课程体系说明：

标号 1 表示必须掌握；

标号 2 表示能够理解；

标号 3 表示简单了解；

标号 4 表示扩展内容；

1. JavaScript 概述

（1）简称 JS

（2）一种脚本语言，脚本语言的特点

Java 语言是一种非脚本语言，属于编译型语言。

JavaScript 语言是一种脚本语言（解释型语言），JavaScript 的“目标程序”是以普通文本的形式保存。用记事本是可以直接打开的。浏览器打开就直接解释执行了。

（3）JavaScript 和 JScript 的关系

JavaScript 是网景公司开发的，javascript 之父是 布兰登艾奇。JavaScript 前身叫做 LiveScript。当时网景公司开发 JavaScript 的目的是为了占领“浏览器”市场。网景公司有一个浏览器，当时非常著名：领航者浏览器 Navigator。JavaScript 语言是为领航者浏览器专门量身打造的。JavaScript 只支持 Navigator 浏览器，其它浏览器不支持。这个时候微软慌了，马上组建团队，开发了一种编程语言叫做 Jscript，专门和 JavaScript 抗衡的，只支持 IE 浏览器。网景公司在某个历史阶段，和 SUN 公司有合作，SUN 公司把 LiveScript 改名为 JavaScript。

（4）JavaScript 主要用来操作 HTML 中的节点，产生动态效果

JavaScript 是一门编程语言，专门用来操作 HTML 页面中的节点，让网页产生动态效果的。JavaScript 中也有变量、数据类型、运算符、if 语句、for 循环、标识符、关键字等。

（5）JavaScript 和 Java 的区别

JavaScript 运行在浏览器当中，浏览器中有执行 JS 代码的内核。

Java 运行在 JVM 当中。JavaScript 和 Java 没有任何关系。

Java 语言是 SUN 公司开发的，JavaScript 这个名字是 SUN 公司给起的名。

JavaScript 选择是对的，真的搭上了 Java 的顺风车！

2. JavaScript 包括三块：ECMAScript、DOM、BOM

3

（1）ECMAScript 是 ECMA 制定的 262 标准，JavaScript 和 JScript 都遵守这个标准，ECMAScript 是 JavaScript 核心语法

（2）DOM 编程是通过 JavaScript 对 HTML 中的 dom 节点进行操作，DOM 是有规范的，DOM 规范是 W3C 制定的。（Document Object Model：文档对象模型）

（3）BOM 编程是对浏览器本身操作，例如：前进、后退、地址栏、关闭窗口、弹窗等。由于浏览器有不同的厂家制造，所以 BOM 缺少规范，一般只是有一个默认的行业规范。（Browser Object Model：浏览器对象模型）

3. 嵌入 JS 三种方式以及 JS 的注释

3.1. *行间事件*

1

（1）<input type="button" value="hello" onclick="window.alert('hello js')" />

（2）JS 是一种基于事件驱动型的编程语言，当触发某个事件之后，执行一段代码

（3）JS 中的任何一个事件都对应一个事件句柄，例如鼠标单击事件 click，对应的事件句柄就是 onclick，事件句柄都是以标签的属性方式存在。在事件句柄后面可以编写 JS 代码，当触发这个事件之后，这段 JS 代码则执行了。

