第二十二天到第二十四天：JavaScript里面的居民们

普通

Varsha (/mentor/detail/id/2) | 零基础学院 (/college/detail/id/5)

开始学习

人数　有 346 人在学习该课程，有 273 人已经完成该课程

时间　平均用时 3.8 天

关键词　JavaScript

课程概述

作业提交截止时间：09-01

第二十二天到第二十四天：JavaScript里面的居民们

课程目标

掌握 JavaScript 中的各个数据类型、对象的概念及常用方法，这次课程的任务量比较多，但不要着急，也不要急于完成任务，认真写好每一个代码。加油！

课程描述

阅读

首先，我们从变量和数据类型入手，同时学习一下 JavaScript 中的数字类型

- W3School 变量 (http://www.w3school.com.cn/js/js_variables.asp)
- W3School 数据类型 (http://www.w3school.com.cn/js/js_datatypes.asp)
- JavaScript中值类型和引用类型的区别 (https://blog.csdn.net/lxcao/article/details/71314605)
- MDN 变量 (https://developer.mozilla.org/zh-CN/docs/Learn/JavaScript/First_steps/Variables)
- W3School 数字 (http://www.w3school.com.cn/js/js_obj_number.asp)
- MDN 数字 (https://developer.mozilla.org/zh-CN/docs/Learn/JavaScript/First_steps/Math)
- Number (http://www.w3school.com.cn/jsref/jsref_obj_number.asp)
- Math (http://www.w3school.com.cn/jsref/jsref_obj_math.asp)

编码

首先练习数字相关的一些操作：

```
<div>
    <label>Number A:<input id="radio-a" type="radio" name="math-obj" value="a"></
label><input id="num-a" type="text">
    <label>Number B:<input id="radio-b" type="radio" name="math-obj" value="b"></
label><input id="num-b" type="text">
</div>
<div>
    <button>判断当前选中的输入框输入内容是否为数字</button>
    <button>把 A 四舍五入为 B 个小数位数的数字</button>
    <button>当前选中数字的绝对值</button>
    <button>对当前选中的数字进行上舍入</button>
    <button>对当前选中的数字进行下舍入</button>
    <button>把当前选中的数字四舍五入为最接近的整数</button>
    <button>返回 A 和 B 中的最高值</button>
    <button>返回 A 和 B 中的最低值</button>
</div>
<p id="result"></p>
```

基于如上HTML，实现需求

- 按照HTML中按钮的描述以此实现功能
- 计算结果显示在 id 为 result 的 P 标签中
- 除了第一个按钮，其它按钮操作时，都需要判断输入是否为数字，否则在 console 中输出错误信息

阅读

- W3School 字符串 (http://www.w3school.com.cn/js/js_obj_string.asp)
- W3School 字符串 (http://www.w3school.com.cn/jsref/jsref_obj_string.asp)
- MDN JavaScript中的字符串 (https://developer.mozilla.org/zh-CN/docs/Learn/JavaScript/First_steps/Strings)
- MDN 有用的字符串方法 (https://developer.mozilla.org/zh-CN/docs/Learn/JavaScript/First_steps/Useful_string_methods)

编码

```
<div>
    <label>String A:
        <input id="radio-a" type="radio" checked="true" name="str-obj" value="a">
    </label>
    <textarea id="str-a"></textarea>
    <label>String B:
        <input id="radio-b" type="radio" name="str-obj" value="b">
    </label>
    <textarea id="str-b"></textarea>
    <label>Num A: <input id="num-a" type="number" value="0"></label>
    <label>Num B: <input id="num-b" type="number" value="1"></label>
</div>
<div>
    <button>获取当前选中输入的内容长度</button>
    <button>当前选中输入中的第3个字符</button>
    <button>把两个输入框的文字连接在一起输出（concat）</button>
    <button>输入B中的内容，在输入A的内容中第一次出现的位置（indexOf）</button>
    <button>输入A中的内容，在输入B的内容中最后一次出现的位置（lastIndexOf）</button>
    <button>使用slice获取选中输入框内容的部分内容，参数为num-a及num-b</button>
    <button>当前选中输入框的行数</button>
    <button>使用substr获取选中输入框内容的子字符串，参数为num-a及num-b</button>
    <button>把所选输入框中的内容全部转为大写</button>
    <button>把所选输入框中的内容全部转为小写</button>
    <button>把所选输入框中内容的半角空格全部去除</button>
    <button>把所选输入框中内容的a全部替换成另外一个输入框中的内容</button>
</div>
<p id="result"></p>
```

