**第一章JS免费课内容**

1. **网页的组成部分**
2. 网页的组成部分：结构层、表现层、行为层
3. 网页的结构决定了网页是什么？——HTML语言

网页的表现决定了网页看起来像什么？——css层叠样式表

网页的行为决定了网页在做什么？——JavaScript语言

1. **JS的组成部分**
2. JavaScript主要是为了网页交互而设计的一种轻量级的脚本语言
3. JS包括三部分：

1）ECMAscript：描述了变量、命名规范、数据类型、操作语句等核心语法知识——完全兼容

2）DOM(document object model)：提供了操作页面元素常用的属性和方法——有兼容问题，我们可以解决

3）BOM(browser object model)：提供了浏览器常用的属性和方法——兼容性问题很大，我们没办法解决

**三．JS引入页面的方式**

1. 行内式：不安全，不推荐使用

2. 内嵌式

3. 链接式（外链式和导入式）：如果放在head里，需要加Window.onload,如果放在页面底部，就不用加

**四．JS输出方式**

1. alert（“hello”）将内容以弹框的形式展示出来；缺点：不宜打印对象

2. confirm(“你要输出的内容”) 返回值：确定（true）取消（false）

3. console.log(“111”)将内容在控制台输出

4. console.dir(“调试引用数据类型”) 将内容在控制台输出，但是输出的内容比log详细

5. console.tabel ( ) 以表格的形式详细打印出对象

6. document.write(“直接将内容在页面显示出来”)

7. debugger(谷歌浏览器专有的调试方法)

**五．JS中变量命名规则和赋值**

**命名规则**

1. 严格区分大小写
2. 首字符只能是字母、下划线、$，不能是数字，其他字符可以是字母、数字、下划线、$
3. 开始是$符的一般是jQuery库或者其他的类库里声明的变量
4. object对象类型的原型，命名以o开头
5. 首字符是下划线的一般表示私有变量，如 var \_total=
6. 采用驼峰命名法，即由多个单词组合，从第二个单词开始首字母大写
7. 可以使用中文，但是不推荐

**赋值**

1. js是弱类型的语言，声明的变量在赋值之前，并不知道他是什么数据类型的，只有赋值之后才知道是什么数据类型
2. 没有赋值的变量返回的是undefined
3. 一个变量的值不能是undefined，undefined表示空的意思，不能让一个空等于另一个空，这是不符合常理的

**六．JS数据类型**

基本数据类型：Numeber, String, Boolean, Null, Undefined

引用数据类型：对象类型(object), 函数(function)

**区别**：基本数据类型操作的是值，引用数据类型操作的是地址

**七．JS基本数据类型Number**

1. 什么情况下会是number类型？

整数（正整数、负整数、0），小数，NaN(not a number)

2. 什么情况下会是NaN?

1)做数学运算符（+，-，\*，/, %）失败的时候

2)强制将其他数据类型转化成number类型失败的时候

3. 其他类型转成number类型的3种方式

1）Number（）

a.括号内是number类型的，值是简单的输入和传出

b.括号内是boolean类型的，Number(true)==1; ，Number(false)==0;

c.括号内是字符串类型的，只要有一个字符不是数字(**·**除外,只能有一个)，结果就是NaN，

注意 Number(“”)==0;

Number ( [ ] ) ==0;

Number( null )==0;

Number (NaN)==0;

Number(undefined)==NaN;

Number ( { } ) ==NaN { }是其他任何都会报错

Number( [undefined])==0; [ ]内只能是number类型，其他会报错

2）parseInt( )将字符串的整数部分提取出来

3）parseFloat( )除了能将字符串的整数部分提取出来，还能将字符串的小数部分提取出来

4. isNaN( )用来判断这个值是否是有效数字，如果是返回false，如果不是，返回ture；

isNaN( )检测分为两步：a.Number（）强制转化成数类型的，b.isNaN()去检测；

**八．JS基本数据类型 String**

1．什么是string类型？

用单引号或双引号包起来的就是字符串，单双引号必须是成对出现的，不成对会报错

2. 单双引号的使用规则：在双引号内可以用单引号，单引号内可以用双引号，但是双引号内不能再用双引号表示双引号，单引号内不能再用单引号表示单引号

3. +在字符串这边不再是运算符，而表示拼接的意思，在拼接的时候会自动把其他的数据类型转化成字符串类型

4. \ 表示转义，\r 表示回车，\n换行，\t制表符(相当于我们Word里面的tab键)

