20230914

javaScript

1.js简介

1. 发展历程

   Netscape（网景）->LiveScript脚本语言，Netscape+Sun（开发java的）->javaScript

2. 特点

   脚本语言：是一种解释型的脚本语言（c c++ 先编译后执行，js可以直接执行）

   基于对象：是一种基于对象的脚本语言，可以创建对象，也可以使用现有对象

   简单：js采用若类型的变量类型，同时对编码格式要求没有特别严谨

   跨平台：js只依赖于浏览器，与操作系统内核无关

   嵌入式：需要在html页面上操作html元素，因此需要调用浏览器提供的html接口。

   开发js需要具备：html css技术

3. 作用

   对浏览器事件做出响应，读写html元素，css样式，数据提交之前验证，检测方可的浏览器信息，控制本地存储cookie，基于Node.js技术进行服务器端编程

2.js内容

1. 入门语法

   1. helloworld范例

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
    <script>
        //弹窗：js向页面输出内容的一种互动方式
        //弹窗-警告框
        alert('Hello,world');
    </script>
</head>
<body>
    <button onclick="myFunction()">Touch me</button>
    <p id="demo"></p>
</body>
<script>
    //向页面输出
    //document代表对文档对象，write（）是该对象的一个函数/方法
    document.write('Hello world')
    document.write('<h1>hello world</h1>')
    function myFunction(){
    var world = "Hello World!"
```

```
    document.getElementById('demo').innerHTML= world;
    }
</script>
</html>
```

2. 外部引入

```
<script src="../../js/js.js"></script>
```

3. 变量与常量

变量：随着程序执行，根据代码需求，不断变化的值，我们把这种值叫变量。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>变量与常量</title>
</head>
<body>
    <script>
        // 变量声明并赋值
        // 格式: var 名字=变量值
        // var varible变量的意思
        var i = 1000;
        console.log(i);

        // 变量重新赋值
        // i=100;
        // console.log(i);
        // for(var i=0;i<100;i++){
        //     document.write(i)
        // }

        // 变量只有声明没有赋值,没有赋值，默认值为uundefined
        var j;
        console.log(j)

        //！！！变量没有声名，没有赋值，报运行错误
        console.log(k);

    </script>
</body>
</html>
```

4. 变量的命名规则

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
```

```
    <title>Document</title>
</head>
<body>
    <script>
        //合法的变量名称
        // 以字母或者下划线，或者美元符开头的名称，不能以数字开头，不能用var这类的来定义
        // 命名规范1. 见名知意2.小写字母开头，驼峰式命名或者下划线分割 oneTwoThree/one_two_three3.如果有数字，必须放在变量最后
        var a100;
        var _abc;
        var $x1000;
    </script>
</body>
</html>
```

5. 变量不声明直接使用

```
// 表示 使用严格模式开发
'use strict';
// 正常模式下 可以正常使用（没有声明 有赋值）
i=100;
alert(i);
```

6. 数据类型

2. 运算符

1. 算数运算符

+加-减*乘/除%取余，-取反

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var a=10,b=3;
        console.log(a+b);
        console.log(a-b);
        console.log(a*b);
        // 1.js除法结果如果除不尽则显示浮点数
        // 2.最后一位数字可能会出现精度丢失现象
        // 3.除不尽浮点数有效位数为16位
        console.log(a/b);
        console.log(a%b);
        // 求余作用：1.判断奇偶
        // 2.能够获取一个大数值的某个位数上的值 12345 获取千位上的数字 对10000求余 2345除1000取整
        console.log(-a);
    </script>
</body>
</html>
```

2. 自增自减运算符

```
var i = 0;
i++;
console.log(i);
```

3. 前置后置运算

```
var i=10;
// ++i: 先自增,再进行其他运算 i++: 先进行其他运算,最后自增
// var j=++i;
var j=i++;
// console.log(i);
// console.log(j);
// 多个变量,合并输出 弱语言可以
console.log(i,j);
```

