HTML

1.系统结构

```
B/S架构：（以后主要走的方向是这个。）
    Browser/Server （浏览器/服务器的交互形式。）
    Browser(浏览器)支持哪些语言：HTML CSS JavaScript
    写HTML CSS JavaScript代码的这波人职位叫做：WEB前端开发工程师。（Java程序员目前来看也需要会一些前端的东西。）
    前端页面上的图片需要UI设计师完成。（PS对java程序员来说没有太高的要求。）
    S是服务器端Server，Server端的语言很多：C C++ Java python.....（我们主要是使用Java语言完成服务器端的开发）
    B/S架构的系统有什么优点和缺点？
        优点：升级方便，只升级服务器端代码即可。维护成本低。
        缺点：速度慢、体验不好、界面不炫酷

    企业内部的解决方案都是采用B/S架构的系统，因为企业内部办公需要的一些系统
    不需要炫酷，不需要特别好的用户体验，只要能做数据的增删改查即可。并且企业
    内部更注重维护的成本。

    B/S架构的系统有哪些代表？
        京东、百度、天猫....
```

```
C/S架构
    Client/Server      （客户端/服务器端的交互形式。）
    缺点：升级麻烦，维护成本较高。
    优点：速度快，体验好，界面炫酷。（娱乐型的系统多数是C/S架构的。）

    常见的C/S架构的系统：
        QQ、微信、支付宝、魔兽、Dota2....
```

2.HTML概述

```
1、什么是HTML？
HTML: Hyper Text Markup Language （超文本标记语言）
    由大量的标签组成，每一个标签都有开始标签和结束标签。
    <标签>
        <标签>
            <标签 属性名="属性值" 属性名="属性值">
            </标签>
        </标签>
    </标签>

    超文本: 流媒体、图片、声音、视频....
```

```
2、怎么开发HTML？
HTML开发的时候使用普通的文本编辑器就行，创建的文件扩展名是.html或者.htm
    HTML也有专业的开发工具，例如：DreamWeaver、HBuilder.....
```

```
3、怎么运行HTML？
直接采用浏览器打开HTML文件就是运行。
```

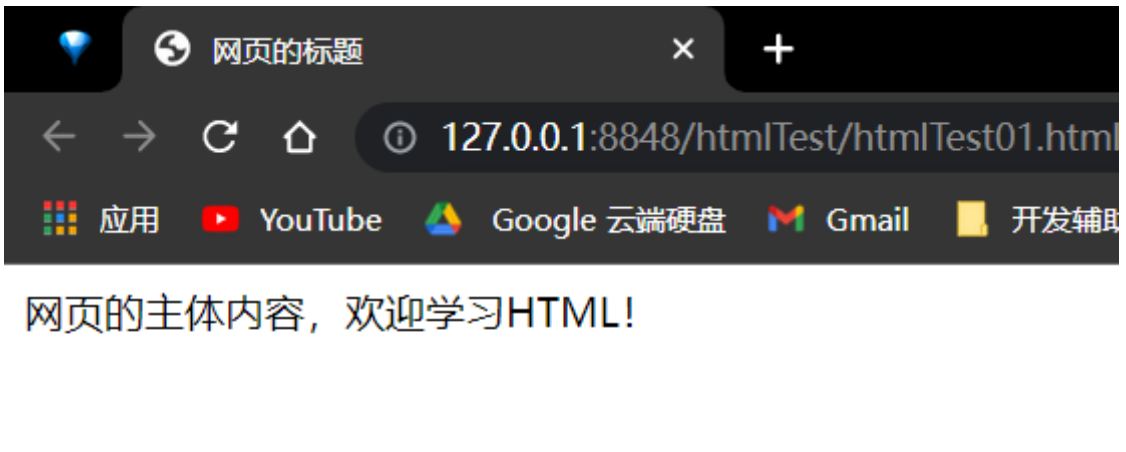
```
4、HTML是谁制定的？
W3C：世界万维网联盟
W3C制定了HTML的规范，每个浏览器生产厂家都会遵守规范。HTML程序员也会按照这个规范去写代码。
HTML规范目前最高的版本是：HTML5.0，简称H5.
我们这里学习HTML4.0（主要是学习一下HTML的基础用法。）

W3C制定了很多规范：
    HTML/XML/http协议/https安全协议......

为了方便中国web前端程序员的开发，提供大量的帮助文档。为开发提供方便。
    w3school:先出现的，和W3C没有关系
    w3cschool：后出现的，和W3C没有关系
```

3.HTML代码及页面展示

3.1 第一个HTML文件

```
<!--
    1、这是HTML的注释
    2、加上以下代码的第一行就表示HTML5语法。去掉就表示HTML4.0
    3、HTML不区分大小写，语法松散不严格。
-->
<!DOCTYPE html>
<!--根-->
```

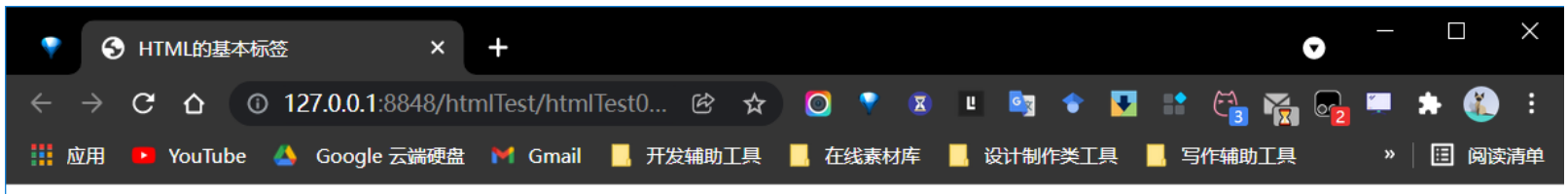
```html
<html>

    <!--头-->
    <head>
        <meta charset="utf-8">
        <!--网页标题，显示在网页左上角-->
        <title>网页的标题</title>
    </head>

    <!--体-->
    <body>
        网页的主体内容，欢迎学习HTML！
    </body>

</html>
```



3.2 HTML的基本标签

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>HTML的基本标签</title>
    </head>
    <body>
        <!--段落标记-->
        <p>
            《黛玉葬花》是文学名著《红楼梦》中的经典片段。林黛玉最怜惜花，觉得花落以后埋在土里最干净，说明她对美有独特的见解。她写了葬花词，以花比喻自己，在《红楼梦》中是最美丽的诗歌之一。贾宝玉和林黛玉在葬花的时候有一段对话，成为《红楼梦》中一场情人之间解除误会的绝唱。
        </p>
        <p>
            《黛玉葬花》是文学名著《红楼梦》中的经典片段。林黛玉最怜惜花，觉得花落以后埋在土里最干净，说明她对美有独特的见解。她写了葬花词，以花比喻自己，在《红楼梦》中是最美丽的诗歌之一。贾宝玉和林黛玉在葬花的时候有一段对话，成为《红楼梦》中一场情人之间解除误会的绝唱。
        </p>

        <!--标题字：是HTML预留的格式，和word中的标题字相同-->
        <h1>一级标题</h1>
        <h2>二级标题</h2>
        <h3>三级标题</h3>
        <h4>四级标题</h4>
        <h5>五级标题</h5>
        <h6>六级标题</h6>

        <!--换行标记，br标签是一个独目标记-->
        hello
        world!
        hello <br>world!

        <!--水平线，独目标记-->
        <hr>
        <!--color和width都是hr标签的属性-->
        <hr color="red" width="50%">
        <!--语法太松散了。-->
        <hr color='green' width=30%>

        <!--预留格式 把格式留住-->
        <pre>
        for(int i = 0; i < 10; i++){
            System.out.println("i = " + i);
        }
        </pre>

        <del>删除字</del>
        <ins>插入字</ins>
```

```
        <b>粗体字</b>
        <i>斜体字</i>

        10<sup>2</sup><!--右上角加字-->

        10<sub>m</sub><!--右下角加字-->

        <!--font标签-->
        <font color="red" size="50">字体标签</font>
    </body>
</html>
```



《黛玉葬花》是文学名著《红楼梦》中的经典片段。林黛玉最怜惜花，觉得花落以后埋在土里最干净，说明 她对美有独特的见解。她写了葬花词，以花比喻自己，在《红楼梦》中是最美丽的诗歌之一。贾宝玉和林黛 玉在葬花的时候有一段对话，成为《红楼梦》中一场情人之间解除误会的绝唱。

《黛玉葬花》是文学名著《红楼梦》中的经典片段。林黛玉最怜惜花，觉得花落以后埋在土里最干净，说明 她对美有独特的见解。她写了葬花词，以花比喻自己，在《红楼梦》中是最美丽的诗歌之一。贾宝玉和林黛 玉在葬花的时候有一段对话，成为《红楼梦》中一场情人之间解除误会的绝唱。

一级标题

二级标题

三级标题

四级标题

五级标题

六级标题

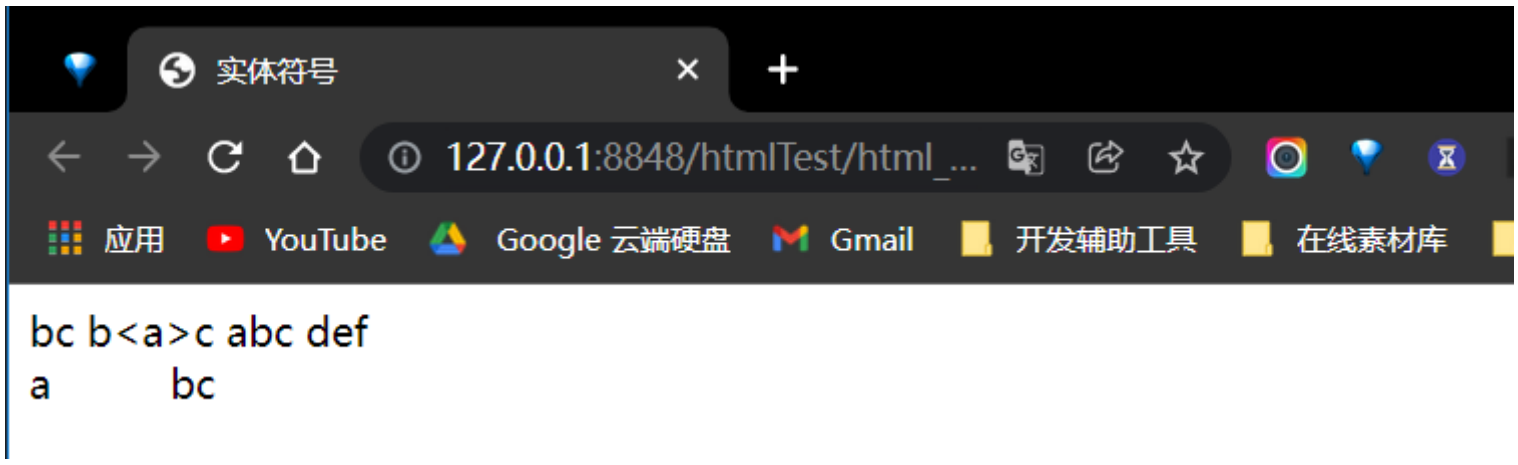
hello world! hello world!

```
for(int i = 0; i < 10; i++){
        System.out.println("i = " + i);
}
```

~~删除字~~ 插入字 **粗体字** *斜体字* 10^2 10_m 字体标签

3.3 HTML的实体符号

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>实体符号</title>
    </head>
    <body>
        b<a>c

        <!-- 实体符号特点是：以‘&’开始，以‘;’结束。'&lt;' 是小于号,'&gt;'是大于号 -->
        b&lt;a&gt;c

        <!--加空格     是空格-->
        abc                                                     def<br>
        a          bc
    </body>
</html>
```

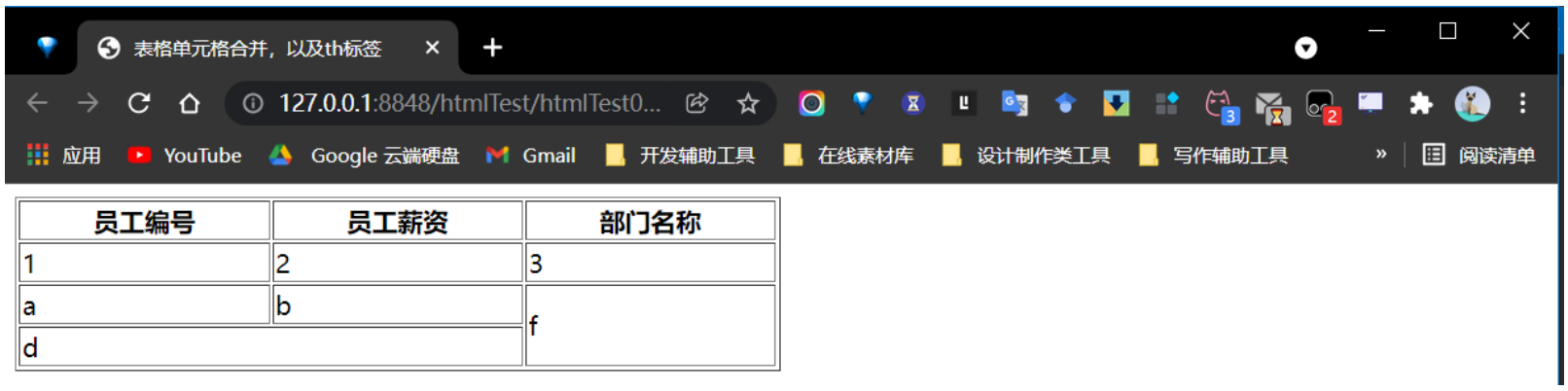


3.4 HTML的表格*

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>表格</title>
    </head>
    <body>
        <br><br><br><br><br><br><br><br><br>
        <center><h1>员工信息列表</h1></center>
        <hr>
        <!--
            border="1px" 设置表格的边框为1像素宽度。
            width 宽度
            height 高度
        -->
        <!--
        <table border="1px" width="300px">
        -->
        <!--表格-->
        <table align="center" border="1px" width="60%" height="150px">
            <!--align对齐方式-->
            <tr align="center"><!--行-->
                <td>a</td><!--单元格-->
                <td>b</td>
                <td>c</td>
            </tr>
            <tr>
                <td>d</td>
                <td>e</td>
                <td>f</td>
            </tr>
            <tr>
                <td>x</td>
                <td>y</td>
                <td align="center">z</td>
            </tr>
        </table>
    </body>
</html>
```

3.5 HTML的单元格合并

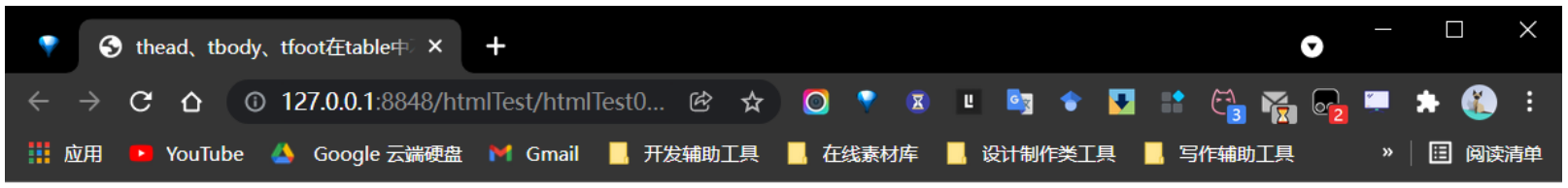
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>表格单元格合并，以及th标签</title>
    </head>
    <body>

        <!--注意事项：
            1、row合并的时候，删除“下面的”单元格
            2、col合并的时候，对删除哪个没有要求。
        -->
        <table border="1px" width="50%">
            <tr>
                <!--
                <td>员工编号</td>
                <td>员工薪资</td>
                <td>部门名称</td>
                -->
                <!-- th 标签也是单元格标签，比td多的是居中、加粗。-->
                <th>员工编号</th>
                <th>员工薪资</th>
                <th>部门名称</th>
            </tr>
            <tr>
                <td>1</td>
                <td>2</td>
                <td>3</td>
            </tr>
            <tr>
                <td>a</td>
                <td>b</td>
                <td rowspan="2">f</td><!--row合并-->
            </tr>
            <tr>
                <td colspan="2">d</td><!--col合并-->
                <!--
                <td>f</td>
                -->
            </tr>
        </table>
    </body>
</html>
```