5.字符串的属性和方法：每个字符串都有下标,也叫索引(index)，从0开始

1）charAt(index) 通过下标找到目标小字符串

2）charCodeAt(index) 通过下标找到汉字的ASCII值

3) indexOf(string) 从前往后找通过汉字找到对应的下标

lastIndexOf(string) 从后往前找通过汉字找到对应的下标

4) str.substr(n,m) 从索引n开始截取m个

str.substring(n,m) 从索引n开始截取到索引m，包前不包后

str.splice(n,m) 和substring用法一样，不同是参数可以是负数

5) str.split(“字符”) 按照指定字符切分成n个字符串

6) str.toUpperCase() 转成大写

7) str.toLowerCase() 转成小写

8)

**九．JS基本数据类型 Boolean**

1．Boolean类型的值：true和false

2. 0, NaN, null, “”,undefined这5个转化成Boolean类型的结果是false，其他都是ture

3**. ！**表示取反，！有两层含义：1）先把！右边的值转化成布尔类型 2）将转化成的布尔值取反

！！相当于对自身做了布尔类型的转化 ！！===Bollean（）

**十．JS基本数据类型 Null和Undefined**

1.．null是一种标识性数据，表示我现在没有，但是我以后会有的

2. . undefined是一种标示性数据类型，表示压根从来没有过

3. null==undefined 结果为true；null===undefined 结果为false；

**十一.JS引用数据类型**

1. 引用数据类型(对象类型)是复合数据类型的载体，可以描述多个特征
2. 对象数据类型存在多对键值对(属性名和属性值)，每个键值对之间以逗号隔开，在普通对象里，属性名可以写双引号，也可以不写；属性名可以是数字，操作属性名是数字的这个属性，只能通过[ ]的方式，[ ]里的引号可以不写，一般不写；
3. 增；通过obj. 或obj[ ]的方式增加属性，[ ]里的属性名一定要用引号引起来
4. 删；obj.属性名=null或obj[“属性名”]=null假删除；delate obj.属性名 真删除；
5. 改；obj.属性名或obj[“属性名”]，先读取再修改，若属性存在就是修改，若不存在就是添加
6. 查；console.log(obj.属性名)或者console.log(obj[“属性名”])；[ ]里写了引号的表示常量，读取属性的时候，发现这个属性名不存在，则返回undefined；如果[ ]里不加引号，则表示变量，如果变量不存在则会报错
7. 遍历对象：在控制台里把对象的属性名和属性值一个一个的打印出来。

for..in循环可以实现遍历对象，有多少键值对就遍历多少次

例：for（var attr in obj）{console.log(obj[attr])} 变量名一般用key和attr; 只能通过obj[key]来获取属性值，不能用obj.key,而且[ ]内不能用引号,加上引号表示固定属性名

1. 自定义属性：给一个元素上添加一个属性并给他赋值；页面加载时首先会干掉不是系统里规定的元素属性，只有一次

**十二. 流程控制语句：判断和循环**

1. for循环：执行次数固定的情况下使用；
2. while循环：循环次数不固定的情况下使用，

for和while循环相同点：都可以写成四步a.定义变量b.条件判断c.执行语句d.自增或自减

3. do..while循环：不管条件是否成立都会执行一次do{ }while( );