4. 赋值运算符

```
// = 等号右侧的值 给等号左侧赋值
var a=10;
a+=5; //等同于 a=a+5
a-=3; //a=a-3
a*=2; //a=a*2
a/=3; //a=a/3
a%=3; //a=a%3
console.log(a);
// 连等
var i=10;
var n;
var j=n=i;
alert(j);
```

5. 比较运算符

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        // 比较运算符可以连接数值，构成表达式，该表达式结果为boolean
        console.log(5>3)

        var a = 5,b=6;
```

```
        console.log(a>b);
        console.log(a<b);
        console.log(a>=b);
        console.log(a<=b);
        console.log(a==b);
        console.log(a!=b);

        //判断是否相等的等号 ==与===
        console.log(1==1)
        console.log(1===1)
        // 其中==只比较直===比较数据类型
        console.log(1=='1')
        console.log(1==='1')
        // 字符串比较 比较的是 首字母的unicode或者ascii的值
        console.log('tom'>'jack')
        //boolean的比较ture自动转换为1 false转换为0
        console.log(true>false)
        console.log(true==1)

        //其他
        console.log(""==0)
       console.log(false=="")
        console.log(false==null)
        console.log(null==undefined)

        console.log("100"/5);//自动计算了
        console.log('abcd'/5);
    </script>
</body>
</html>
```

6. 字符串拼接

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        // 字符串拼接符
        var str = 'hello'
        str = str+'world'
        console.log(str)
        // 作用场景：当加号两侧至少有一侧不为数值，则加号起拼接作用。不做加法运算
        console.log(1+2+3+'4'+5+6)
    </script>
</body>
</html>
```

7. 转义字符

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var str = 'hello\'111\' world'
        console.log(str);
    </script>
</body>
</html>
```

8. 字符串拼接案例

```
// 拼接操作 将变量X 拼接到ABD中指定位置
// X="C"
// 'ABD'
// 'AB'+'D'
// 'AB'+X+'D'
// prompt()函数赋值,值为用户在输入框输入的内容 点击取消，则为null值
var color=prompt("请输入颜色名称");
var width=prompt("请输入宽度");
// 使用字符串 拼接为hr标签
var html_str='<hr width="'+width +'" color="'+ color+'">';
console.log(html_str);
document.write(html_str);
```

9. 字符串拼接案例2

```
var path=prompt("请输入图片的名称");
var html_str='<img src="../../img/'+path+'"/>'
document.write(html_str);
```

10. ES6模板字符串

    ES:ECMA Script为js制定了ECMA标准

    最新标准ES6

```
var path=prompt("请输入图片的名称");
// ES6语法进行插值
// ${变量名}
// 整个字符串 改为 反引号包裹
var html_str=`<img src="../../img/${path}">`;
console.log(html_str);
document.write(html_str);
```

11. 逻辑运算符

与 &&或||非！逻辑运算符连接的是boolean或结果为布尔值的表达式

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        // 逻辑与 并且所有条件同时满足 结果为才为真 否则是假 两个都得是真的
        console.log(true&&true)
        console.log(false&&true)
        console.log(false&&false)

        // 逻辑或（或者），条件中只要一个真的，就是真
        console.log(true||true)
        console.log(false||true)
        console.log(false||false)

        //逻辑非取反
        console.log(!true)
        console.log(!false)

        //应用
        console.log(4<6&&8<0)
        //优先级 !>&&>||
        console.log(4<6||10<2&&5<3)
        console.log(true||false&&false)
        //通过加括号调整判断顺序
        console.log((4<6||10<2)&&5<3)
    </script>
</body>
</html>
```

12. 短路

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var a=10;

        var result = 5<3&&++a>10; //++a没有机会执行
        // 短路现象：表达式1&&表达式2&&……当表达式1为fales时，结果已经确定，表达式2
以及后续的表达式不再执行
        console.log(result);
```

```
        console.log(a);
    </script>
</body>
</html>
```

13. 逻辑运算符的其他

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        // var result = 5&&3;
        var result=0||3||5

        console.log(result);

        result=undefined&&'jack';
        result=undefined||'jack';
    </script>
</body>
</html>
```