3.6 thead、tbody、tfoot

```
<!DOCTYPE html>
<html>
    <head>
        <!--这行代码的作用是告诉浏览器采用哪一种字符集打开当前页面。
        注意：并不是设置当前页面的字符编码方式。-->
        <meta charset="gbk">
        <title>thead、tbody、tfoot在table中不是必须的，只是这样做便于后期的JS代码的编写。</title>
    </head>
    <body>
        <table border="1px" width="50%">
            <!--头-->
            <thead>
                <tr>
                    <th>员工编号</th>
                    <th>员工薪资</th>
                    <th>部门名称</th>
                </tr>
            </thead>

            <!--体-->
            <tbody>
                <tr>
                    <td>1</td>
```

```
                    <td>2</td>
                    <td>3</td>
                </tr>
                <tr>
                    <td>a</td>
                    <td>b</td>
                    <td rowspan="2">f</td>
                </tr>
                <tr>
                    <td colspan="2">d</td>
                </tr>
            </tbody>

            <!--脚-->
            <tfoot>
                <tr>
                    <td>1</td>
                    <td>2</td>
                    <td>3</td>
                </tr>
            </tfoot>
        </table>

    </body>
</html>
```



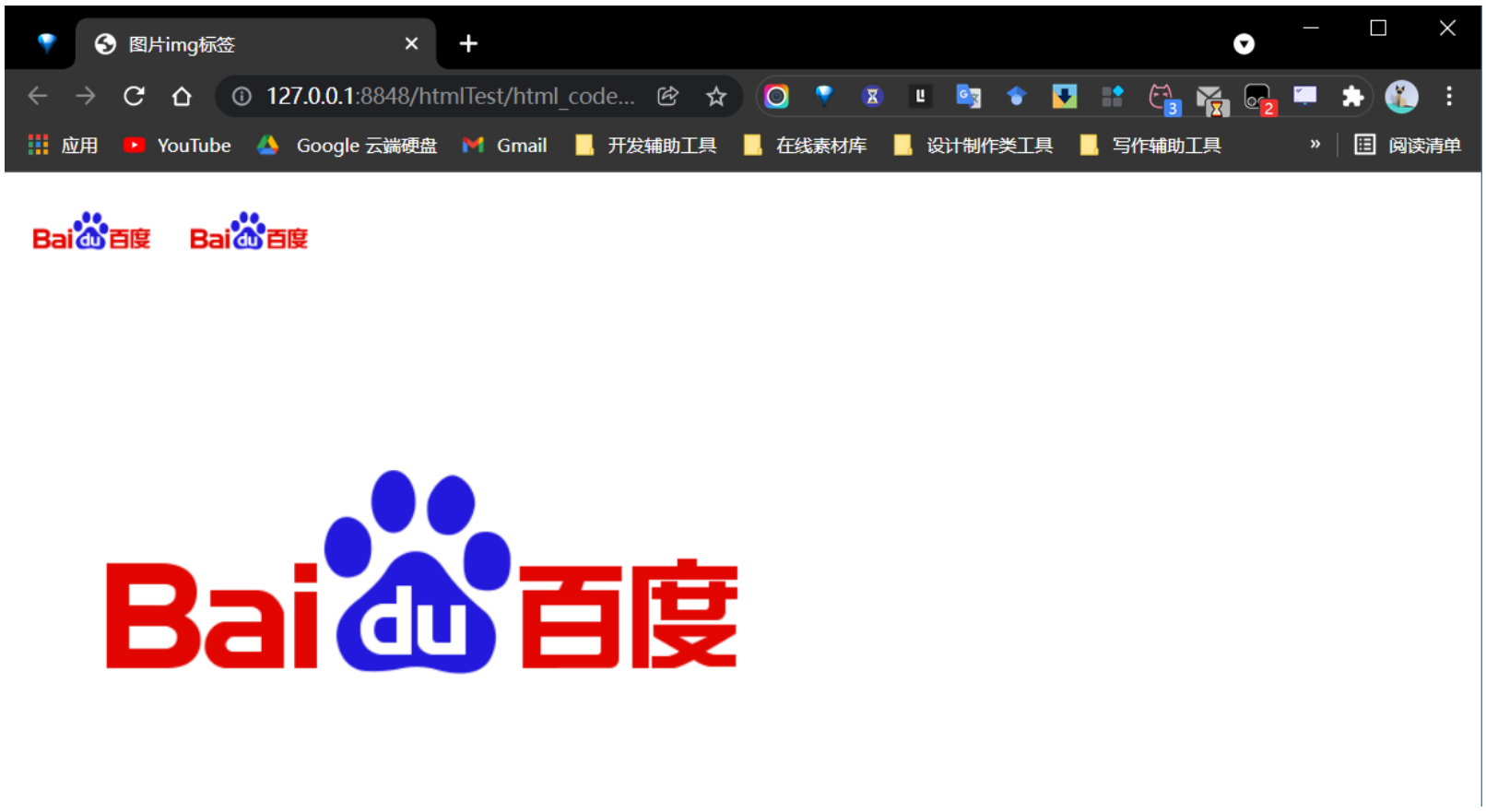
员工编号	员工薪资	部门名称
1	2	3
a	b	f
d		
1	2	3

3.7 背景颜色和背景图片

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>背景颜色和背景图片</title>
    </head>
    <!--
        bgcolor : 设置背景色
        background : 设置背景图片
        以上的设置都是对背景进行设置。
        背景图片会覆盖背景色
    -->
    <body bgcolor="red" background="img/bd_logo1.png">
    </body>
</html>
```

3.8 HTML图片img标签

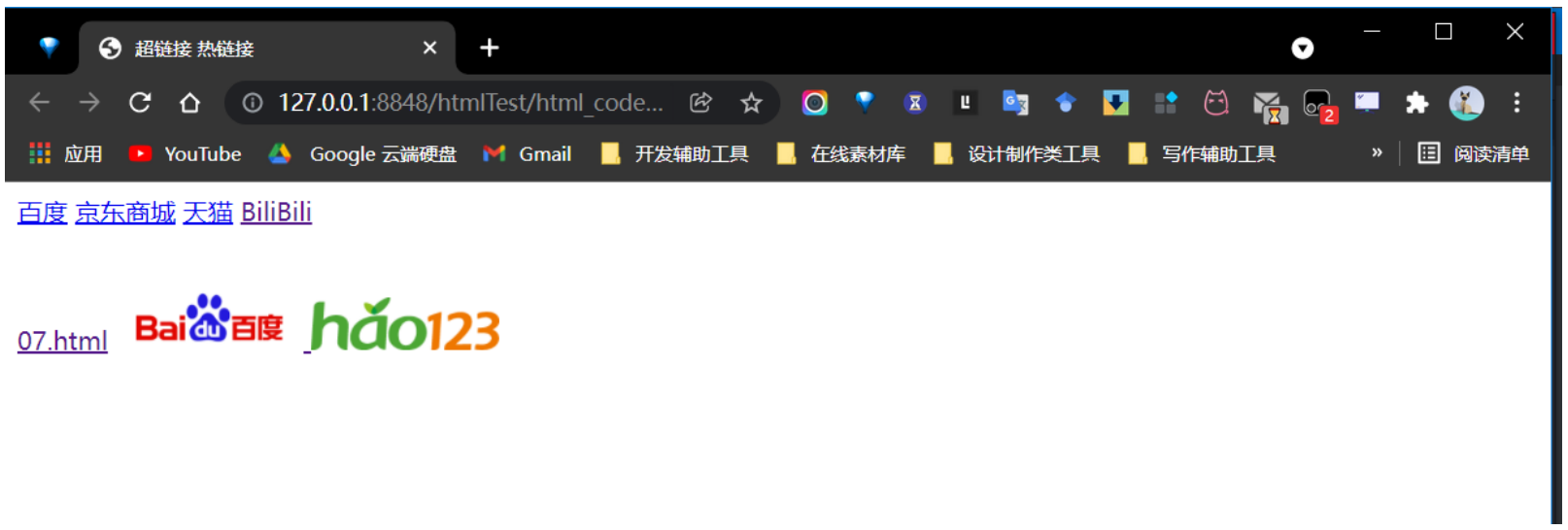
```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>图片img标签</title>
    </head>
    <body>
        <!--
            1、设置图片宽度和高度的时候，只设置宽度，高度会进行等比例缩放。
            2、img标签就是图片标签
            3、src属性是图片的路径
            4、width设置宽度,height设置高度
            5、title设置鼠标悬停时显示的信息。
            6、alt设置图片加载失败时显示的提示信息。
        -->
        <img src="img/bd_logo1.png" width="100px" title="我是百度图片哦" alt="图片找不到哦!"/>

        <img src="img/bd_logo1.png" width="100px" title="我是百度图片哦" alt="图片找不到哦!"></img>

        <br><br><br>

        <img src="img/bd_logo1.png" />
    </body>
</html>
```



3.9 HTML超链接、热链接*

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>超链接 热链接</title>
    </head>
    <body>
        <!--
            超链接的特点：
                有下划线
                鼠标停留在超链接上面显示小手形状。
                点击超链接之后还能跳转页面。
        -->
        <a href="http://www.baidu.com">百度</a>
        <a href="http://www.jd.com/">京东商城</a>
        <a href="http://www.tmall.com/">天猫</a>
        <a href="http://www.bilibili.com/">BiliBili</a>

        <br><br>

        <!--
            href：hot references 热引用
            href属性后面一定是一个资源的地址。

            href后面的路径可以是绝对路径也可以是相对路径，可以是网络中某个资源的路径，也可以是本地资源的路径。
        -->
        <a href="07.html">07.html</a>

```

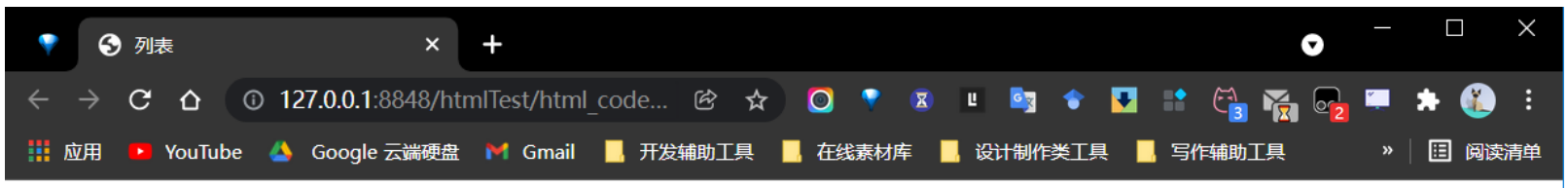
```
        <!--图片超链接-->
        <a href="https://www.baidu.com/">
            <img src="img/bd_logo1.png" width="120px"/>
        </a>

        <!--
            超链接有一个target属性：
                可取值：
                    _blank : 新窗口
                    _self :  当前窗口（默认就是这种方式。）
                    _top :  顶级窗口
                    _parent :  父窗口
        -->
        <a href="https://www.hao123.com/" target="_self">
            <img src="img/hao123.png" width="120px"/>
        </a>
    </body>
</html>

<!--
    超链接的作用：
        通过超链接可以从浏览器向服务器发送请求。
        浏览器向服务器发送数据（请求：request）
        服务器向浏览器发送数据（响应：response）

        B/S结构的系统：每一个请求都会对应一个响应。

    用户点击超链接和用户在浏览器地址栏上直接输入URL，有什么区别？
        本质上没有区别，都是向服务器发送请求。

    从操作上来讲，超链接使用更方便。
-->
```



3.10 HTML列表

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>列表</title>
    </head>
    <body>
        <!--有序列表-->
        <ol type="I">
            <li>水果
                <ol type="a">
                    <li>苹果</li>
                    <li>香蕉</li>
                    <li>橘子</li>
                </ol>
            </li>
            <li>蔬菜
                <ol type="a">
                    <li>西红柿</li>
                    <li>黄瓜</li>
                </ol>
            </li>
            <li>甜点</li>
        </ol>

        <hr>

        <!--无序列表-->
        <ul type="circle">
            <li>中国
                <ul type="square">
                    <li>南京
```

```
                        <ul type="disc">
                            <li>浦口区</li>
                            <li>玄武区</li>
                            <li>栖霞区</li>
                        </ul>
                    </li>
                    <li>盐城</li>
                    <li>无锡</li>
                </ul>
            </li>
            <li>美国</li>
            <li>日本</li>
        </ul>

