给当前节点添加一个子节点(newChild添加在targetNode节点的末尾添加新的节点。如果文档树中已经存在了newChild，它将从文档树种删除，然后插入新位置)。

5.cloneChild(deep)：复制当前子节点，当deep为true时，复制当前节点和其所有的子节点，否则仅复制当前节点本身。

6.insertBefore(newNode,refNode)：把一个newChild节点插入到redNode节点之前。

7.removeChild(Child)：删除指定的子节点。

8.replaceChild(newChild,oldChild)：用newChild子节点代替oldChild子节点。

9.selectNodes(patten)：获得符合指定类型的所有节点。

10.selectSingleNodes(patten)：获取符合指定类型的第一个节点。

11.TransformNode(styleSheetOBJ)：利用指定的样式来变换当前节点及其子节点。

(firstChild和lastChild指向当前标记的子节点集合内的第一个和最后一个子节点，但是在多数情况下使用childNodes集合，用循环遍历子节点。如果没有子节点，则childNodes的长度为0。)

表：处理文本节点的方法

1.appendData(string)：在文本节点的结尾添加一个由string指定的字符串。

2.deleteData(offset,count)：删除从offset处开始由count指定的字符串个数。

3.insertData(offset,string)：在offset处插入string指定的文本。

4.replaceData(offset,offset,string)：用string指定的文本替换自offset开始的由count指定数目的字符。

5.splitText(offset)：从offset处将原文本节点一分为二，其中左半部分作为原节点内容，而右半部分作为新节点内容。

6.substringData(offset,count)：返回一个字符串，该字符串包含自offset开始的由count指定的数目的一组字符。

8.4事件处理

鼠标事件：onclick、ondbclick、onmouseover、onmouseout、onmousedown、onmouseup、onselect

(onmousedown和onmouseup事件有先后顺序，在同一个对象上前者在先后者在后，共同使用，同事控制同一对象的状态改变)

(onSelect具体过程是从鼠标按键被按下，到鼠标开始移动并选中内容的过程。这个过程并不包括鼠标键的放开。)

键盘事件：onkeydown、onkeypress、onkeyup。

html事件：html文件状态改变时触发的、用户可以捕获的事件。

变动事件：onblur、onfocus、onchange

(onchange事件只在事件对象的值发生改变并且事件对象失去焦点时触发。)

onLoad：当文档载入时，产生该事件。onLoad的一个作用就是在首次载入一个文档是检测cookie的值，并用一个变量为其赋值，使它可以被源代码使用。？？

onUnload：当web页面退出时引发该事件，并可更新cookie的状态。？？

8.4.4使用event对象

使用window.event或直接使用event访问。

主要属性和方法：

type：事件的类型，去掉on前缀

srcElement：事件源，就是发生事件的元素。比如<a>链接

button：声明了被按下的鼠标键，是一个整数。0代表没有按键，1代表鼠标左键，2代表鼠标右键，4代表鼠标中间键。同时按下了多个键，则相加。

clientX/clientY：鼠标按下时的横纵坐标

offestX/offsetY：鼠标指针相对于源元素的位置，可确定单机Image对象的哪个元素。

altKey、ctrlKey、shiftKey：指鼠标事件发生时，是否同时按住了alt、ctrl、shift键，返回一个布尔值。

keyCode：返回按键的代码以及keypress事件的Unicode字符。

fromElement、toElement：前者是指代mouseover事件移动过的文档元素，后者指代mouseout事件中鼠标移动到的文档元素。

cancelBubble：布尔值，true时阻止事件冒泡。

returnValue：布尔值，false时可以阻止浏览器执行默认的事件动作。

attachEvent()和detachEvent()方法：为指定的dom对象事件类型注册多个事件处理函数的方法，它们有两个参数，第一个是事件类型，第二个是事件处理函数。在attachEvent事件执行的时候，this关键字指向的是window对象，而不是发生事件的那个元素。

9.处理窗口和文档对象

9.1窗口对象

表1：windows对象的属性

表2：windows对象的常用方法

9.2文档对象

表1：document对象的属性

表2：document对象的方法

10.级联样式表

10.3在html中使用css

行内样式、内嵌样式、链接样式、导入样式

10.3.4导入样式

导入样式和链接样式基本相同，都是创建一个单独的css文件，然后再引入到html文件中。只不过语法和运作方式有差别。采用导入样式的样式表，在html文件初始化时，会被导入到html文件内，作为文件的一部分，类似于内嵌效果。而链接样式是在html标记需要的样式风格时才以链接方式引入。(html什么时候需要样式风格？？)