基于如上HTML，实现需求

- 按照HTML中按钮的描述以此实现功能
- 计算结果显示在 id 为 result 的 P 标签中

编码

```
/*
实现一个字符串头尾去除空格的函数
注意需要去除的空格，包括全角、半角空格
暂时不需要学习和使用正则表达式的方式
*/
function diyTrim(str) {
    var result = "";

    // do something

    return result
}

// 测试用例
console.log(diyTrim(' a f b    ')); // ->a f b
console.log(diyTrim('    ffdaf    ')); // ->ffdaf
console.log(diyTrim('1    ')); // ->1
console.log(diyTrim('   f')); // ->f
console.log(diyTrim('      a f b      ')); // ->a f b
console.log(diyTrim(' ')); // ->
console.log(diyTrim('  ')); // ->
console.log(diyTrim('')); // ->

/*
去掉字符串str中，连续重复的地方
*/
function removeRepetition(str) {
    var result = "";

    // do something

    return result;
}

// 测试用例
console.log(removeRepetition("aaa")); // ->a
console.log(removeRepetition("abbba")); // ->aba
console.log(removeRepetition("aabbaabb")); // ->abab
console.log(removeRepetition("")); // ->
console.log(removeRepetition("abc")); // ->abc
```

如以上代码，分别实现 diyTrim 及 removeRepetition 函数，并跑通代码中的测试用例。

阅读

- W3School 对象 (http://www.w3school.com.cn/js/js_objects.asp)
- MDN JavaScript 对象基础 (https://developer.mozilla.org/zh-CN/docs/Learn/JavaScript/Objects/Basics)

编码

```
var tree = {
    "id": 0,
    "name": "root",
    "left": {
        "id": 1,
        "name": "Simon",
        "left": {
            "id": 3,
            "name": "Carl",
            "left": {
                "id": 7,
                "name": "Lee",
                "left": {
                    "id": 11,
                    "name": "Fate"
                }
            },
            "right": {
                "id": 8,
                "name": "Annie",
                "left": {
                    "id": 12,
                    "name": "Saber"
                }
            }
        },
        "right": {
            "id": 4,
            "name": "Tony",
            "left": {
                "id": 9,
                "name": "Candy"
            }
        }
    },
    "right": {
        "id": 2,
        "name": "right",
        "left": {
            "id": 5,
            "name": "Carl",
        },
        "right": {
            "id": 6,
            "name": "Carl",
            "right": {
                "id": 10,
                "name": "Kai"
            }
        }
    }
}

// 假设id和name均不会重复，根据输入name找到对应的id
function findIdByName(name) {

}
```

```
// 假设id和name均不会重复，根据输入id找到对应的name
function findNameById(id) {

}

// 把这个对象中所有的名字以"前序遍历"的方式全部输出到console中
function getListWithDLR() {

}

// 把这个对象中所有的名字以"中序遍历"的方式全部输出到console中
function getListWithLDR() {

}

// 把这个对象中所有的名字以"后序遍历"的方式全部输出到console中
function getListWithLRD() {

}
```

有如上对象，分别实现代码下方的几个函数，满足以下需求：

- 假设id和name均不会重复，根据输入name找到对应的id
- 假设id和name均不会重复，根据输入id找到对应的name
- 把这个对象中所有的名字以“前序遍历”的方式全部输出到console中
- 把这个对象中所有的名字以“中序遍历”的方式全部输出到console中
- 把这个对象中所有的名字以“后序遍历”的方式全部输出到console中

阅读

接下来我们学习一个非常有用的数据结构：数组

- W3School 数组 (http://www.w3school.com.cn/js/js_obj_array.asp)
- W3School 数组参考 (http://www.w3school.com.cn/jsref/jsref_obj_array.asp)
- MDN 数组 (https://developer.mozilla.org/zh-CN/docs/Learn/JavaScript/First_steps/Arrays)
- 队列 (https://baike.baidu.com/item/%E9%98%9F%E5%88%97/14580481?fr=aladdin)
- 栈 (https://baike.baidu.com/item/%E6%A0%88/12808149)

编码

练习如何使用数组来实现队列，综合考虑使用数组的 push，pop，shift，unshift操作

```
<input id="queue-input" type="text">
<p id="queue-cont">队列内容：apple-&gt;pear</p>
<button id="in-btn">入队</button>
<button id="out-btn">出队</button>
<button id="font-btn">打印队头元素内容</button>
<button id="empty-btn">判断队列是否为空</button>

<script>

var queue = ["apple", "pear"];

</script>
```

基于以上代码，实现如按钮中描述的功能：

- 实现如阅读材料中，队列的相关入队、出队、获取队头、判空的操作
- 队头对应数组中最后一个元素
- 入队和出队操作后，需要在 id 为 queue-cont 的 p 标签中更新显示队列中的内容，队头在最右侧，中间用 -> 连接（练习使用数组的join方法）

编码

对上面练习稍作小调整：

```
<input id="queue-input" type="text">
<p id="queue-cont">队列内容：apple&lt;-pear</p>
<button id="in-btn">入队</button>
<button id="out-btn">出队</button>
<button id="font-btn">打印队头元素内容</button>
<button id="empty-btn">判断队列是否为空</button>

<script>

var queue = ["apple", "pear"];

</script>
```