    </body>
</html>
```



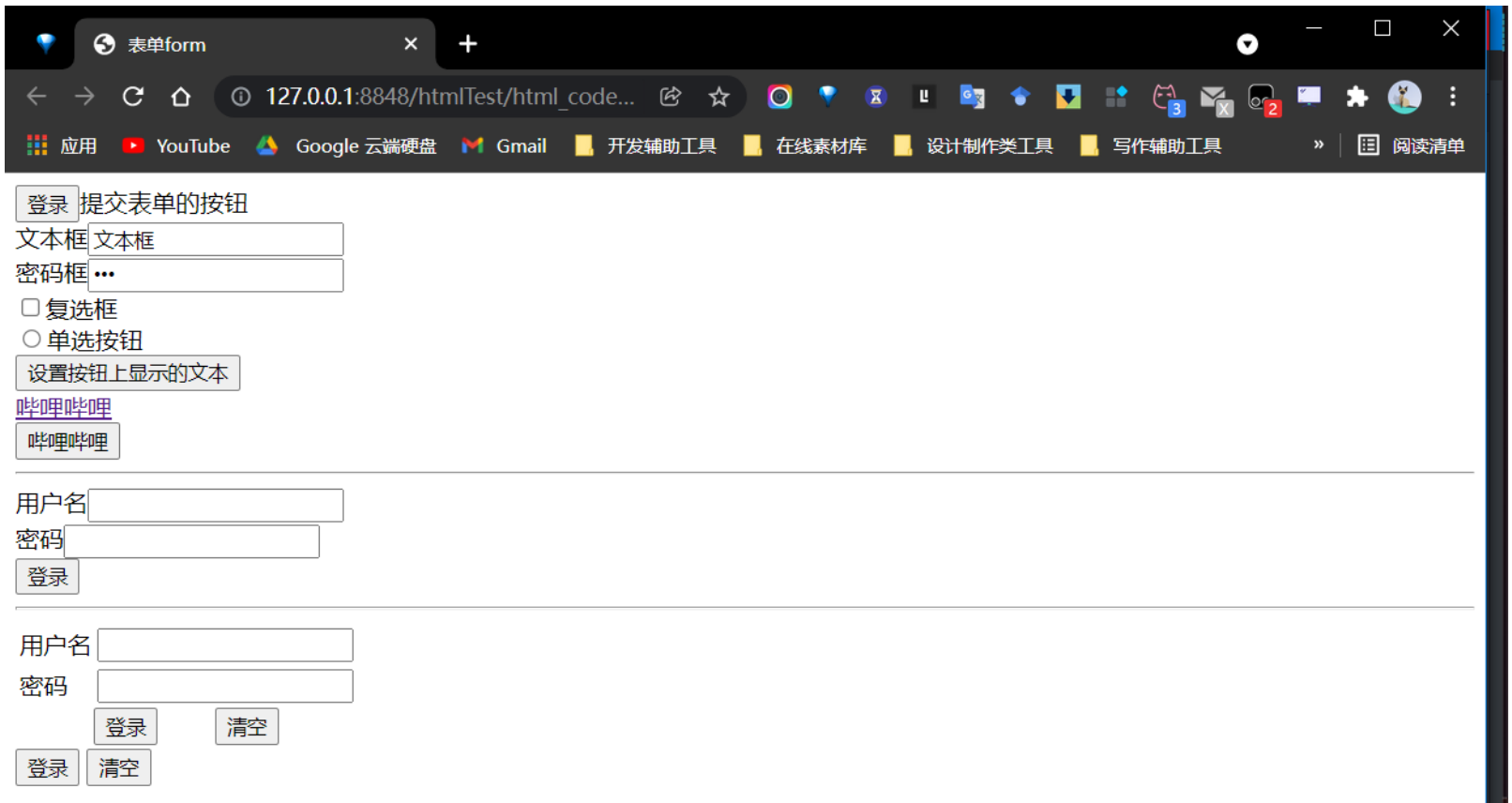
I. 水果
    a. 苹果
    b. 香蕉
    c. 橘子
II. 蔬菜
    a. 西红柿
    b. 黄瓜
III. 甜点

- 中国
  - 南京
    - 浦口区
    - 玄武区
    - 栖霞区
  - 盐城
  - 无锡
- 美国
- 日本

3.11 HTML表单*

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>表单form</title>
    </head>
    <body>
        <!--
            1、表单有什么用？
                收集用户信息。表单展现之后，用户填写表单，点击提交按钮提交数据给服务器。
            2、怎么画一个表单？
                使用form标签画表单。
            3、一个网页当中可以有多个表单form。
            4、表单最终是需要提交数据给服务器的，form标签有一个action属性，这个属性用来指定服务器地址：
                action属性用来指定数据提交给哪个服务器。
                action属性和超链接中的href属性一样。都可以向服务器发送请求（request）
            5、http://192.168.111.3:8080/oa/save 这是请求路径，表单提交数据最终提交给：
                192.168.111.3机器上的8080端口对应的软件。
        -->

        <form action="http://192.168.111.3:8080/oa/save">
            <!-- 画一个提交按钮，这个按钮可以提交表单-->
            <!-- 画按钮可以使用input输入域，type="submit"的时候表示该按钮是一个提交按钮，具有提交表单的能力。-->
            <!-- 对于按钮来说，按钮的value属性用来指定按钮上显示的文本信息。-->
            <input type="submit" value="登录"/>提交表单的按钮<br>
            文本框<input type="text" value="文本框"/><br>
            密码框<input type="password" value="密码框"/><br>
            <input type="checkbox"/>复选框<br>
            <input type="radio"/>单选按钮<br>

            <!--这是一个普通按钮，不具备提交表单的能力。-->
            <input type="button" value="设置按钮上显示的文本"/>
        </form>

        <!--超链接-->
        <a href="http://www.bilibili.com" target="_blank">哔哩哔哩</a>
```

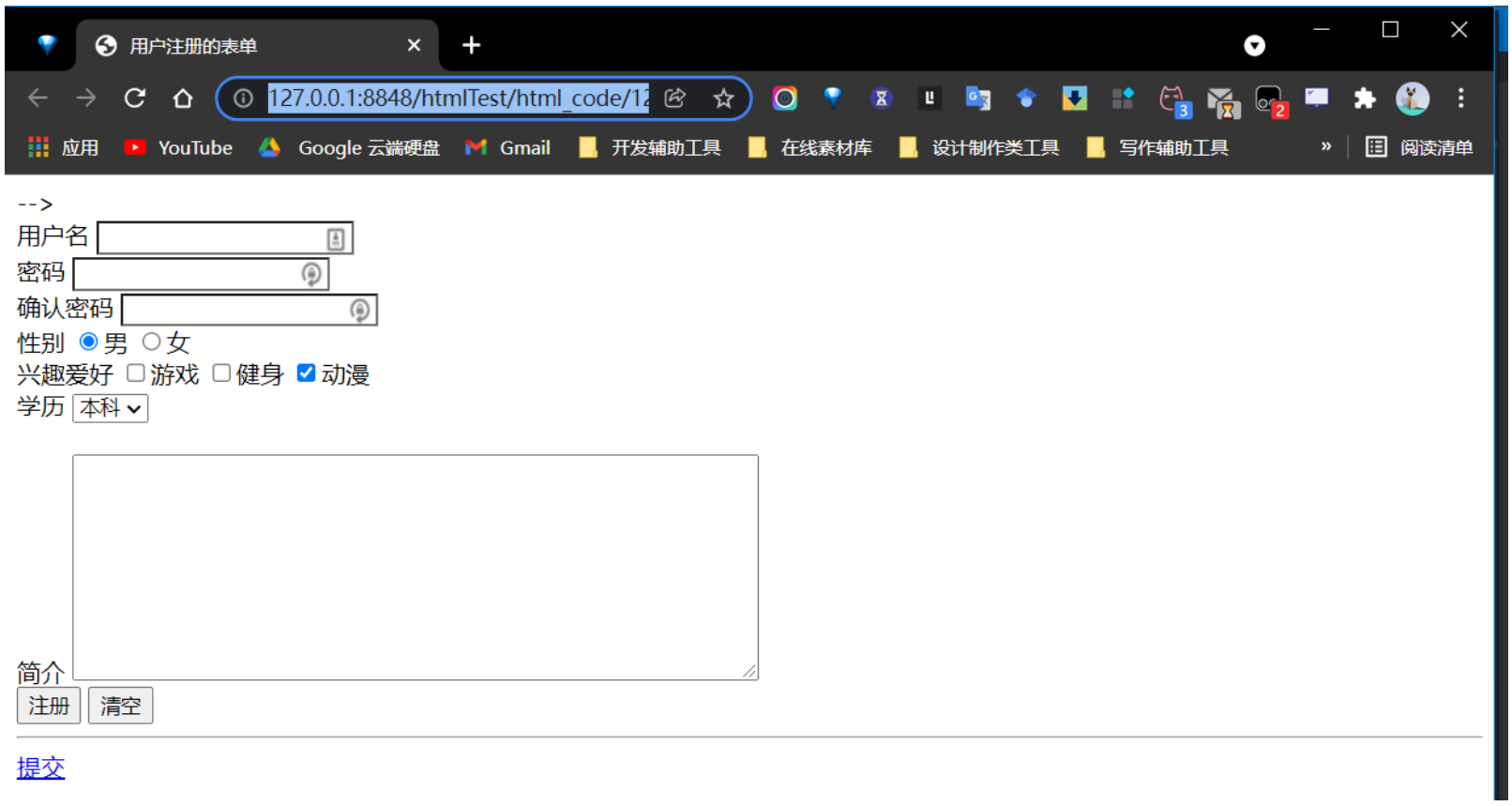
```
        <!--这个按钮和普通的超链接没什么太大的区别。（超链接和表单都可以向服务器发送请求，只不过表单发送请求的同时可以携带数据。）-->
        <form action="http://www.bilibili.com">
            <input type="submit" value="哔哩哔哩" />
        </form>

        <hr >

        <form action="http://localhost:8080/jd/login">
            用户名<input type="text" /><br>
            密码<input type="password" /><br>
            <input type="submit" value="登录" />
        </form>

        <!--
            表单是以什么格式提交数据给服务器的？
                http://localhost:8080/jd/login?username=abc&userpwd=111
                格式：action?name=value&name=value&name=value&name=value&name=value...
                W3C的HTTP协议规定的，必须以这种格式提交给服务器。

            重点强调：表单项写了name属性的，一律会提交给服务器。不想提交这一项，就不要写name属性。

            文本框和密码框的value不需要程序员指定，用户输入什么value就是什么。

            当name没有写的时候，该项不会提交给服务器。
            但是当value没有写的时候，value的默认值是空字符串""，会将空字符串提交给服务器。java代码得到的是：String username = "";
        -->
        <hr >
        <form action="http://localhost:8080/jd/login">
            <table>
                <tr>
                    <td>用户名</td>
                    <td><input type="text" name="username" /></td>
                </tr>
                <tr>
                    <td>密码</td>
                    <td><input type="password" name="userpwd" /></td>
                </tr>
                <tr align="center">
                    <td colspan="2">
                        <input type="submit" value="登录" />
                              
                        <input type="reset" value="清空" />
                    </td>
                </tr>
            </table>
        </form>

        <!--submit必须放到form标签内部-->
        <input type="submit" value="登录" />
        <!--必须放到form标签内部-->
        <input type="reset" value="清空" />
        <form></form>
    </body>
</html>
```



3.12 用户注册的表单

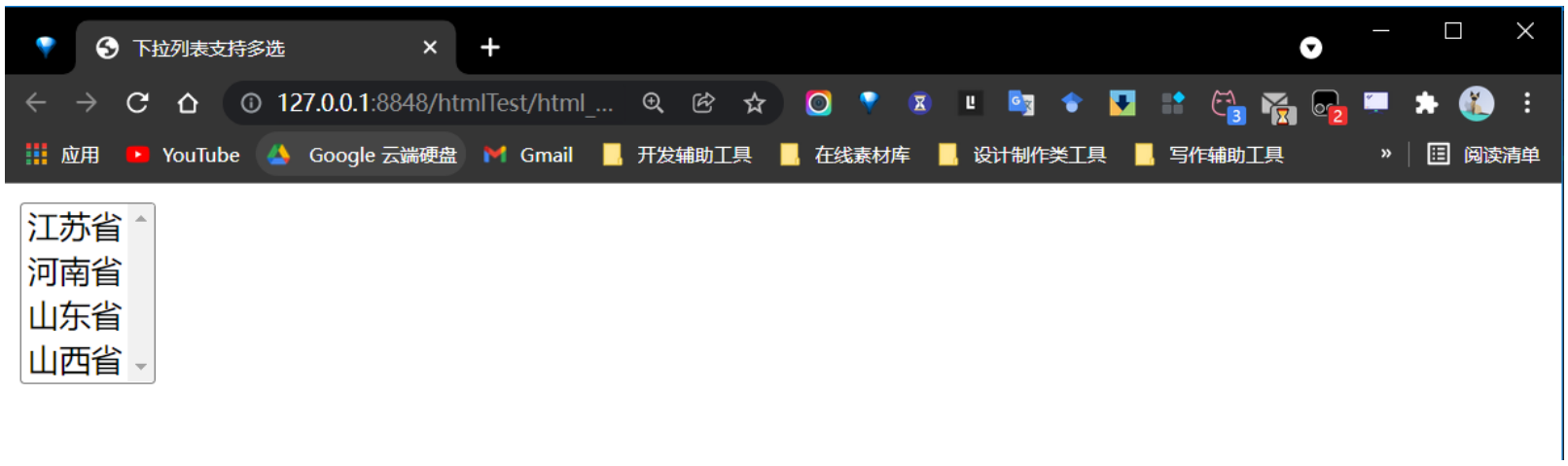
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>用户注册的表单</title>
    </head>
    <body>
        <!--
            用户注册：
                用户名
                姓名
                密码
                确认密码
                性别
                兴趣爱好
                学历
                简介

            form表单method属性：
                get:采用get方式提交的时候，用户提交的信息会显示在浏览器的地址栏上。
                post：采用post方式提交的时候，用户提交的信息不会显示在浏览器地址栏上。
                当用户提交的信息中含有敏感信息，例如：密码，建议采用post方式提交。

            method属性不指定，或者指定get，这种情况下都是get。
            只有当method属性指定为post的时候才是post请求。
            剩下所有的请求都是get请求。

            post提交的时候提交的数据格式和get还是一样的，只不过不再地址栏上显示出来。
            POST提交的数据还是：name=value&name=value&name=value.....