导入外部样式表相当于将样式表导入到内部样式表中，其方式更有优势。导入外部样式表必须在样式表的开始部分，其他样式表上面。

优先级：行内样式最高，链接样式看在哪引入

( CSS3语言对于所有属性和值都有相对严格的要求，如果声明的属性在CSS3规范中没有，或者某个属性值不符合属性要求，都不能使CSS语句生效。)

10.4 CSS选择器

(设置全局选择器样式，不能只设置html、body选择器样式，有些属性是不能继承的？)

继承选择器：即后代元素选择器

问题：

1.CSS定义的字体在不同浏览器中大小会不一样吗？

会的。例如，使用font-size:14px定义的宋体文字，在IE下实际高是16px，下空白是3px，Firefox浏览器下实际高是17px、上空1px、下空3px。其解决方法时在文字定义时设定line-height，并确保所有文字都有默认的line-height值。

2.CSS在网页制作中一般有多种用法，具体使用时应该采用哪种用法？

当有多个网页要用到CSS时，采用外链CSS文件的方式，这样网页的代码可以大大减少，修改起来非常方便；只对单个网页中使用的CSS采用文档头部方式；网页中只有一两个地方才用到CSS时，才采用行内插入方式。

3.CSS的行内样式、内嵌样式和链接样式可以在一个网页中混用吗？

可以。浏览器先检查有没有行内样式，再检查内嵌样式、链接样式。

(实践链接写在内嵌样式之后，看哪个优先级高？)

11.表单和表单元素

11.1表单概述

表单主要用于收集网页上浏览者的相关信息。其标签为<form></form>。表单的基本语法格式如下：

<form action="url" method="get|post" enctype="mime">

</form>

antion="url"指定处理提交表单的格式，可以是一个url地址或一个电子邮件地址。

method提交表单的http方法。

enctype="mime"指明用来把表单提交给服务器时的互联网媒体形式。

11.2表单元素

11.3表单高级元素的使用

11.3.1 url元素

适用于说明网站网址的。显示为输入URL地址的一个文本框。在提交表单时，会自动验证url。<input type="url" name="">

用户可以使用普通属性设置url输入框，例如可以使用max属性设置其最大值、min属性设置其最小值、step属性设置合法的数字间隔、利用value属性规定其默认值。对于另外的高级属性中同样的设置不再重复讲述。

11.3.2 email属性

提交表单时，会自动验证email域的值。

11.3.3 date和times

在html中，新增了一些日期和时间输入类型，包括date(选取日、月、年)、datetime(选取时间、日、月、年)、datetime-local(选取时间、日、月、年-本地时间)、month(选取月、年)、week(选取周、年)和time(选取时间)。

11.3.4 number属性

提供了一个输入数值的输入类型。用户可以直接输入数值，或者通过单击微调框中的向上或向下按钮选择数值。

11.3.5 range属性

显示一个滑条控件。与number属性一样，用户可以使用max、min和step属性控制控件的范围，分别表示最大值和最小值。

默认情况下，滑块位于滑条的中间位置。如果用户指定的最大值小于最小值，则允许使用反向滚动轴，目前浏览器对这一属性还不能很好地支持。

11.3.6 required属性

规定必须在提交之前填写输入域(不能为空)，适用于text、search、url、email、password、date、pickers、number、checkbox和radio等输入类型。

11.4 表单对象在js中的应用

一个表单有表单标签(包含了处理表单数据所用CGI程序的URL以及数据提交到服务器的方法)、表单域(包含了文本框等)、表单按钮(包含提交按钮、复位按钮和一般按钮；用于将数据传送到服务器上的CGI脚本或取消输入，还可以用表单按钮来控制其他定义了处理脚本的处理工作)等3个基本组成部分。

11.4.1 html表单基础

1.定义表单

<form>对象的属性：

name表单的名称，可以在不命名的情况下使用表单，但是为了方便在js中使用表单，需要为其指定一个名称。

method可选属性。默认为post方法。(使用get方法提交表单要注意，url长度应限制在8192个字符内。如果数据量太大，数据将被截断。)

action指定表单处理程序的url地址。其内容可以是某个处理程序或页面(还可以使用#代替action的值，指明当前form的action就是其本身)，但是需要注意的是action属性的值必须包含具体的网络路径。例如，指定当前action为check文件夹下的userCheck.html。用户也可使用js脚本指定action的值，如document.loginForm.action="/check/userCheck.html"。

target指定当前form提交后目标文档的显示方法。可选属性，默认为\_self。

enctype指定form表单提交时的编码方式。默认的编码方式是application/x-www-form-urlencoded。如果需要在提交form时上传文件，entype值必须是multipart/form-data。(text/plain:纯文本编码。application/x-www-form-urlencoded：url编码。multipart/form-data：mime编码。)

element[]包含所有为目标form元素对象所引入的用户界面元素形成的数组(按钮、单选按钮)，且数组元素的下标按元素载入顺序分配。

encoding是表单的MIME类型，用enctype属性定义。在一般情况下，该属性是不必要的。

另外，form对象有submit()和reset()两个方法，可以通过这两个方法来提交数据或重置表单，而不需要用户单击某个按钮。

reset()该方法执行效果和点击Reset按钮相同。

submit()同Submit按钮。

(如果使用submit()方法向服务器发送数据或通过电子邮件发送数据，多数浏览器会提示用户是否核实提交的信息。如果用户取消发送，就再也无法这样发送了。)