14. 三目运算符

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var result=5<3?10:100;
        var result=5<3?"abc":"cba"
        console.log(result);

        var a = 40;
        var b = 20;

        var max = (a > b) ? a : b;

        console.log(max); // 输出 20

    </script>
</body>
```

```
</html>
```

15. 三目运算案例

    三目运算符：两侧除了写运算符之外，也可以写执行代码

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var score=prompt('puat your score in the document')

        score>=60?document.write('you are good'):alert('you are faled')
    </script>
</body>
</html>
```

16. 浮点运算精度丢失

    统一将附带逆运算改为整数运算，结果调整为浮点数

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var a=0.1,b=0.2;

        var c=
    </script>
</body>
</html>
```

17. string转换number

    parseInt（需要转换类型的数字）

    Number（需要转换类型的数字）

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var a = prompt("请输入数字1")
        var b = prompt("请输入数字2")

        var c = a+b //结果为12说明prompt()结果为string字符串类型

        var c = parseInt(a)+Number(b) //
        // 如果无法转换成数字显示NaN

        console.log(c)
    </script>
</body>
</html>
```

18. parsInt和parsFloat

    parsenInt字符串转换数字

    parsenFloat字符串转换浮点数

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var a = '100.5'
        var b = parseInt(a);//取整抹除小数部分

        console.log(b);
    </script>
</body>
</html>
```

19. isNaN()

    判断是否不是数字

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
```

```
</head>
<body>
    <script>
        var a = 'abcd'
        console.log(isNaN(a))
    </script>
</body>
</html>
```

作业

1.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var inPutNumber = prompt("请输入一个五位数")

        if (inPutNumber.length === 5 && !isNaN(inPutNumber)){

            var number = parseInt(inPutNumber)

            var num5 = number%10
            number = Math.floor(number/10)
            var num4 = number%10
            number = Math.floor(number/10)
            var num3 = number%10
            number = Math.floor(number/10)
            var num2 = number%10
            number = Math.floor(number/10)
            var num1 = number%10
            number = Math.floor(number/10)

            console.log(num1)
            console.log(num2)
            console.log(num3)
            console.log(num4)
            console.log(num5)
        }
        else{
            console.log('无效')
        }
    </script>
</body>
</html>
```

2.

```
<!DOCTYPE html>
```

```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>
    <script>
        var a = 1;
        var b = 2;

        var c = a;
        a=b;
        b=c;

        console.log(a,b);
    </script>
</body>
</html>
```

3.

```
<!DOCTYPE html>
<html>
<head>
    <title>动态创建表格</title>
</head>
<body>

<script>
    // 弹窗输入表格宽度
    var width = prompt("请输入表格宽度（单位：像素）:");

    // 弹窗输入表格高度
    var height = prompt("请输入表格高度（单位：像素）:");

    // 弹窗输入背景颜色
    var bgColor = prompt("请输入背景颜色:");

    // 弹窗输入对齐方式
    var textAlign = prompt("请输入对齐方式（left, center, right）:");

    // 弹窗输入单元格1的内容
    var cell1Content = prompt("请输入单元格1的内容:");

    // 弹窗输入单元格2的内容
    var cell2Content = prompt("请输入单元格2的内容:");

    // 创建表格元素
    var table = document.createElement("table");
    table.style.width = width + "px";
    table.style.height = height + "px";
    table.style.backgroundColor = bgColor;

    // 创建表格行
    var row = document.createElement("tr");
```

```
    // 创建单元格1
    var cell1 = document.createElement("td");
    cell1.style.textAlign = textAlign;
    cell1.textContent = cell1Content;

    // 创建单元格2
    var cell2 = document.createElement("td");
    cell2.style.textAlign = textAlign;
    cell2.textContent = cell2Content;

    // 将单元格1和单元格2添加到行中
    row.appendChild(cell1);
    row.appendChild(cell2);

    // 将行添加到表格中
    table.appendChild(row);

    // 将表格添加到页面
    document.body.appendChild(table);
</script>

</body>
</html>
```