            <!--
            get提交
            http://localhost:8080/jd/register?
username=qqq&userpwd=123&gender=1&interest=comic&grade=bk&introduce=good+man
            -->
            <!--
            post提交
            http://localhost:8080/jd/register
            -->
        -->
        <form action="http://localhost:8080/jd/register" method="post">
            用户名
            <input type="text" name="username"/>
            <br>
            密码
            <input type="password" name="userpwd"/>
            <br>
            确认密码
            <input type="password"/>
            <br>
            性别<!--单选按钮的value必须手动指定--><!--checked表示默认选中-->
            <input type="radio" name="gender" value="1" checked/>男
            <input type="radio" name="gender" value="0"/>女
```

```
            <br>
            兴趣爱好
            <input type="checkbox" name="interest" value="game" />游戏
            <input type="checkbox" name="interest" value="bodybuilding" />健身
            <input type="checkbox" name="interest" value="comic" checked/>动漫
            <br>
            学历
            <select name="grade"><!--下拉按钮-->
                <option value ="gz">高中</option>
                <option value ="dz">大专</option>
                <option value ="bk" selected>本科</option><!--默认选中-->
                <option value ="ss">硕士</option>
            </select>
            <br><br>
            简介<!--文本域，文本域没有value属性，用户填写的内容就是value-->
            <textarea rows="10" cols="60" name="introduce"></textarea>
            <br>
            <input type="submit" value="注册" />
            <input type="reset" value="清空" />
        </form>

        <hr >
        <!--超链接也可以提交数据给服务器，但是提交的数据都是固定不变的。-->
        <!--超链接是get请求。不是post请求。-->
        <!--http://localhost:8080/oa/save?username=zhangsan&password=111-->
        <a href="http://localhost:8080/oa/save?username=zhangsan&password=111">提交</a>
    </body>
</html>
```



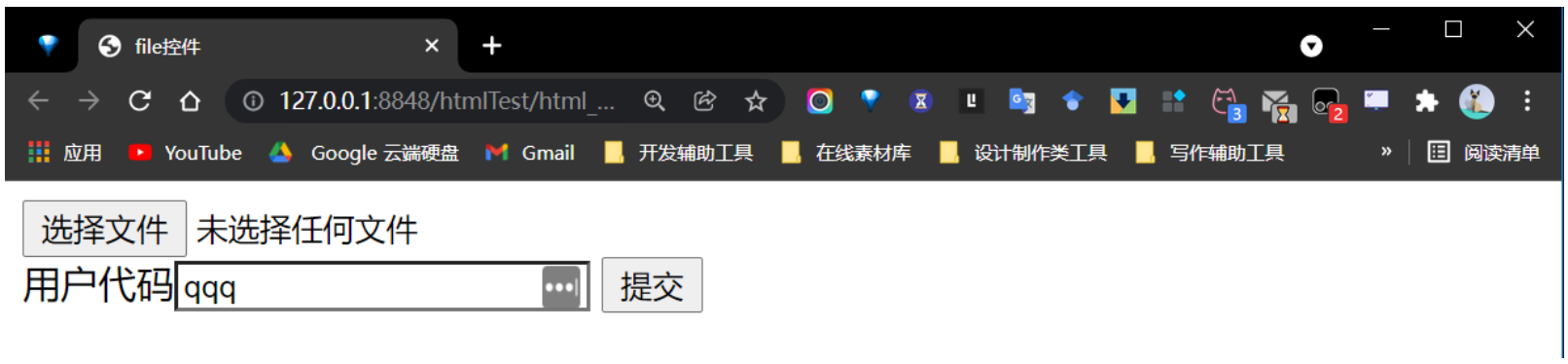
3.13 下拉列表支持多选

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>下拉列表支持多选</title>
    </head>
    <body>
        <!-- multiple="multiple" 支持多选的 size设置显示条目数量。-->
        <select multiple="multiple" size="4">
            <option>江苏省</option>
            <option>河南省</option>
            <option>山东省</option>
            <option>山西省</option>
        </select>
    </body>
</html>
```



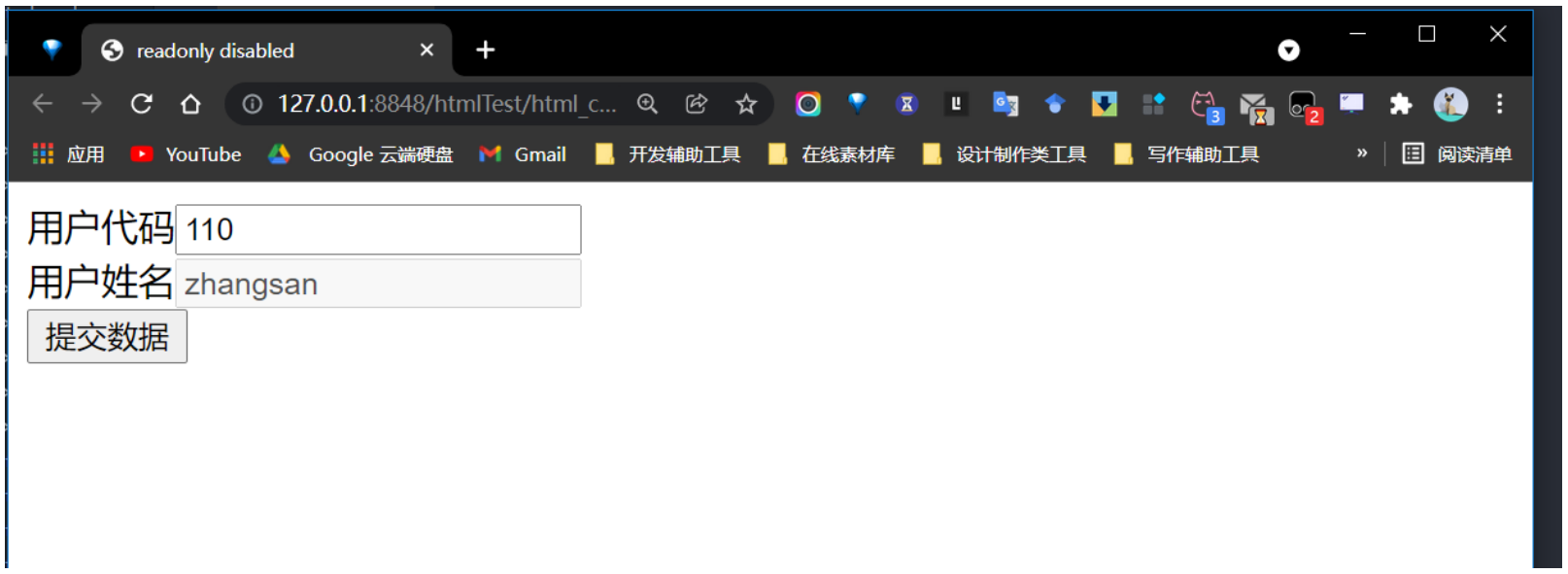
3.14 file控件

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>file控件</title>
    </head>
    <body>
        <!--file控件：文件上传专用。-->
        <input type="file" />

        <form action="http://localhost:8080/oa/save">
            <!--隐藏域：网页上看不到，但是表单提交的时候数据会自动提交给服务器。-->
            <input type="hidden" name="userid" value="111" />

            用户代码<input type="text" name="usercode" />
            <input type="submit" value="提交" />
            <!--http://localhost:8080/oa/save?userid=111&usercode=qqq-->
        </form>
    </body>
</html>
```



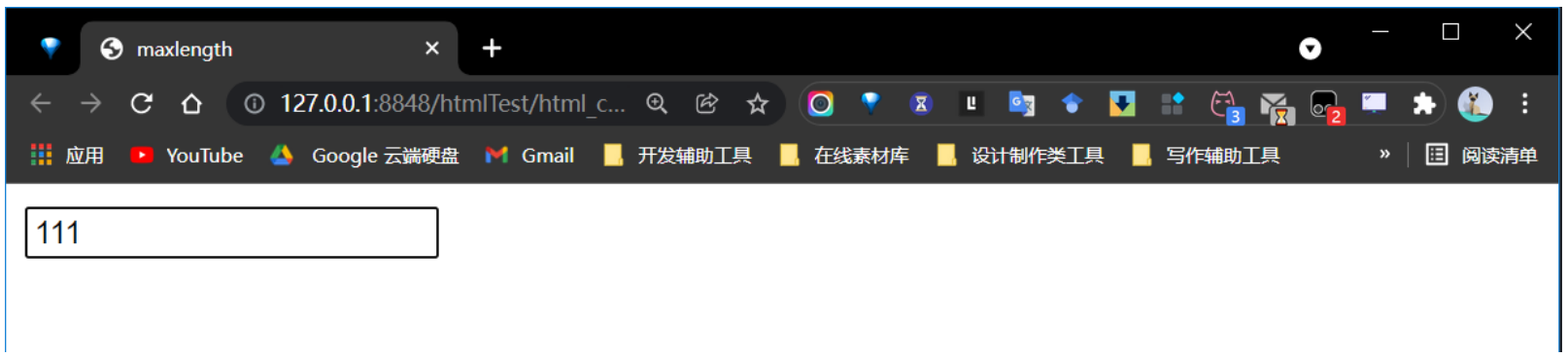
3.15 readonly和disabled

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>readonly disabled</title>
    </head>
    <body>
        <!--
            readonly和disabled相同点：都是只读不能修改。
            但是readonly可以提交给服务器，disabled数据不会提交（即使有name属性也不会提交。）
        -->
        <form action="http://localhost:8080/taobao/save">
            用户代码<input type="text" name="usercode" value="110" readonly />
            <br>
            用户姓名<input type="text" name="username" value="zhangsan" disabled />
            <br>
            <input type="submit" value="提交数据" />
            <!--http://localhost:8080/taobao/save?usercode=110-->
        </form>
    </body>
</html>
```