（4）JS 中的字符串可以使用单引号括起来，也可以使用双引号括起来

（5）window 是 JS 中的内置 BOM 顶级对象，代表当前浏览器窗口，window 对象有一个 alert()函数，该函数可以在浏览器上弹出消息框。

（6）JS 中的一条语句结束后可以使用“;”结尾，也可以不写。

（7）window.alert()中的 window.可以省略。

3.2. *页面 script 标签嵌入*

1

（1）<script type="text/javascript">JS 代码</script>

（2）window.alert()的执行会阻塞当前页面的加载

（3）一个页面中可以写多个脚本块

（4）脚本块的位置没有限制

（5）暴露在脚本块中的 JS 代码在页面打开的时候遵循自上而下的顺序依次逐行执行

3.3. *外部引入*

1

（1）<script type="text/javascript" src="js 文件路径"></script>

（2）<script type="text/javascript" src="js 文件路径">这里不能写 JS 代码</script>

（3）这种写法错误：<script type="text/javascript" src="js 文件路径"/>

4. 标识符和关键字

3

（1）标识符命名规则和规范按照 java 执行

（2）关键字不需要刻意记

5. 变量

5.1. *变量的声明与赋值*

1

（1）变量未赋值，系统默认赋值 undefined

（2）JS 是一种弱类型编程语言，一个变量可以接收任何类型的数据

（3）一行上也可以声明多个变量

5.2. *函数的定义与调用*

1

（1）函数类似于 java 语言中的方法，是一段可以完成某个功能的可以被重复利用的代码片段

（2）定义函数的两种语法

第一种：普通函数定义，这种方式较多

```
function  函数名(形式参数列表){
    函数体;
}
```

例如：

```
function sum(a, b){
    return a + b;
}
```

注意：

a 和 b 是形式参数列表，也是两个局部变量。

JS 中的函数不需要指定返回值类型，因为 JS 是弱类型编程语言，变量可以接收任何类型的数据，也就是说 JS 中的函数可以返回任何类型的数据，当然也可以不返回任何数据。返回数据使用 return 语句。

JS 中的函数在调用的时候，实参可以随意，例如调用以上的 sum 函数，可以这样调用：sum()，没有传任何实参的时候 a 和 b 变量没有赋值，则 a 和 b 都是 undefined。也可以这样调用 sum(10)，这样就表示 a 变量赋值 10，b 变量仍然是 undefined。还可以这样调用：sum(1,2)，这样则表示 a 是 1，b 是 2。

第二种：如果是把函数的声明当做类进行定义这种方式较多

```
函数名  = function(形式参数列表){
    函数体;
}
```

例如：

```
sum = function(a, b){
    return a + b;
}
```

（3）JS 中的函数定义在脚本块中，页面在打开的时候，函数并不会自动执行，函数是需要手动调用才能执行的。

（4）由于 JS 是一种弱类型编程语言，所以函数不能同名，没有重载机制

（5）这样的代码顺序是可以的，页面打开的时候会先进行所有函数的声明，函数声明优先级较高。

```
<script type="text/javascript">
    sayHello();
    function sayHello(){
        alert("Hello JS");
    }
</script>
```

（6）用户点击按钮，调用函数

```
<script type="text/javascript">
    function sayHello(){
        alert("hello js");
    }
</script>
<input type="button" value="hello" onclick="sayHello();"/>
```

5.3. *局部变量和全局变量*

（1）局部变量：函数的形参是局部变量，另外使用 var 关键字在函数体中声明的变量是局部变量，函数执行结束之后，局部变量的内存就释放了。

（2）全局变量：在函数体外声明的变量属于全局变量，另外不使用 var 关键字声明的变量无论位置在哪，它都是全局变量，全局变量在浏览器关闭时销毁。

6. JS 数据类型

6.1. *typeof 运算符*

6.1.1. JS 中为什么会有 typeof 运算符

2

6.1.2. typeof 运算符怎么用，代码怎么写

1

语法格式是：

```
function sum(a, b){
    if("number" === typeof a && "number"=== typeof b){
        return a + b;
    }
    alert("数据格式不合法");
    return 0;
}
```

6.1.3. typeof 运算符的运算结果都是全部小写的字符串

1

"undefined"

"number"

"string"

"boolean"

"object"

"function"

6.2. *ES6 版本之前的数据类型有 6 种*

6.2.1. Undefined

1

只有一个值 undefined，变量声明没赋值，系统默认赋值 undefined

6.2.2. Number

（1）Number 类型包括哪些值：0,1，-1,3.14,12,300，NaN，Infinity

（2）parseInt()函数

（3）parseFloat()函数

（4）Math.ceil()函数：向上取整

（5）isNaN()函数

6.2.3. String

（1）可以使用单引号，也可以用双引号

（2）JS 中的字符串包括小 String，也包括大 String，小 String 属于原始类型，大 String 是 JS 的内置对象，大 String 属于 Object 类型。

（3）无论大 String 还是小 String，它们的属性和方法都是通用的。

（4）字符串中常用方法讲一些，主要讲解字符串的 substr()和 substring()的区别。

6.2.4. Null

（1）该类型只有一个值：null

（2）typeof 运算符的执行结果是"object"

6.2.5. Boolean

（1）只有两个值：true 和 false

（2）Boolean()函数

（3）JS 中的 if 语句自动调用 Boolean()函数。

6.2.6. Object

1

（1）JS 中如何定义一个类。

（2）JS 中如何创建一个对象。

（3）JS 中如何访问对象属性，调用对象的方法。

（4）JS 中的一个函数，既是函数声明，又是类的定义，同时函数名也可以看做构造方法名。直接调用函数表示普通函数调用，如果使用 new 运算符来调用该函数则会创建对象。

