二阶段需要的 js 内容

一、DOM

1.DOM 的简述

DOM 全称 Document Object Model，即文档对象模型，它允许脚本(js)控制 Web 页面、窗口和文档。

简单说就是浏览器窗口内的所有元素。

2. DOM 基本功能

- 查询某个元素
- 查询某个元素的祖先、兄弟以及后代元素
- 获取、修改元素的属性
- 获取、修改元素的内容
- 创建、插入和删除元素

简单说就是对页面元素的增删改查。

二、元素的操作（应用部分）

1.通过元素 id 获取到该元素元素

`document.getElementById()`在文档中按照 id 属性查找元素参数写 id 名。

```js
//获取到 html 元素中 id 叫做 myDiv 的元素整体
var myDiv = document.getElementById('myDiv');
```

2.事件——与元素交互

可以在 DOM 元素上绑定`onclick、onmouseover、onmouseout、onmousedown、onmouseup、ondblclick、onkeydown、onkeyup`等。JavaScript 能够在事件发生时执行。

（1）在 DOM 中直接绑定事件

直接在标签中设置事件属性，绑定方法，从而完成交互。

```html
<input type="button" value="点我" onclick="hello(1)" />
<script>
    function hello(a) {
        alert('hello world!');
        // alert(a);
    }
</script>
```

（2）在 JavaScript 代码中绑定事件

可以通过 js 获取元素，在 js 中绑定获取的元素相关的事件，调用函数处理事件功能。

```js
var btn = document.getElementById('mybtn');
btn.onclick = function () {
    alert('hello');
  //其他语句，包括调用函数语句如下：
  //abc()
};
```

2. 获取/设置元素的内容 innerHTML

Element.innerHTML 获取/设置元素的内容。

innerHTML 在 JS 是双向功能：获取对象的内容或向对象插入内容。

```html
<div id="myDiv">div 元素</div>
<script>
    var myDiv = document.getElementById('myDiv');
    myDiv.innerHTML = '<p>一个元素</p>';
</script>
```

- 可以通过`document.getElementById("myDiv").innerHTML`来获取`id`为

`myDiv`的对象的内嵌内容;

-也可以对某对象插入内容,如`document.getElementById("myDiv").innerHTML='这是被插入的内容';`这样就能向 id 为 myDiv 的对象插入内容。

3. value 值的获取

元素的 value 属性可以直接获取到具有 value 属性元素的的值。多数情况用在获取表单相关元素中。

```html
<input type="text" id="nn" />
<input type="button" value="获取" onclick="print()" />
<script>
    var a = document.getElementById('nn');
    //通过事件获取值
    function print() {alert(a.value);}
</script>
```

练习

> 在页面上放入一个文本输入框、一个密码输入框、一个按钮

> 使用 js 获取两个输入框元素和一个按钮元素

> 点击按钮元素触发点击事件，获得文本输入框和密码输入框的内容，并在控制台查

看

> 创建两个 p 标签，使用 js 获取两个 p 元素

> 将获取的用户输入的用户名和密码显示页面上

4. 清空多余空格-trim()方法

- `trim()`方法用于删除字符串的"头尾"空白符，空白符包括：空格、制表符 tab、换行符等其他空白符等。
- `trim()`方法不会改变原始字符串。
- `trim()`方法不适用于 null, undefined, Number 类型。

```js
var str = '    hello     ';
alert(str.trim());
```

非空验证的方法

```js
var str1 = '     ';
var str2 = ''
var str3 = ' tom ';
function notEmpty(s){
   var nstr = s.trim();
```

```
    return nstr.length;
}
notEmpty(str1)
notEmpty(str2)
notEmpty(str3)
```

三、JSON

1. JSON 简述

JSON 是一种轻量级的数据交换格式。采用完全独立于编程语言的文本格式来存储和表示数据。

2. JSON 语法规则

JSON 是一个序列化的对象或数组。属性名（key）用双引号包裹，值可以是对象、数组、数字、字符串或者三个字面值(false、null、true)中的一个。值中的字面值中的英文必须使用小写。

（1）对象结构

```json
{
    "key1": "value1",
    "key2": "value2"
}
```

```

（2）数组结构

```json
[
    {
        "key1": "value1",
        "key2": "value2"
    },
    {
        "key3": "value3",
        "key4": "value4"
    }
]
```

2. JSON 和 JS 对象的关系

JSON 是 JS 的一种简单数据格式，JSON 是 JavaScript 原生格式，它是一种严格的 JS 对象的格式，JSON 的属性名必须有双引号，如果值是字符串，也必须是双引号。

区别	JSON	Javascript
含义	仅仅是一种数据格式	表示类的实例
传输	可以跨平台数据传输，速度快	不能传输
表现	1.键值对方式，键必须加双引号	1.键值对方式，键不加引号
	2.值不能是方法函数，不能是undefined/NaN	2.值可以是函数、对象、字符串、数字、boolean 等
相互转换	Json转换Js对象	js对象转换Json
	1.JSON.parse(JsonStr);(不兼容IE7)	JSON.stringify(jsObj);
	2.eval("("+jsonStr+")");(兼容所有浏览器，但不安全，会执行json里面的表达式)	
其他	调用JSON官网的JS,实现parse和stringify在各个浏览器的兼容:	
```

其中，eval()函数是自行参数内，字符串的 js 代码，如:eval("2+2")

2. JSON 字符串和 JSON 对象的区别

JSON 字符串：指的是符合 JSON 格式要求的 JS 字符串。

需要注意的是，服务器的返回数据，以及前台向后台的传输数据都可能是 JSON 字符串，而不是 JSON 对象。

```js
var jsonStr = "{id:'001',name:'tom',age:18}";
```

JSON 对象：指符合 JSON 格式要求的 JS 对象。

```js
var jsonObj = { id: '001', name: 'tom', age: 18 };
```

3. 转换方式

为什么要使用 JSON 的转化，因为前后台数据的传输均为 JSON 字符串格式，如不通过解析和转化将无法使用。

`JSON.stringify()`用于将一个值转为字符串。该字符串应该符合 JSON 格式，并且可以被`JSON.parse`方法还原。

```js
var json = JSON.stringify({ a: 'Hello', b: 'World' });
//结果是 '{"a": "Hello", "b": "World"}'
```

`JSON.parse()` 用于将 JSON 字符串转化成对象。

```js
var obj = JSON.parse('{"a": "Hello", "b": "World"}');
//结果是 {a: 'Hello', b: 'World'}
```