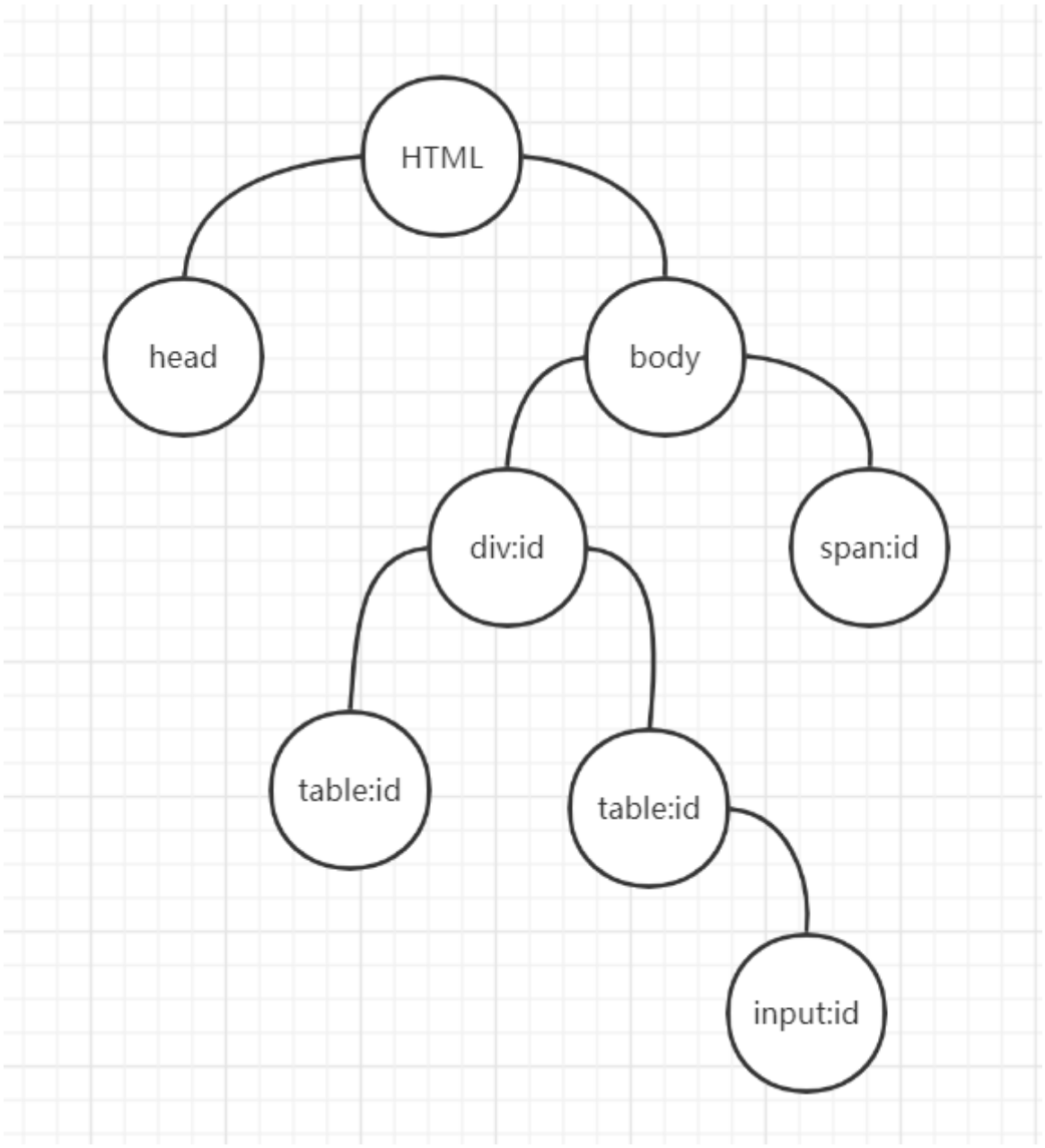
3.16 input控件的maxlength属性

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>maxlength</title>
    </head>
    <body>
        <!--
            maxlength 设置文本框中可输入的字符数量。
        -->
        <input type="text" maxlength="3" />
    </body>
</html>
```



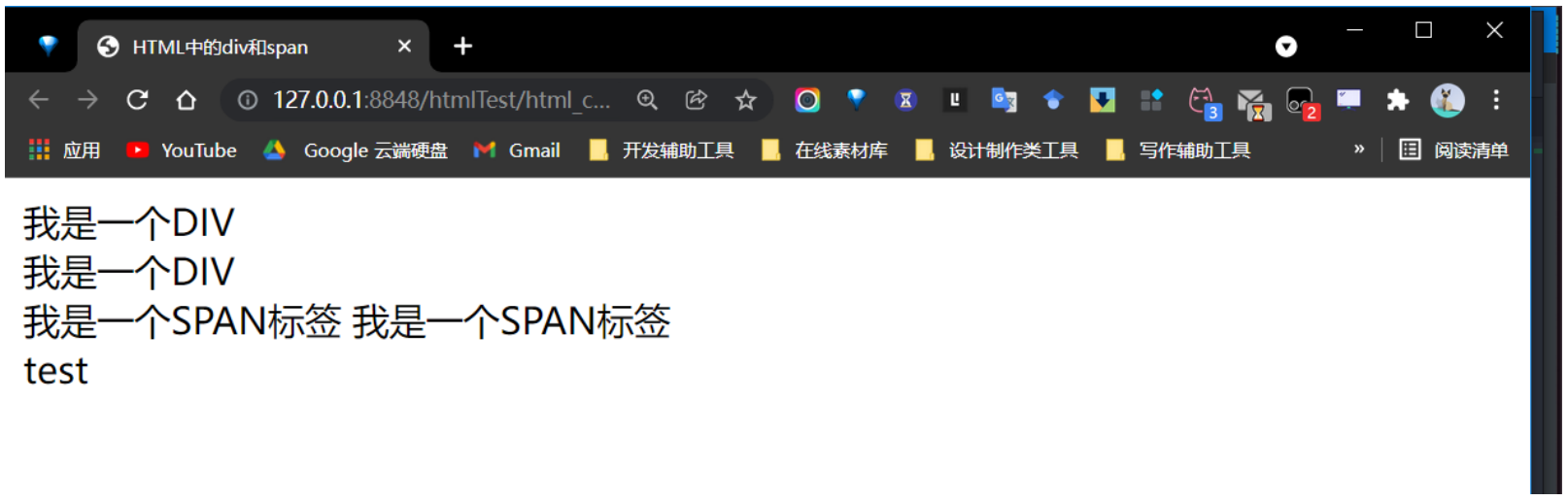
3.17 HTML中元素的id属性

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>HTML中元素的id属性</title>
    </head>
    <body id="mybody">
        <!--
            1、在HTML文档当中，任何元素（节点）都有id属性，id属性是该节点的唯一标识。所以在同一个HTML文档当中id值不能重复。
            2、注意：表单提交数据的时候，只和name有关系，和id无关。
            3、id有什么用？
                javascript语言：可以对HTML文档当中的任意节点进行增删改操作。
                javascript可以对HTML文档当中的任意节点进行增删改，那么增删改之前需要先拿到这个节点，通常我们通过id来拿节点对象。
                id的存在让我们获取元素（节点）更方便。
            4、HTML文档是一棵树，树上有很多节点，每一个节点都有唯一的id。
                javascript主要就是对这棵DOM树上的节点进行增删改的。
                DOM(Document)树。
        -->
        <form id="myform">
            <input type="text" id="username" name="username"/>
            <input type="password" id="userpwd" name="userpwd"/>

            <!--id就是节点的身份证号码，不能重复。-->·
            <!-- <input type="text" id="username" /> -->
        </form>
    </body>
</html>
```



3.18 HTML中的div和span

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>HTML中的div和span</title>
    </head>
    <body>
        <!--
            1、div和span是什么？有什么用？
                * div和span都可以称为“图层”
                * 图层的作用是为了保证页面可以灵活的布局。
                * 图层就是一个一个的盒子，div嵌套div就是盒子套盒子。
                * div和span是可以定位的，只要定下div的左上角的x轴和y轴坐标即可。
            2、其实最早的网页是采用table进行布局的，但是table不灵活，太死板。
            现代的网页开发中div布局使用最多，几乎很少使用table进行布局了。

            3、div和span的区别？
                div独自占用一行（默认情况下）
                span不会独自占用一行。
        -->
        <div id="div1">我是一个DIV</div>
        <div id="div2">我是一个DIV</div>

        <span id="span1">我是一个SPAN标签</span>
        <span id="span2">我是一个SPAN标签</span>

        <div id="div3">
            <div>
                <div>test</div>
            </div>
        </div>

    </body>
</html>
```


HTML中的div和span
127.0.0.1:8848/htmlTest/html_c...
应用
YouTube
Google 云端硬盘
Gmail
开发辅助工具
在线素材库
设计制作类工具
写作辅助工具
阅读清单
我是一个DIV
我是一个DIV
我是一个SPAN标签 我是一个SPAN标签
test

CSS

1、CSS概述

```
1、什么是CSS，有什么作用？
    CSS(Cascading Style Sheet):层叠样式表语言。
    CSS的作用是：
        修饰HTML页面，设置HTML页面中的某些元素的样式，让HTML页面更好看。
        CSS好比是HTML的化妆品一样。
    HTML还是主体，CSS依赖HTML。CSS的存在就是修饰HTML，所以新建的文件还是xx.html文件。
```

```
2、CSS我们要求掌握到什么程度？
    * 常见的CSS样式要求会写。
    * 别人写的CSS样式要能看懂。
```

```
3、在HTML页面中嵌套使用CSS的三种方式：
第一种方式：在标签内部使用style属性来设置元素的CSS样式，这种方式称为内联定义方式。
    语法格式：
        <标签 style="样式名:样式值; 样式名:样式值; 样式名:样式值;..."></标签>

第二种方式：在head标签中使用style块，这种方式被称为样式块方式。
    语法格式：
        <head>
            <style type="text/css">
                选择器 {
                    样式名 : 样式值;
                    样式名 : 样式值;
                    .....
                }
                选择器 {
                    样式名 : 样式值;
                    样式名 : 样式值;
                    .....
                }
            </style>
        </head>

第三种方式：链入外部样式表文件，这种方式最常用。（将样式写到一个独立的xxx.css文件当中，在需要的网页上
直接引入css文件，样式就引入了）
    语法格式：
        <link type="text/css" rel="stylesheet" href="css文件的路径" />
    这种方式易维护，维护成本较低。
        xxx.css文件
            1.html中引入了
            2.html中引入了
            3.html中引入了
            4.html中引入了
```

```
4、HTML标签属性分类
1.基本属性：
    大多数HTML标签都拥有属性，是一个非常庞大群体
    比如 id属性，相当于身份证编号，用于区分HTML标签
    <input type="text" id="one"/>
    <input type="text" id="two"/>
    比如 name属性，相当于人名字,允许一组标签拥有相同name
    <input type="text" id="one"  name="myText"/>
    <input type="text" id="two"  name="myText"/>

2.样式属性：
    是一个非常庞大群体，通知浏览器将HTML标签中数据在
    浏览器中以指定形态展示
    <div style="background-color:red;color:green;width:300px;height:200px"></div>

3.工作状态属性:
    只存在于【表单域标签】中，用于表示【表单域标签】状态.
        checked:存在于radio与checkbox中，表示标签是否被选中
        disabled:表示标签处于不可用状态
        readOnly:表示标签处于只读状态
        seleteced：存在option标签，表示标签是否被选中

4.监听属性：
    监听属性用户与HTML标签之间进行通信通道，监听属性
    用于监听用户在何时对当前标签进行何种操作,当指定
    操作产生时，监听属性将会通知浏览器调用对应JavaScript
    方法处理当前请求
```

```
5、CSS选择器
1.介绍：
```

```
    CSS选择器，实际上就是一组定位条件用于定位HTML标签。
    CSS选择器有9个大的分类
2.CSS选择器语法格式:
      <html>
         <head>
            <!--type='text/css', -->
            <style type="text/css">
                定位条件{
                    "样式属性1":"值1";
                    "样式属性2": "值2"
                }
            </style>
        </head>
    </html>
```

```
6、id选择器
1.介绍:
    根据HTML标签中ID属性的值进行定位
2.语法:
    <style type="text/css">
         #id编号{
            "样式属性1":"值1";
            "样式属性2": "值2"
        }
    </style>
```

```
7、标签选择器
1.介绍:
    根据HTML标签类型进行定位
2.语法:
    <style type="text/css">
        标签类型名{
            "样式属性1":"值1";
            "样式属性2": "值2"
        }
    </style>
```

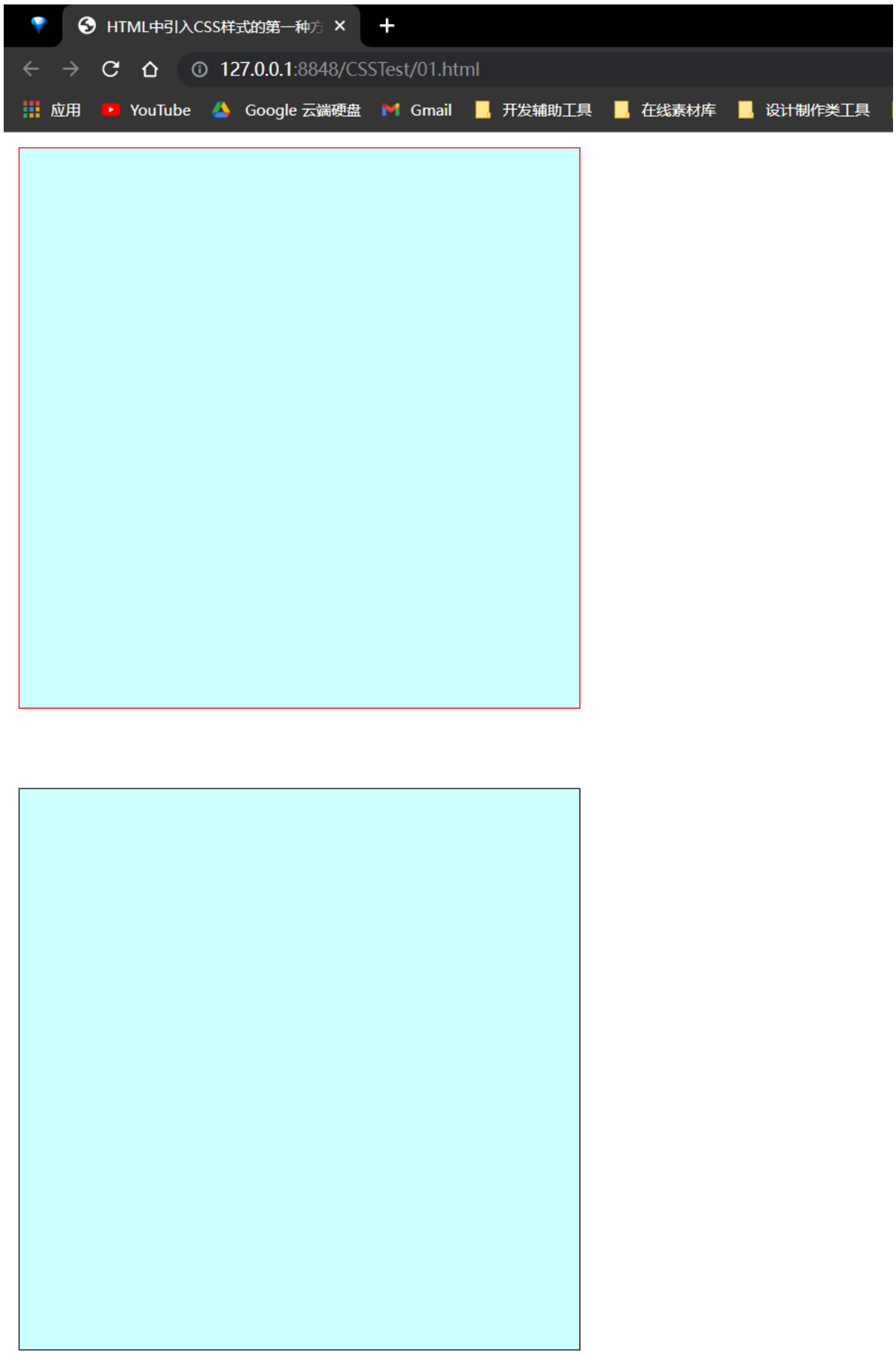
```
8、层级选择器
1.HTML标签之间关系:
    父子关系     兄弟关系
2.父子关系:
    即为包含关系
        <tr>
            <td>
                <input type="text">
            </td>
        </tr>
        td标签是tr标签的子标签
        input标签是td标签的子标签
3.兄弟关系:
    一组标签拥有相同的父标签，并且彼此之间
    没有任何包含关系，即为兄弟
        <body>
            <div>1</div>    大哥
            <p>2</p>        二哥
            <span>3</span> 三弟
        </body>
4.层级选择器介绍:
    根据标签之间父子关系或则兄弟关系进行定位
5.简单的层级选择器
     <style type="text/css">
        定位父标签条件   定位子标签条件{

        }
        找到指定父标签下满足条件的所有子标签
    </style>
```

```
9、自定义选择器
1.介绍:
    如果一组HTML标签之间没有相同的特征，但是却需要
    对指定属性赋值相同内容，此时将自定义选择器绑定
    到对应标签上
2.语法:
    <style type="text/css">
         .自定义选择器名{
             color:red;
        }
    </style>

    <div class="自定义选择器名"></div>
    <p   class="自定义选择器名"></p>
```

2、CSS代码及展示

2.1 内联定义方式

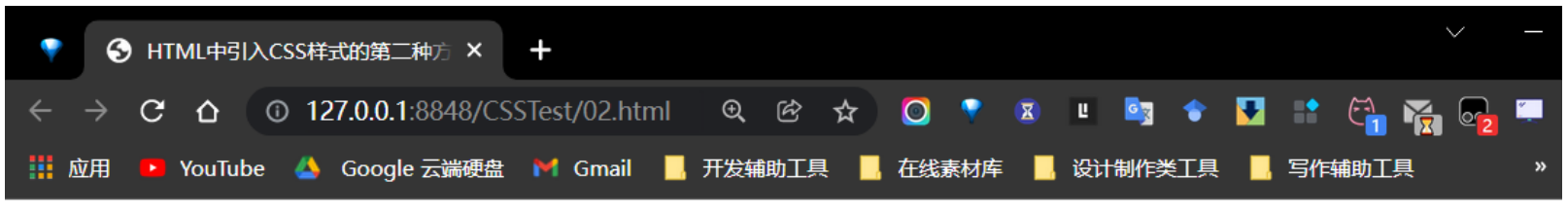
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>HTML中引入CSS样式的第一种方式：内联定义方式</title>
    </head>
    <body>
        <!--
            width 宽度样式
            height 高度样式
            background-color 背景色样式
            display 布局样式（none表示隐藏，block表示显示）
            border-style 边框样式
            border-width 边框宽度
            border-color 边框颜色
        -->
        <div style="width:300px; height:300px; background-color:#CCFFFF; display:block;
        border-color:red; border-width:1px; border-style:solid;"></div>

        <br><br>

        <!--
            样式还能这样写：
                border : 1px solid black;
        -->
        <div style="width:300px; height:300px; background-color:#CCFFFF; display:block;
        border:1px solid black;"></div>
    </body>
</html>
```



2.2 样式块

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>HTML中引入CSS样式的第二种方式：样式块</title>