1. for…in循环：用来循环遍历对象中的属性名和属性

——以上是三个循环——

1. if..else判断
2. 三目：三元表达式，用于简单的判断；写法：条件？结果1：结果2；
3. switch…case判断

———以上是三个判断———

十三. **JS的引用数据类型 ary**

1. 数组是把对象里属性名是数字的单独独立出来，为了方便操作；在数组这边数字属性名称为索引，并且索引是隐藏的；数组是有序排列的集合，第一项索引为0，最后一项是length-1；
2. 数组常用的方法

1）push() 往数组末尾添加一项内容，**参数：**添加的内容；**返回值：**添加完内容后数组的长度

其他方法：ary.length=x; ary[ary.length]=x；ary.aplice(length-1,0,x)

2）pop() 删除数组的最后一项内容,**没有参数**；**返回值**：删除的内容

其他方法：ary.length--;

3）shift() 删除数组的第一项内容，**没有参数**；**返回值**：删除的内容

4）unshift() 将内容添加到数组的首位置，**参数**：添加的内容；**返回值**：添加后数组的 长度

5）splice(n) 从索引n的位置删除到最后，**返回值**：删除的内容

splice(n,m) 从索引n的位置开始，删除m个；返回值：删除的内容

splice(n,m,x) 从索引n的位置开始删除m个，用x（可以是多项）代替

splice(n,0,x) 把x的内容添加到索引n之前

6) reverse() 将数组倒着排列；**返回值**：倒着排列后的数组

7) sort() 十以内的数从小到大排列；

sort(function(a,b){return a-b})

——以上是原有数组发生改变的

8）slice（n）从索引n开始一直截取到最后

slice(n,m) 从索引n开始,一直截取到索引m(包前不包后),**返回值**:截取的新的数组

slice的**参数可以是负数**(这时候的索引=负数索引+索引的长度)

9）join（）按照指定的字符拼接成字符串，**返回值：**拼接后的字符串；

可以把用eval把字符串转换成js语句

10）concat() 将多个数组拼成一个数组，**返回值**：拼接后的新数组

11) indexOf(n) 从数组的前面往后找,如果找不到**返回—1**,如果能找到,则**返回这一项的索引值**

12) lastIndexOf() 从数组的后往前找,如果找不到返回—1,如果能找到,则返回这一项的索引值， ——以上这两个方法可以判断数组里是否有这一项

13) toString() 把数组转换成字符串

——以上是原有数组没有发生改变的

**十四.JS的引用数据类型 Math**

1. math的常用方法

1）Math.abs();求绝对值

2）Math.floor();向下取整

3）Math.ceil();向上取整

4）Math.round();四舍五入

5）Math.random();0到1之间的随机数

6）Math.toFixed();取小数点的位数

7）Math.max();取最大值

8）Math.min();去最小值

9）Math.sqrt();开平方

10) Math.pow();幂次方

**十五.JS的引用数据类型 function**

1. 函数包含两部分:定义和调用

函数定义的步骤：1）开辟一个空间地址2）把函数中的JS代码作为字符串存在这个空间地址中3）把空间地址赋值给函数名

函数调用的步骤：1）开辟一个新的私有作用域2）把以前空间中存在的字符串类型的代码作为js代码来执行

1. 函数封装：对于相同的功能进行封装，以后再需要此功能，只需要调用即可，避免冗余代码，优化执行

函数封装步骤：1）首先要HTML代码结构一致，在父级下获得多个元素2）主要程序的实现3）传参

1. 参数：

参数的第一层理解：1）形参：如果定义了形参，没有传值的时候，结果就是undefined；

2）arguments：函数内置的对象参数机制，用来存所有的实参，不管调用时是否赋值都可以；arguments不是数组，是类数组；

3）length属性：代表传的参数个性

4）callee方法：代表函数本身这个整体

参数的第二层理解：参数可以为我们学过的所有数据类型

1. return：
2. 作用一：函数返回值；函数返回值是undefined：一种是没有写return，一种是写了return没有返回值
3. 作用二：阻断函数里面代码的执行；可以让函数里面的代码执行到我们想要的位置停止
4. 闭包：函数执行的时候，会形成一个新的私有作用域，保护里面的变量不受外界的干扰，函数的这种保护机制，叫做闭包