（5）使用 prototype 属性动态的给对象扩展属性以及方法。

6.3. *ES6 版本及之后包括的数据类型有 8 种*

3

除了以上 6 种类型之外，还有两种类型分别叫做：Symbol 和 BigInt

7. null NaN undefined 区别

1

（1）=、==、===三者的区别

（2）null NaN undefined 三者类型不同，null 和 undefined 的值可以等同

8. JS 中的事件

8.1. *常用事件*

1

（1）blur 失去焦点

（2）change 下拉列表选中项改变，或文本框内容改变

（3）click 鼠标单击

（4）dblclick 鼠标双击

（5）focus 获得焦点

（6）keydown 键盘按下

（7）keyup 键盘弹起

（8）load 页面加载完毕

（9）mousedown 鼠标按下

（10）mouseover 鼠标经过

（11）mousemove 鼠标移动

（12）mouseout 鼠标离开

（13）mouseup 鼠标弹起

（14）reset 表单重置

（15）select 文本被选定

（16）submit 表单提交

8.2. *注册事件的两种方式*

（1）在标签中使用事件句柄的方式注册事件

```
<body onload="sayHello()"></body>
```

（2）在页面加载完毕后使用 JS 代码给元素绑定事件

```
<script>
    window.onload = sayHello;
</script>


<script>
    window.onload = function(){
```



```
    }
</script>
```

重点：通过事件注册，理解回调函数的概念

8.3. *代码的执行顺序*

这是一种错误的写法：

```
<body>
    <script type="text/javascript">
        var elt = document.getElementById("btn");
    </script>
    <input type="button" id="btn" value="mybtn"/>
</body>
```

这样写：

```
<body>
    <input type="button" id="btn" value="mybtn"/>
    <script type="text/javascript">
        var elt = document.getElementById("btn");
    </script>
</body>
```

或者这样写：

```
<body>
    <script type="text/javascript">
        window.onload = function(){
            var elt = document.getElementById("btn");
```



```
        }
    </script>
    <input type="button" id="btn" value="mybtn"/>
</body>
```

8.4. *通过 keydown 事件演示回车键 13，ESC 键 27*

1

9. JS 运算符之 void

1

Js 中其它运算符和 java 一样用。void 主要讲：javascript:void(0)的用法。

10. JS 之控制语句

3

告诉学生控制语句和 Java 一样用，课堂上不再讲解。只讲一下 for..in 语句的使用，使用 for..in 语句遍历数组，以及遍历一个对象的属性。

11. JS 内置对象

11.1. *Array*

1

（1）创建数组

（2）JS 中的数组特点

（3）JS 中数组对象常用方法：push，pop，join，reverse 等。

（4）数组遍历

11.2. *Date*

1

（1）new Date() 获取当前系统时间

（2）new Date().getTime()获取时间戳

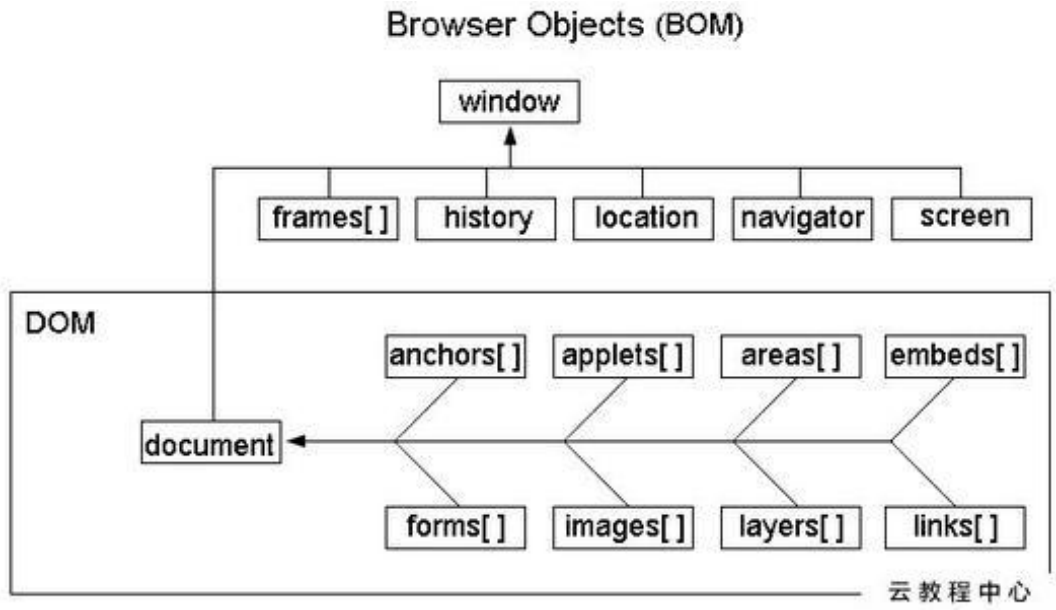
（3）new Date().getFullYear()、getMonth()等方法。