        <style type="text/css">
            /*  这是CSS的注释。 */
            /*
                id选择器
                语法格式：
                    #id{
                        样式名 : 样式值;
                        样式名 : 样式值;
                        ....
                    }
             */
           #usernameErrorMsg {
                color : red;
                font-size : 12px;
           }

           /*
            标签选择器
            语法格式：
                标签名 {
                    样式名 : 样式值;
                    样式名 : 样式值;
                    样式名 : 样式值;
```

```
                    ....
                }
             标签选择器作用的范围比id选择器广。
            */
            div {
             background-color : black;
             border : 1px solid red;
             width : 100px;
             height : 50px;
            }

            /*
             类选择器
             语法格式:
                 .类名{
                     样式名 : 样式值;
                     样式名 : 样式值;
                     样式名 : 样式值;
                     ....
                 }
            */
            .student {
             border : 1px solid red;
             width : 400px;
             height : 30px;
            }
        </style>
    </head>
    <body>
        <!--
            设置样式字体大小12px，颜色为红色！
        -->
        <span id="usernameErrorMsg">对不起，用户名不能为空！</span>

        <div></div>
        <div></div>

        <!--class相同的标签可以认为是同一类标签。-->
        <br><br><br>
        <input type="text" class="student"/>

        <br><br><br>

        <select class="student">
            <option>研究生</option>
            <option>本科</option>
        </select>
    </body>
</html>
```



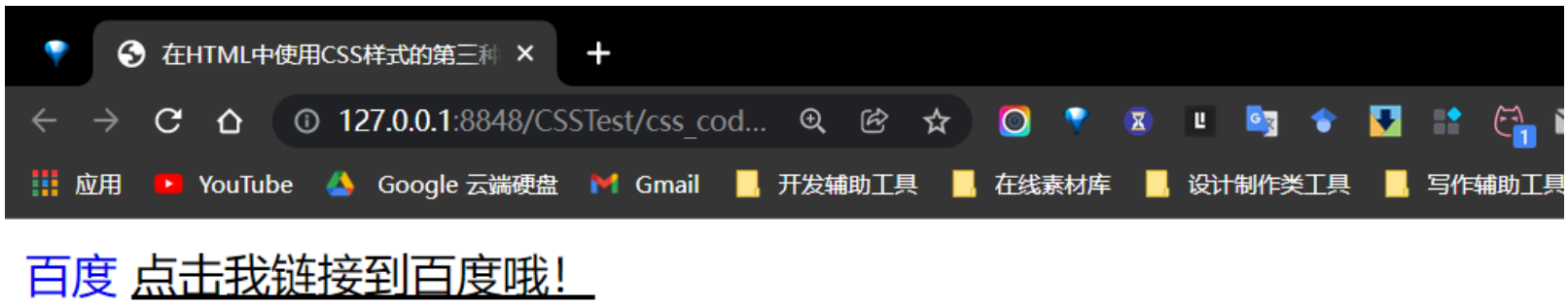
对不起，用户名不能为空！

研究生

2.3 引入外部独立的css文件

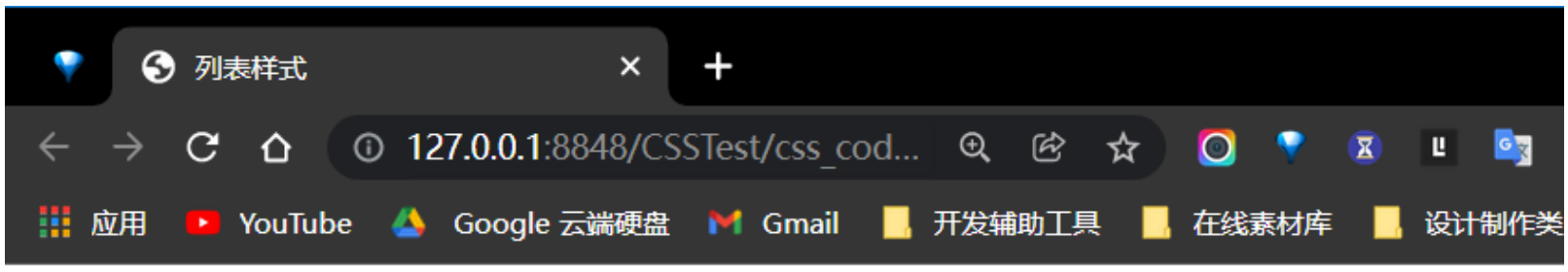
```css
/* 路径：css/1.css */
a{
    text-decoration: none; /*下划线消失*/
}
/*
    cursor : 鼠标样式，pointer是小手，hand也是，但是hand有浏览器兼容问题。建议使用pointer
*/
#baiduSpan {
    text-decoration: underline; /*下划线出现*/
    cursor: pointer;
}
```

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>在HTML中使用CSS样式的第三种方式：引入外部独立的css文件</title>
        <!--引入css-->
        <link rel="stylesheet" type="text/css" href="css/1.css" />
    </head>
    <body>
        <a href="http://www.baidu.com">百度</a>
        <span id="baiduSpan">点击我链接到百度哦！</span>
    </body>
</html>
```



2.4 列表样式

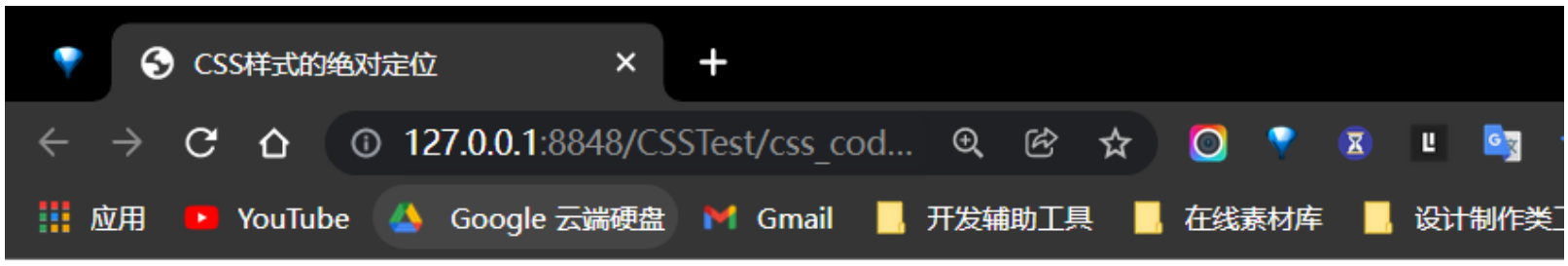
```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>列表样式</title>
        <style type="text/css">
            ul{
                /*list-style-type: none;*/
                /*list-style-type: circle ;*/
                list-style-type: square ;
            }
        </style>
    </head>
    <body>
        <ul>
            <li>中国
                <ul>
                    <li>北京</li>
                    <li>上海</li>
                    <li>重庆</li>
                </ul>
            </li>
            <li>美国</li>
            <li>俄国</li>
        </ul>
    </body>
</html>
```

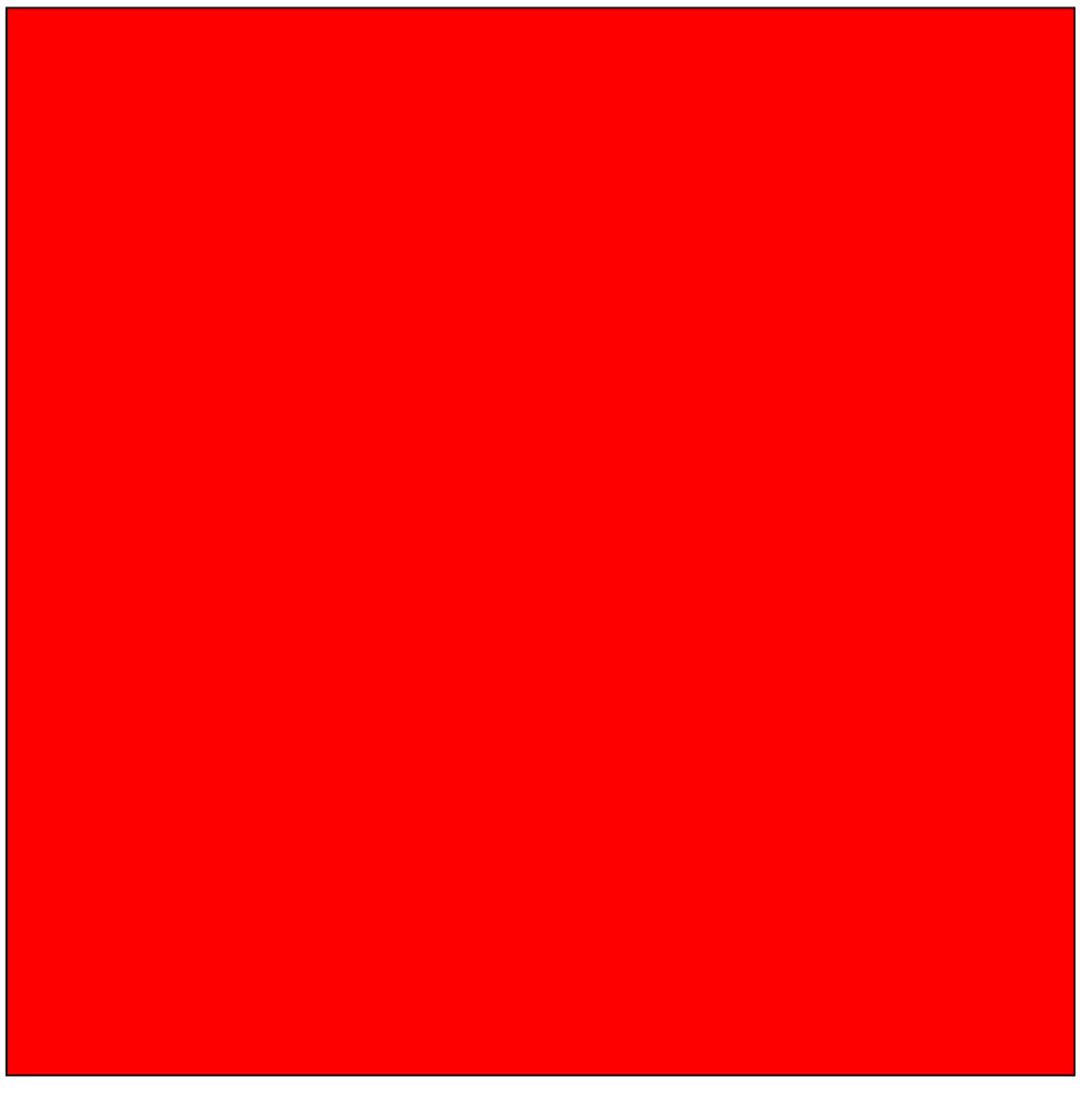


- 中国
  - 北京
  - 上海
  - 重庆
- 美国
- 俄国

2.5 CSS样式的绝对定位

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS样式的绝对定位</title>
        <style type="text/css">
            #div1{
                background-color: red;
                border: 1px black solid;
                width: 300px;
                height: 300px;
                position : absolute; /*绝对定位*/
                left: 100px; /*离左边100像素*/
                top: 100px; /*离顶部100像素*/
            }
        </style>
    </head>
    <body>
        <div id="div1"></div>
    </body>
</html>
```


CSS样式的绝对定位
127.0.0.1:8848/CSSTest/css_cod...
应用
YouTube
Google 云端硬盘
Gmail
开发辅助工具
在线素材库



JavaScript

1、JavaScript概述

```
1、什么是JavaScript，有什么用？
    JavaScript是运行在浏览器上的脚本语言。简称JS。
    JavaScript是网景公司（NetScape）的 布兰登艾奇（JavaScript之父）开发的，最初叫做LiveScript。
    LiveScript的出现让浏览器更加的生动了，不再是单纯的静态页面了。页面更具有交互性。
    在历史的某个阶段，SUN公司和网景公司他们之间有合作关系，SUN公司把LiveScript的名字修改为JavaScript。
    JavaScript这个名字中虽然带有“Java”但是和Java没有任何关系，只是语法上有点类似。他们运行的位置不同，
    Java运行在JVM当中，JavaScript运行在浏览器的内存当中。

    JavaScript程序不需要我们程序员手动编译，编写完源代码之后，浏览器直接打开解释执行。
    JavaScript的“目标程序”以普通文本形式保存，这种语言都叫做“脚本语言”。

    Java的目标程序以.class形式存在，不能使用文本编辑器打开，不是脚本语言。

    网景公司1998年被美国在线收购。

    网景公司最著名的就是领航者浏览器：Navigator浏览器。

    LiveScript的出现，最初的时候是为Navigator浏览器量身定制一门语言，不支持其他浏览器。

    当Navigator浏览器使用非常广泛的时候，微软害怕了，于是微软在最短的时间内组建了一个团队，
    开始研发只支持IE浏览器的脚本语言，叫做JScript。

    JavaScript和JScript并存的年代，程序员是很痛苦的，因为程序员要写两套程序。
    在这种情况下，有一个非营利性组织站出来了，叫做ECMA组织（欧洲计算机协会）
    ECMA根据JavaScript制定了ECMA-262号标准，叫做ECMA-Script。

    现代的javascript和jscript都实现了ECMA-Script规范。（javascript和jscript统一了。）

    以后大家会学习一个叫做JSP的技术，JSP和JS有啥区别？
        JSP : JavaServer Pages（隶属于Java语言的，运行在JVM当中）
        JS : JavaScript（运行在浏览器上。）
```