————以上都是实名函数————

1. 常见的匿名函数：
2. 函数表达式：把函数定义部分作为值赋值给变量或元素事件行为
3. 自执行函数：函数定义和调用一起完成

自执行函数的写法：a.(function(){})()；b. ~function(){}()；c. !function(){}();

d. +function(){}()；e. +function(){}();

**十六.数据类型的转化**

1. = 是赋值的意思；== 是非严格s比较；===是严格比较
2. ==两边数据类型不一样的时候，就会发生数据类型的转变
3. 对象==对象；比较的是引用地址
4. [ ]==[ ]；false
5. 对象==字符串；
6. 对象==布尔；
7. 对象==数字；
8. 布尔==字符串；
9. 布尔==数字；
10. 字符串==数字；
11. NaN==NaN；NaN和任何数比较都不相等，包括它自己
12. null==undefined；true;
13. null===undefined；false

**十七．DOM常用的操作元素的方法**

1. document.getElementById("div1") 通过id名获得元素,获得的元素是唯一的.

2. document.getElementsByTagName("p") 通过标签名获得一组元素

3. document.getElementsByClassName("a") 通过类名获得一组元素，IE6-IE8不兼容

4. document.getElementsByName("a") 通过name属性获得一组元素，主体只能是document，IE不兼容非表单元素的name属性，所以一般用在表单元素得选择

5. document.querySelector(".p1"); 通过选择器获取一个元素——移动端

6. document.querySelectorAll("ul>li");通过选择器获取一组元素——移动端

7. document.documentElement 获得整个HTML

8. document.body 获得整个body

9. documetn.documentElement.clientWidth||document.body.clientWidth; 获得可视窗口的宽

**十八.动态操作DOM**

1. createElement; 创建元素节点，主体只能是document
2. createTextNode; 创建文本节点 主体只能是document
3. appendChild; 把文本节点添加到页面上，添加到末尾的位置，主体是操作元素的父级元素 ，如果页面上已经有这个节点，就是移动，而不是添加
4. insertBefore（newEle,oddEle）; 把文本节点添加到页面上，添加到末尾的位置，主体是操作元素的父级元素；如果页面上已经有这个节点，就是移动，而不是添加
5. repalceChild(newEle,oddEle); 用newEle去替换oddEle,操作元素是父标签
6. removeChild(); 删除某个节点
7. cloneNode（true）; 深克隆，参数为false或空表示浅克隆
8. setAttribute(属性名,值)；设置元素属性(自定义属性/固有属性)
9. getAttribute
10. removeAttrbute 通过setAttribute设置的属性，必须通过getAttribute和removeAttrbute来操作

**十九.自定义属性**

1. 自定义属性：在元素标签上自己定义一个属性名和属性值
2. 问题：浏览器在加载页面的时候，会自动过滤掉他不认识的属性和属性值；解决办法：浏览器加载完成后，我们通过JS来动态加载属性和属性值
3. 我们可以获得系统自带的属性值，我们自定义的属性值，不能通过 **.** 来获取

**二十.节点**

节点类型 nodeType nodeName nodeValue

文本节点 3 #text 文本内容（包括回车、空格）

元素节点 1 大写的标签名字 null

注释节点 8 #comment 注释内容

文档节点 9 #document null

**二十一.节点与节点之间的关系**

1. firstChild 第一个子节点
2. firstElementChild 第一个子元素节点
3. lastChild 最后一个子节点
4. lastElementChild 最后一个子元素节点
5. previousSibling 最近的哥哥节点
6. previousElementSibling 最近的哥哥元素节点
7. childNodes 所有的子节点
8. children 所有的子元素节点
9. parentNode 父节点(唯一的)
10. 带有element的IE6-8不兼容

**二十二.定时器**

1. setTimeout([function(){}]),duration); duration秒之后开始调用方法，只执行一次
2. clearTimeout(timer),清除定时器
3. setInterval ([function(){}]),duration);每隔duration秒调用一下方法本身

**r**