2.在js中访问form对象

直接访问表单：document.myform(myform为表单name值)

通过数组访问表单：document.form[0]

3.表单数据的传递

原则：当函数需要访问表单中多于一个的表单对象时，传递整个form元素对象；当函数需要访问表单中某个特定的表单元素对象时，传递整个表单控件对象；当函数只需访问表单中特定的表单元素对象的某个属性时，传递该属性值的引用。

完全引用法：document.form[0].text.value(text为表单中元素的name)，这种方法虽然比较直观，但在需引用目标表单多个表单元素的场合，就会增加js脚本代码的冗余，执行效率不高，一般用于访问文档中多个表单的场合。

使用this.form作为参数传递：onchange="checkData(this.form)"，函数中使用targetForm.elements[0].value或targetForm.text.value。

直接传递：使用this作为参数，表示为当前元素的引用；this.form表示对当前整个表单的引用。函数中使用this.value获取值，或传递时直接使用this.value。

11.4.2编辑表单元素的脚本

1.文本框

常用事件：onclick,onblur,onchange,onfocus,onselect

单行文本框

多行文本框：wrap属性-定义输入内容大于文本域时显示的方式。默认值即virtual：文本自动换行，数据被提交处理时自动换行的地方不会有换行符出现；Off：用来避免文本换行，文字超出时文本将向左滚动，必须用return才能能插入点移到下一行；Virtual：允许文本自动换行；physical：让文本换行，当数据被提交时，换行符也一并被提交。

密码文本框

文件上传框：包含了一个浏览按钮。在使用文件域前，要先确定自己的服务器是否允许匿名上传文件。表单标签中必须设置enctype="multipart/form-date"，传送方式必须改为post。

2.按钮

普通按钮

重置按钮

提交按钮：点击提交按钮，相当于执行方法document.forms[0].submit()。

(使用图像按钮方法：创建一个a标签，内嵌img标签)

3.复选框

4.单选按钮

5.下拉列表框和组合框

11.4.3 用js获取网页内容实现数据验证

eg：document.forms[0].elements("txtName")

document.all.txtName

问题：

1.如何检查浏览器的版本？

var brower = navigator.appName;//浏览器名称

var b\_version = navigator.appVersion;

var version = parseFloat(b\_version);//浏览器版本

12.js的处理和调试

12.1常见的错误和异常

XHTML规范要求所有的属性值都必须使用双引号括起来。

12.2处理异常的方法

使用onerror事件处理异常和使用try-catch-finally模型。

12.2.1用onerror事件处理异常

使用onerror事件时一种早期的、标准的在网页中捕获js错误的方法。需要注意的是，目前chrome、opera、safari浏览器都不支持。

只要页面中出现脚本错误，就会产生error事件。如果需要利用onerror事件，就必须创建一个处理错误的函数-onerror事件处理器。参数msg(错误消息)、url(发生错误的页面url)、line(发生错误的代码行)。

eg：window.onerror = function(sMessage,sUrl,sLine){

alert('错误提示');

return true;

}

浏览器是否显示标准的错误消息，取决于onerror的返回值。如果返回值为false，浏览器的错误报告也会显示出来，所以为了隐藏报告，一般返回true。

另外图像对象也可触发onerror事件：document.images[0].onerror（没有定义图像的src就会触发onerror事件）

12.2.2用try-catch-finally语句处理异常

12.2.3使用throw语句抛出异常

问题：

1.如何优化js代码？

js的优化主要优化的是脚本程序代码的下载时间和执行效率，因为js运行前不需要进行编译而直接在客户端运行，所以代码的下载时间和执行效率直接决定了网页的打开速度，从而影响用户体验。

合理地声明变量：使用显示声明var，不用隐式声明变量。少声明全局变量。

简化代码：服务器上的代码，去除不必要的字符，包括注释、不必要的空格、换行等。

多使用内置的函数库：js的内置函数的属性犯法都是经过C、C++之类的语言编译的，而开发者自己写的函数在运行前还要进行编译，所以在运行速度上内置函数更快。

13.js和ajax技术

14.jQuery

14.5.2浏览器的兼容性

针对CSS3与浏览器的兼容性，用户可以通过http://www.css3.info网站来测试自己所使用的浏览器版本对属性选择器的兼容性程度。