```
2、在HTML中怎么嵌入JavaScript代码？
    三种方式。
```

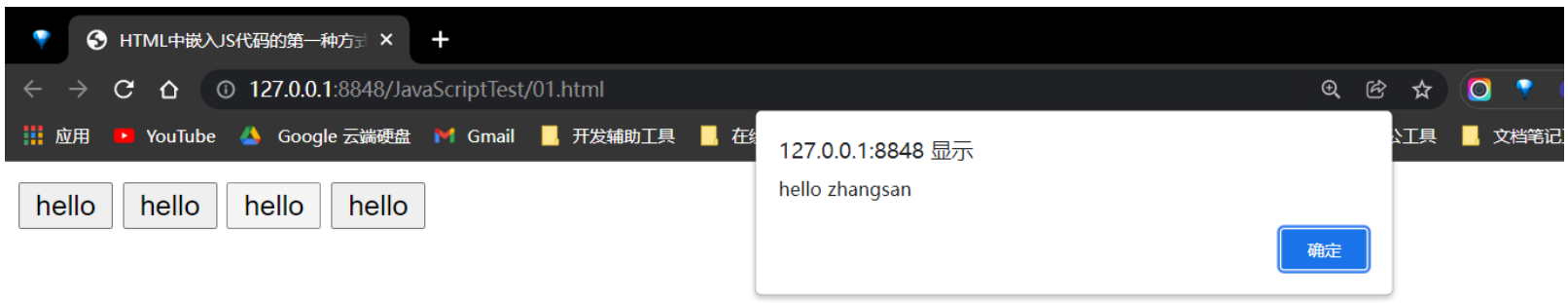
2、JavaScript语言代码

2.1 ECMAScript

2.1.1 HTML中嵌入JS代码的第一种方式

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>HTML中嵌入JS代码的第一种方式</title>
    </head>
    <body>
        <!--
            1、要实现的功能：
                用户点击以下按钮，弹出消息框。

            2、JS是一门事件驱动型的编程语言，依靠事件去驱动，然后执行对应的程序。
            在JS中有很多事件，其中有一个事件叫做：鼠标单击，单词：click。并且任何
            事件都会对应一个事件句柄叫做：onclick。
            【注意：事件和事件句柄的区别是：事件句柄是在事件单词前添加一个on。】，
            而事件句柄是以HTML标签的属性存在的。

            3、onclick="js代码"，执行原理是什么？
                页面打开的时候，js代码并不会执行，只是把这段JS代码注册到按钮的click事件上了。
                等这个按钮发生click事件之后，注册在onclick后面的js代码会被浏览器自动调用。

            4、怎么使用JS代码弹出消息框？
                在JS中有一个内置的对象叫做window，全部小写，可以直接拿来使用，window代表的是浏览器对象。
                window对象有一个函数叫做:alert，用法是：window.alert("消息");这样就可以弹窗了。

            5、JS中的字符串可以使用双引号，也可以使用单引号。

            6、JS中的一条语句结束之后可以使用分号“;”，也可以不用。
        -->
        <input type="button" value="hello" onclick="window.alert('hello js')"/>

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

        <input type="button" value="hello" onclick="window.alert('hello zhangsan')
                                                    window.alert('hello lis')
```

```
                                            window.alert('hello wangwu')"/>

        <!-- window. 可以省略。-->
        <input type="button" value="hello" onclick="alert('hello zhangsan')
                                            alert('hello lis')
                                            alert('hello wangwu')"/>
    </body>
</html>
```



2.1.2 HTML中嵌入JS代码的第二种方式：脚本块

```
<!--
    javascript的脚本块在一个页面当中可以出现多次。没有要求。
    javascript的脚本块出现位置也没有要求，随意。
-->
<script type="text/javascript">
    // alert有阻塞当前页面加载的作用。（阻挡，直到用户点击确定按钮。）
    window.alert("first.......");
</script>

<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>HTML中嵌入JS代码的第二种方式</title>
        <!--样式块-->
        <style type="text/css">
            /*
                css代码
            */
        </style>

        <!--脚本块-->
        <script type="text/javascript">
            window.alert("head.............");
        </script>
    </head>
    <body>
        <input type="button" value="我是一个按钮对象1" />

        <!--第二种方式：脚本块的方式-->
        <script type="text/javascript">
            /*
                暴露在脚本块当中的程序，在页面打开的时候执行，
                并且遵守自上而下的顺序依次逐行执行。（这个代
                码的执行不需要事件）
            */
            window.alert("Hello World!"); // alert函数会阻塞整个HTML页面的加载。

            // JS代码的注释，这是单行注释。
            /*
                JS代码的多行注释。和java一样。
            */
            window.alert("Hello JavaScript!");
        </script>

        <input type="button" value="我是一个按钮对象" />
    </body>
</html>

<script type="text/javascript">
window.alert("last.......");
</script>
```

2.1.3 引入外部独立的js文件

```
//路径  js/1.js
window.alert("hello js!");
window.alert("hello js!");
window.alert("hello js!");
window.alert("hello js!");
window.alert("hello js!");
window.alert("hello js test!");
```

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>HTML中嵌入JS代码的第三种方式：引入外部独立的js文件。</title>
    </head>
    <body>
        <!--在需要的位置引入js脚本文件-->
        <!--引入外部独立的js文件的时候，js文件中的代码会遵循自上而下的顺序依次逐行执行。-->
        <script type="text/javascript" src="js/1.js"></script>

        <!--同一个js文件可以被引入多次。但实际开发中这种需求很少。-->
        <script type="text/javascript" src="js/1.js"></script>

        <!--这种方式不行，结束的script标签必须有。-->
        <!--
        <script type="text/javascript" src="js/1.js" />
        -->
        <!--
        <script type="text/javascript" src="js/1.js"></script>
        -->

        <script type="text/javascript" src="js/1.js">
            // 这里写的代码不会执行。
            // window.alert("Test");
        </script>

        <script type="text/javascript">
            alert("hello jack!");
        </script>
    </body>
</html>
```

2.1.4 关于JS中的变量

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>关于JS中的变量</title>
    </head>
    <body>
        <script type="text/javascript">
            /*
                回顾java中的变量：
                1、java中怎么定义/声明变量？
                    数据类型 变量名;
                    例如：
                        int i;
                        double d;
                        boolean flag;
                2、java中的变量怎么赋值？
                    使用“=”运算符进行赋值运算。（"="运算符右边先执行，将右边执行的结果赋值给左边的变量。）
                    变量名 = 值;
                    例如：
                        i = 10;
                        d = 3.14;
                        flag = false;
                3、java语言是一种强类型语言，强类型怎么理解？
                    java语言存在编译阶段，假设有代码：int i;
                    那么在Java中有一个特点是：java程序编译阶段就已经确定了
                    i变量的数据类型，该i变量的数据类型在编译阶段是int类型，
                    那么这个变量到最终内存释放，一直都是int类型，不可能变成
                    其他类型。
                        int i = 10;
                        double d = i;
                        这行代码是说声明一个新的变量d，double类型，把i变量中保存的值传给d。
                        i还是int类型。

                        i = "abc"; 这行代码编译的时候会报错，因为i变量的数据类型是int类型，不能将字符串赋给i。
```

```
                                java中要求变量声明的时候是什么类型，以后永远都是这种类型，不可变。编译期强行固定变量的数据类型。
                                称为强类型语言。

                                public void sum(int a, int b){}
                                sum(?,?);

                    javascript当中的变量？
                        怎么声明变量？
                            var 变量名;
                        怎么给变量赋值？
                            变量名 = 值;
                        javascript是一种弱类型语言，没有编译阶段，一个变量可以随意赋值，赋什么类型的值都行。
                            var i = 100;
                            i = false;
                            i = "abc";
                            i = new Object();
                            i = 3.14;
                        重点：JavaScript是一种弱类型编程语言。
                 */
                // 在JS当中,当一个变量没有手动赋值的时候,系统默认赋值undefined
                var i;
                // undefined 在JS中是一个具体存在值.
                alert("i = " + i);  //i = undefined

                alert(undefined); // undefined
                var k = undefined;
                alert("k = " + k);// k = undefined

                // 一个变量没有声明/定义,直接访问?
                // alert(age); //语法错误: age is not defined (变量age不存在。不能这样写。)

                var a, b, c = 200;
                alert("a = " + a);// a = undefined
                alert("b = " + b);// b = undefined
                alert("c = " + c);// c = 200

                a = false;
                alert(a); // false

                a = "abc";
                alert(a); // abc

                a = 1.2;
                alert(a); // 1.2
            </script>
        </body>
    </html>
```

2.1.5 JS函数初步

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JS函数初步</title>
    </head>
    <body>
        <script type="text/javascript">
            /*
            1、JS中的函数：
                等同于java语言中的方法，函数也是一段可以被重复利用的代码片段。
                函数一般都是可以完成某个特定功能的。

            2、回顾java中的方法？
                [修饰符列表] 返回值类型 方法名(形式参数列表){
                    方法体;
                }
                例如：
                public static boolean login(String username,String password){
                    ...
                    return true;
                }
                boolean loginSuccess = login("admin","123");

            3、JS中的变量是一种弱类型的，那么函数应该怎么定义呢？
                语法格式：
                    第一种方式：
                        function 函数名(形式参数列表){
                            函数体;
                        }
                    第二种方式：
                        函数名 = function(形式参数列表){
```

```
                            函数体;
                        }
                    重点：JS中的函数不需要指定返回值类型，返回什么类型都行。
             */
            function sum(a, b){
                 // a和b都是局部变量,他们都是形参(a和b都是变量名，变量名随意。)
                 alert(a + b);
            }
            // 函数必须调用才能执行的.
            // sum(10, 20);//30

            // 定义函数sayHello
            sayHello = function(username){
                 alert("hello " + username);
            }
            // 调用函数
            //sayHello("zhangsan");//hello zhangsan
        </script>

        <input type="button" value="hello" onclick="sayHello('jack');" />
        <input type="button" value="计算10和20的求和" onclick="sum(10, 20);" />
    </body>
</html>
```

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JS函数初步</title>
    </head>
    <body>
        <script type="text/javascript">
            /*
            java中的方法有重载机制，JS中的函数能重载吗？
            JS当中的函数在调用的时候，参数的类型没有限制，并且参数的个数也没有限制，JS就是这么随意。（弱类型）

            重载的含义：
            方法名或者函数名一样，形参不同（个数、类型、顺序）
            */
            function sum(a, b){
                return a + b;
            }

            // 调用函数sum
            var retValue = sum(1, 2);
            alert(retValue);//3

            var retValue2 = sum("jack"); // jack赋值给a变量,b变量没有赋值系统默认赋值undefined
            alert(retValue2); // jackundefined //jack + undefined

            var retValue3 = sum();
            alert(retValue3); // NaN (NaN是一个具体存在的值，该值表示不是数字。Not a Number)

            var retValue4 = sum(1, 2, 3);
            alert("结果=" + retValue4); // 结果=3

            function test1(username){
                alert("test1");
            }
            //在JS当中，函数的名字不能重名，当函数重名的时候，后声明的函数会将之前声明的同名函数覆盖。
            function test1(){
                alert("test1 test1");
            }

            test1("lisi"); // test1 test1 // 这个调用的是第二个test1()函数.
        </script>
    </body>
</html>
```

2.1.6 JS的局部变量和全局变量

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JS的局部变量和全局变量</title>
    </head>
    <body>
        <script type="text/javascript">
            /*
            全局变量：
                在函数体之外声明的变量属于全局变量，全局变量的生命周期是：
```

```
                浏览器打开时声明，浏览器关闭时销毁，尽量少用。因为全局变量会一直在浏览器的内存当中，耗费内存空间。
                能使用局部变量尽量使用局部变量。
            局部变量：
                在函数体当中声明的变量，包括一个函数的形参都属于局部变量，
                局部变量的生命周期是：函数开始执行时局部变量的内存空间开辟，函数执行结束之后，局部变量的内存空间释放。
                局部变量生命周期较短。
            */
            // 全局变量
            var i = 100;
            function accessI(){
                // 访问的是全局变量
                alert("i = " + i);
            }
            accessI();//i = 100

            // 全局变量
            var username = "jack";
            function accessUsername(){
                // 局部变量
                var username = "lisi";
                // 就近原则:访问局部变量
                alert("username = " + username); // username = lisi
            }
            // 调用函数
            accessUsername();
            // 访问全局变量
            alert("username = " + username); // username = jack

            function accessAge(){
                var age = 20;
                alert("年龄 = " + age);
            }
            accessAge(); //年龄 = 20
            // 报错(语法不对)
            // alert("age = " + age);

            // 以下语法是很奇怪的.
            function myfun(){
                // 当一个变量声明的时候没有使用var关键字,那么不管这个变量是在哪里声明的,都是全局变量.
                myname = "dujubin";
            }
            // 访问函数
            myfun();
            alert("myname = " + myname); // myname = dujubin
        </script>
    </body>
</html>
```

2.1.7 JS中的数据类型

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JS中的数据类型</title>
    </head>
    <body>
        <script type="text/javascript">
            /*
            1、虽然JS中的变量在声明的时候不需要指定数据类型，但是在赋值，每一个数据还是有类型的，所以
            这里也需要学习一下JS包括哪些数据类型？
                JS中数据类型有：原始类型、引用类型。
                原始类型：Undefined、Number、String、Boolean、Null
                引用类型：Object以及Object的子类
            2、ES规范(ECMAScript规范)，在ES6之后，又基于以上的6种类型之外添加了一种新的类型：Symbol
            3、JS中有一个运算符叫做typeof，这个运算符可以在程序的运行阶段动态的获取变量的数据类型。
                typeof运算符的语法格式：
                    typeof 变量名
                typeof运算符的运算结果是以下6个字符串之一：注意字符串都是全部小写。
                    "undefined"
                    "number"
                    "string"
                    "boolean"
                    "object"
                    "function"
            4、在JS当中比较字符串是否相等使用“==”完成。没有equals。
            */
            // 求和,要求a变量和b变量将来的数据类型必须是数字,不能是其他类型
            // 因为以下定义的这个sum函数是为了完成两个数字的求和.
            function sum(a, b){
                if(typeof a == "number" && typeof b == "number"){
                    return a + b;
                }
```

```
                alert(a + "," + b + "必须都为数字! ");
            }
            // 别人去调用以上你写的sum函数.
            var retValue = sum(false, "abc");// false,abc必须都为数字!
            alert(retValue); // undefined
            var retValue2 = sum(1, 2);
            alert(retValue2); // 3

            var i;
            alert(typeof i); // undefined

            var k = 10;
            alert(typeof k); // number

            var f = "abc";
            alert(typeof f); // string

            var d = null;
            alert(typeof d); // object  //null属于Null类型,但是typeof运算符的结果是"object"

            var flag = false;
            alert(typeof flag); // boolean

            var obj = new Object();
            alert(typeof obj); // object

            // sayHello是一个函数.
            function sayHello(){}
            alert(typeof sayHello); // function
        </script>
    </body>
</html>
```