基于以上代码，实现如按钮中描述的功能：

- 实现如阅读材料中，队列的相关入队、出队、获取队头、判空的操作
-
  - 队头对应数组中第一个元素
- 入队和出队操作后，需要在 id 为 queue-cont 的 p 标签中更新显示队列中的内容，队头在最左侧，中间用 <- 连接（练习使用数组的join方法）

编码

练习如何使用数组来实现栈，综合考虑使用数组的 push，pop，shift，unshift操作

```
<input id="stack-input" type="text">
<p id="stack-cont">栈内容：apple-gt;pear</p>
<button id="push-btn">进栈</button>
<button id="pop-btn">退栈</button>
<button id="font-btn">打印栈顶元素内容</button>
<button id="empty-btn">判断栈是否为空</button>

<script>

var stack = ["apple", "pear"];

</script>
```

基于以上代码，实现如按钮中描述的功能：

- 实现如阅读材料中，队列的相关进栈、退栈、获取栈顶、判空的操作
- 栈顶对应数组中最后一个元素
- 进栈和退栈操作后，需要在 id 为 stack-cont 的 p 标签中更新显示栈中的内容，栈顶在最右侧，中间用 -> 连接（练习使用数组的join方法）

编码

对上面练习进行小调整

```
<input id="stack-input" type="text">
<p id="stack-cont">栈内容：applelt;-pear</p>
<button id="push-btn">进栈</button>
<button id="pop-btn">退栈</button>
<button id="font-btn">打印栈顶元素内容</button>
<button id="empty-btn">判断栈是否为空</button>

<script>

var stack = ["apple", "pear"];

</script>
```

基于以上代码，实现如按钮中描述的功能：

- 实现如阅读材料中，队列的相关进栈、退栈、获取栈顶、判空的操作
- 栈顶对应数组中第一个元素
- 进栈和退栈操作后，需要在 id 为 stack-cont 的 p 标签中更新显示栈中的内容，栈顶在最左侧，中间用 -< 连接（练习使用数组的join方法）

阅读

- MDN 排序 (https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Reference/Global_Objects/Array/sort)

编码

```
var arr = [43, 54, 4, -4, 84, 100, 58, 27, 140];
```

将上面数组分别按从大到小以及从小到大进行排序后在console中输出

```
var arr = ['apple', 'dog', 'cat', 'car', 'zoo', 'orange', 'airplane'];
```

将上面数组分别按字母顺序a-z及z-a进行排序，在console中输出

```
var arr = [[10, 14], [16, 60], [7, 44], [26, 35], [22, 63]];
```

将上面的二维数组，按照每个元素中第二个数从大到小的顺序进行排序输出，输出结果应该为：

```
[[22, 63], [16, 60], [7, 44], [26, 35], [10, 14]]
```

```
var arr = [
    {
        id: 1,
        name: 'candy',
        value: 40
    }, {
        id: 2,
        name: 'Simon',
        value: 50
    }, {
        id: 3,
        name: 'Tony',
        value: 45
    }, {
        id: 4,
        name: 'Annie',
        value: 60
    }
];
```

将上面数组分别按元素对象的value值从小到大进行排序后输出

编码

学习通用的数据用不同的数据结构进行存储，以及相互的转换

对象转为数组：

```
var scoreObject = {
    "Tony": {
        "Math": 95,
        "English": 79,
        "Music": 68
    },
    "Simon": {
        "Math": 100,
        "English": 95,
        "Music": 98
    },
    "Annie": {
        "Math": 54,
        "English": 65,
        "Music": 88
    }
}
```

如上有一个用来存储学习成绩的对象，编写一个函数，将其转为如下的二维数组

```
var scoreArray = [
    ["Tony", 95, 79, 68],
    ……
];
```

数组转为对象：

```
var menuArr = [
    [1, "Area1", -1],
    [2, "Area2", -1],
    [3, "Area1-1", 1],
    [4, "Area1-2", 1],
    [5, "Area2-1", 2],
    [6, "Area2-2", 2],
    [7, "Area1-2-3", 4],
    [8, "Area2-2-1", 6],
];
```

如上有一个用来存储多级菜单数据的数组，编写一个函数，将其转为如下的对象

```
var menuObject = {
    "1": {
        name: "Area1",
        subMenu: {
            "3": {
                name: "Area1-1"
            },
            "4": {
                name: "Area1-2",
                subMenu: {
                    "7": {
                        name: "Area1-2-3"
                    }
                }
            }
        }
    }

    ……

}
```

进阶任务

如果你很快就完成上面的任务，可以去LeetCode上去多进行一些练习。

提交

把你今天觉得做得最好的代码放在Github后进行提交

总结

依然把今天的学习用时，收获，问题进行记录

下一个任务预告

下一个任务将继续学习 JavaScript，我们将接触更多的 JavaScript 对象

课程优秀学习笔记

暂无优秀学习笔记~