12. BOM 和 DOM 的区别与联系

2

BOM: Browser Object Model（浏览器对象模型），通过 BOM 的对象和方法可以完成浏览器窗口的操作，例如：关闭浏览器，前进，后退，修改地址栏上的地址等，这些操作都属于 BOM。BOM 的顶级内置对象是 window。

DOM: Document Object Model（文档对象模型），通过 DOM 的对象和方法可以完成网页中元素的增删改，让网页产生动态效果，DOM 的顶级内置对象是 document。



13. DOM 编程案例

13.1. *innerHTML innerText 操作 div 和 span*

1

13.2. *JS 的正则表达式(Regular Expression)*

13.2.1. 正则表达式概述

2

（1）正则表达式是一门独立的学科，不止用在 JS 中

（2）正则表达式专门用来做字符串格式匹配的

13.2.2. 常用的正则表达式符号

1

参考 30 分钟入门正则表达式：

^ 字符串开始

$ 字符串结束

\s 空白

* 0~N 次

+ 1~N 次

? 0 或 1 次

{3} 3 次

{3,} 3~N 次

{3,5} 3~5 次

(a|b) a 或 b

[a-z] a 到 z

[^abc] 不是 abc

13.2.3. 会写简单的正则表达式

1

（1）qq 号正则

（2）必须由数字和字母组成，不能含有其它符号的正则

（3）给学生一些常用的正则表达式

13.2.4. 会创建 JS 中的正则表达式对象

1

（1）var regExp = new RegExp("^[1-9][0-9]{4,}$");

（2）var regExp = /^[1-9][0-9]{4,}$/;

13.2.5. 会调用 JS 中正则表达式对象的 test()函数

1

写一个校验用户名只能由数字和字母组成的案例

13.3. *表单验证*

1

（1）用户名不能为空

（2）用户名必须在 6-14 位之间

（3）用户名只能有数字和字母组成，不能含有其它符号（正则表达式）

（4）密码和确认密码一致，邮箱地址合法。

（5）统一失去焦点验证

（6）错误提示信息统一在 span 标签中提示，并且要求字体 12 号，红色。

（7）文本框再次获得焦点后，清空错误提示信息，如果文本框中数据不合法要求清空文本框的 value

（8）最终表单中所有项均合法方可提交

13.4. *复选框全选和取消全选*

1

document.getElementById()

document.getElementsByName()

document.getElementsByTagName()

以上三个函数告知学生很重要

13.5. 获取下拉列表选中项的 value

1

change 事件

13.6. 显示网页时钟

1

window.setInterval()

window.clearInterval()

主要两个函数

捎带着提一下 window.setTimeout()

13.7. 拼接 html 的方式，设置 table 的 tbody

1

```
<table>
    <thead></thead>
    <tbody id="userListTbody"></tbody>
</table>
<script>
    var html = "";
    html += "<tr>";
    html += "<td>";
    html += "zhangsan";
    html += "</td>";
    html += "<td>";
    html += "2000-10-11";
    html += "</td>";
```



```
    html += "</tr>";


    html += "<tr>";
    html += "<td>";
    html += "lisi";
    html += "</td>";
    html += "<td>";
    html += "2001-10-11";
    html += "</td>";
    html += "</tr>";


    var userListTbody = document.getElementById("userListTbody");
    userListTbody.innerHTML = html;
</script>
```

14. BOM 编程案例

14.1. *window.open()和 window.close()*

1

14.2. *window.alert()和 window.confirm()*

1

14.3. *如果当前窗口不是顶级窗口，将当前窗口设置为顶级窗口*

1

```
if(window.top != window.self){
    window.top. location = window.self. location;
```

```
}
```

14.4. *历史记录*

1

```
window.history.back(); window.history.go(-1); window.history.go(1);
```

14.5. *window.location.href*

1

提示一下 document.location.href 也可以完成同样功能