2.1.8 Undefined类型

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Undefined类型</title>
    </head>
    <body>
        <script type="text/javascript">
            /*
                Undefined类型只有一个值，这个值就是 undefined
                当一个变量没有手动赋值，系统默认赋值undefined
                或者也可以给一个变量手动赋值undefined。
            */
            var i; // undefined
            var k = undefined; // undefined

            alert(i == k); // true

            var y = "undefined"; // "undefined" 字符串
            alert(y == k); // false
        </script>
    </body>
</html>
```

2.1.9 Number类型

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Number类型</title>
    </head>
    <body>
        <script type="text/javascript">
        /*
            1、Number类型包括哪些值？
                -1 0 1 2 2.3 3.14 100 .... NaN Infinity
                整数、小数、正数、负数、不是数字、无穷大都属于Number类型。
            2、isNaN() : 结果是true表示不是一个数字，结果是false表示是一个数字。
            3、parseInt()函数: 可以将字符串自动转换成数字,并且取整数位.
            4、parseFloat()函数: 可以将字符串自动转换成数字.
            5、Math.ceil() 函数（Math是数学类，数学类当中有一个函数叫做ceil()，作用是向上取整。）
        */
            var v1 = 1;
            var v2 = 3.14;
            var v3 = -100;
            var v4 = NaN; //不是一个数字
```

```
            var v5 = Infinity;//无穷大

            alert(typeof v1);// "number"
            alert(typeof v2);// "number"
            alert(typeof v3);// "number"
            alert(typeof v4);// "number"
            alert(typeof v5);// "number"

            // 关于NaN (表示Not a Number，不是一个数字，但属于Number类型)
            // 什么情况下结果是一个NaN呢？
            // 运算结果本来应该是一个数字,最后算完不是一个数字的时候,结果是NaN.
            var a = 100;
            var b = "中国人";
            alert(a / b); // 除号显然最后结果应该是一个数字,但是运算的过程中导致最后不是一个数字,那么最后的结果是NaN

            var e = "abc";
            var f = 10;
            alert(e + f); // "abc10"

            // Infinity (当除数为0的时候，结果为无穷大)
            alert(10 / 0);

            // 思考:在JS中10 / 3 = ?
            alert(10 / 3); // 3.3333333333333335

            // 关于isNaN函数?
            // 用法：isNaN(数据) ,结果是true表示不是一个数字， 结果是false表示是一个数字.
            // isNaN : is Not a Number
            function sum(a, b){
                if(isNaN(a) || isNaN(b)){
                    alert("参与运算的必须是数字! ");
                    return;
                }
                return a + b;
            }
            sum(100, "abc"); // 参与运算的必须是数字!
            alert(sum(100, 200)); // 300

            // parseInt():可以将字符串自动转换成数字,并且取整数位.
            alert(parseInt("3.9999")); // 3
            alert(parseInt(3.9999)); // 3

            // parseFloat():可以将字符串自动转换成数字.
            alert(parseFloat("3.14") + 1); // 4.140000000000001
            alert(parseFloat("3.2") + 1); // 4.2

            // Math.ceil() 向上取整
            alert(Math.ceil("2.1")); // 3
        </script>
    </body>
</html>
```

2.1.10 Boolean类型

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Boolean类型</title>
    </head>
    <body>
        <script type="text/javascript">
        /*
            1、 JS中的布尔类型永远都只有两个值：true和false （这一点和java相同。）
            2、在Boolean类型中有一个函数叫做：Boolean()。
                语法格式:
                    Boolean(数据)
                Boolean()函数的作用是将非布尔类型转换成布尔类型。
        */
            var username = "lucy";
            //var username = "";
            if(Boolean(username)){ //为 "lucy" 进去，为""进else
                alert("欢迎你" + username);
            }else{
                alert("用户名不能为空! ");
            }

            if(username){ //为 "lucy" 进去，为""进else
                alert("欢迎你" + username);
            }else{
                alert("用户名不能为空! ");
            }

```

```
            // 规律:“有"就转换成true,"没有"就转换成false.
            alert(Boolean(1));          // true
            alert(Boolean(0));          // false
            alert(Boolean(""));         // false
            alert(Boolean("abc"));      // true
            alert(Boolean(null));       // false
            alert(Boolean(NaN));        // false
            alert(Boolean(undefined)); // false
            alert(Boolean(Infinity));  // true

            /*while(10 / 3){ // 死循环
                alert("hehe");
            }*/

             for(var i = 0; i < 10; i++){
                alert("i = " + i);
             }
             // Null类型只有一个值,null
            alert(typeof null); // "object"
        </script>
    </body>
</html>
```

2.1.11 String类型

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>String类型</title>
    </head>
    <body>
        <script type="text/javascript">
        /*
        String类型：
            1、在JS当中字符串可以使用单引号，也可以使用双引号。
                var s1 = 'abcdef';
                var s2 = "test";
            2、在JS当中，怎么创建字符串对象呢？
                两种方式：
                    第一种：var s = "abc";
                    第二种（使用JS内置的支持类String）：var s2 = new String("abc");
                需要注意的是：String是一个内置的类，可以直接用，String的父类是Object。
            3、无论小string还是大String，他们的属性和函数都是通用的。
            4、关于String类型的常用属性和函数？
                常用属性：
                    length 获取字符串长度
                常用函数：
                    indexOf          获取指定字符串在当前字符串中第一次出现处的索引
                    lastIndexOf      获取指定字符串在当前字符串中最后一次出现处的索引
                    replace          替换
                    substr           截取子字符串
                    substring        截取子字符串
                    toLowerCase      转换小写
                    toUpperCase      转换大写
                    split            拆分字符串
        */
            // 小string(属于原始类型String)
            var x = "king";
            alert(typeof x); // "string"

            // 大String(属于Object类型)
            var y = new String("abc");
            alert(typeof y); // "object"

            // 获取字符串的长度
            alert(x.length); // 4
            alert(y.length); // 3

            alert("http://www.baidu.com".indexOf("http")); // 0
            alert("http://www.baidu.com".indexOf("https")); // -1

            // 判断一个字符串中是否包含某个子字符串？
            alert("http://www.baidu.com".indexOf("https") >= 0 ? "包含" : "不包含"); // 不包含

            // replace (注意：只替换了第一个)
            alert("name=value%name=value%name=value".replace("%","&")); //
name=value&name=value%name=value

            // 继续调用replace方法,就会替换第“二”个.
            // 想全部替换需要使用正则表达式.
            alert("name=value%name=value%name=value".replace("%","&").replace("%", "&")); //
name=value&name=value&name=value
```

```
            // 考点:经常问 substr和substring的区别?
            // substr(startIndex, length)
            alert("abcdefxyz".substr(2,4)); //cdef
            // substring(startIndex, endIndex) 注意:含头不含尾
            alert("abcdefxyz".substring(2,4)); //cd
        </script>
    </body>
</html>
```

2.1.12 Object类型

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Object类型</title>
    </head>
    <body>
        <script type="text/javascript">
        /*
            Object类型:
                1、Object类型是所有类型的超类，自定义的任何类型，默认继承Object。
                2、Object类包括哪些属性?
                    prototype属性（常用的，主要是这个）：作用是给“类”动态的扩展属性和函数。
                    constructor属性
                3、Object类包括哪些函数?
                    toString()
                    valueOf()
                    toLocaleString()
                4、在JS当中定义的类默认继承Object，会继承Object类中所有的属性以及函数。
                换句话说，自己定义的类中也有prototype属性。
                5、在JS当中怎么定义类?怎么new对象?
                    定义类的语法:
                        第一种方式:
                            function 类名(形参){
                            }
                        第二种方式:
                            类名 = function(形参){
                            }
                    创建对象的语法:
                        new 构造方法名(实参); // 构造方法名和类名一致。
        */
            function sayHello(){
            }
            // 把sayHello当做一个普通的函数来调用.
            sayHello();
            // 这种方式就表示把sayHello当做一个类来创建对象.
            var obj = new sayHello(); // obj是一个引用,保存内存地址指向堆中的对象.

            // 定义一个学生类
            function Student(){
                alert("Student.....");
            }
            // 当做普通函数调用
            Student(); // Student.....
            // 当做类来创建对象
            var stu = new Student(); // Student.....
            alert(stu); // [object Object]

            // JS中的类的定义,同时又是一个构造函数的定义
            // 在JS中类的定义和构造函数的定义是放在一起来完成的.
            function User(a, b, c){ // a b c是形参,属于局部变量.
                // 声明属性 (this表示当前对象)
                // User类中有三个属性:sno/sname/sage
                this.sno = a;
                this.sname = b;
                this.sage = c;
            }
            // 创建对象
            var u1 = new User(111, "zhangsan", 30);
            // 访问对象的属性
            alert(u1.sno);   // 111
            alert(u1.sname); // zhangsan
            alert(u1.sage);  // 30

            var u2 = new User(222, "jackson", 55);
            alert(u2.sno);   // 222
            alert(u2.sname); // jackson
            alert(u2.sage);  // 55
            // 访问一个对象的属性,还可以使用这种语法
            alert(u2["sno"]);   // 222
            alert(u2["sname"]); // jackson
```

```
            alert(u2["sage"]);  // 55

            // 定义类的另一种语法
            Emp = function(ename,sal){
                // 属性
                this.ename = ename;
                this.sal = sal;
            }
            var e1 = new Emp("SMITH", 800);
            alert(e1["ename"] + "," + e1.sal); // SMITH,800

            Product = function(pno,pname,price){
                // 属性
                this.pno = pno;
                this.pname = pname;
                this.price = price;
                // 函数
                this.getPrice = function(){
                    return this.price;
                }
             }
             var xigua = new Product(111, "西瓜", 4.0);
             var pri = xigua.getPrice();
             alert(pri); // 4.0

             // 可以通过prototype这个属性来给类动态扩展属性以及函数
             Product.prototype.getPname = function(){
                 return this.pname;
             }

             // 调用后期扩展的getPname()函数
             var pname = xigua.getPname();
             alert(pname) // 西瓜

            // 给String扩展一个函数
             String.prototype.suiyi = function(){
                alert("这是给String类型扩展的一个函数，叫做suiyi");
             }
             "abc".suiyi(); // 这是给String类型扩展的一个函数，叫做suiyi
        </script>
    </body>
</html>

<!--
    java语言怎么定义类，怎么创建对象？（强类型）
        public class User{
            private String username;
            private String password;
            public User(){
            }
            public User(String username,String password){
                this.username = username;
                this.password = password;
            }
        }
        User user = new User();
        User user = new User("lisi","123");


    JS语言怎么定义类，怎么创建对象？（弱类型）
        User = function(username,password){
            this.username = username;
            this.password = password;
        }
        var u = new User();
        var u = new User("zhangsan");
        var u = new User("zhangsan","123");
-->
```

2.1.13 null、NaN、undefined的区别

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>null NaN undefined这三个值有什么区别</title>
    </head>
    <body>
        <script type="text/javascript">
            // null NaN undefined 数据类型不一致.
            alert(typeof null); // object
            alert(typeof NaN); // number
            alert(typeof undefined); // undefined

```

```
            // null和undefined可以等同.
            alert(null == NaN); // false
            alert(null == undefined); // true
            alert(undefined == NaN); // false

            // 在JS当中有两个比较特殊的运算符
            // ==(等同运算符：只判断"值"是否相等)
            // ===(全等运算符：既判断"值"是否相等，又判断"数据类型"是否相等)
            alert(null === NaN); // false
            alert(null === undefined); // false
            alert(undefined === NaN); // false

            alert(1 == true); // true
            alert(1 === true); // false
        </script>
    </body>
</html>
```

2.1.14 JS的常用事件-注册事件的两种方式

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JS的常用事件-注册事件的两种方式</title>
    </head>
    <body>
        <script type="text/javascript">
        /*
            JS中的事件：
                blur        失去焦点
                focus       获得焦点

                click       鼠标单击
                dblclick    鼠标双击

                keydown     键盘按下
                keyup       键盘弹起

                mousedown   鼠标按下
                mouseover   鼠标经过
                mousemove   鼠标移动
                mouseout    鼠标离开
                mouseup     鼠标弹起

                reset       表单重置
                submit      表单提交

                change      下拉列表选中项改变，或文本框内容改变
                select      文本被选定
                load        页面加载完毕（整个HTML页面中所有的元素全部加载完毕之后发生。）

            任何一个事件都会对应一个事件句柄，事件句柄是在事件前添加on。
            onXXX这个事件句柄出现在一个标签的属性位置上。（事件句柄以属性的形式存在。）
        */

        // 对于当前程序来说,sayHello函数被称为回调函数(callback函数)
        // 回调函数的特点:自己把这个函数代码写出来了,但是这个函数不是自己负责调用,由其他程序负责调用该函数.
            function sayHello(){
                alert("hello js!");
            }
        </script>

        <!--注册事件的第一种方式，直接在标签中使用事件句柄-->
        <!--以下代码的含义是：将sayHello函数注册到按钮上，等待click事件发生之后，该函数被浏览器调用。我们称这个函数为回调函数。-->
        <input type="button" value="hello" onclick="sayHello()"/>

        <input type="button" value="hello2" id="mybtn" />
        <input type="button" value="hello3" id="mybtn1" />
        <input type="button" value="hello4" id="mybtn2" />
        <script type="text/javascript">
            function doSome(){
                alert("do some!");
            }
            /*
                第二种注册事件的方式，是使用纯JS代码完成事件的注册。
            */
           // 第一步:先获取这个按钮对象(document是全部小写，内置对象，可以直接用，document就代表整个HTML页面)
           var btnObj = document.getElementById("mybtn");
           // 第二步:给按钮对象的onclick属性赋值
           btnObj.onclick = doSome; // 注意:千万别加小括号. btnObj.onclick = doSome();这是错误的写法.
                                    // 这行代码的含义是,将回调函数doSome注册到click事件上.
```

```
63
64              var mybtn1 = document.getElementById("mybtn1");
65              mybtn1.onclick = function(){ // 这个函数没有名字,叫做匿名函数,这个匿名函数也是一个回调函数.
66                  alert("test.........."); // 这个函数在页面打开的时候只是注册上,不会被调用,在click事件发生之后才会调用.
67              }
68
69              document.getElementById("mybtn2").onclick = function(){
70                  alert("test22222222.........");
71              }
72          </script>
73
74      </body>
75  </html>
76
77  <!--
78  java中也有回调函数机制:
79      public class MyClass{
80          public static void main(String[] args){
81              // 主动调用run()方法，站在这个角度看run()方法叫做正向调用。
82              run();
83          }
84
85          // 站在run方法的编写者角度来看这个方法，把run方法叫做回调函数。
86          public static void run(){
87              System.out.println("run...");
88          }
89      }
90  -->
```

2.1.15 JS代码的执行顺序

```
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <meta charset="utf-8">
5           <title></title>
6       </head>
7       <!-- load事件什么时候发生？页面全部元素加载完毕之后才会发生。-->
8       <body onload="ready()">
9           <script type="text/javascript">
10              /*
11              // 第一步:根据id获取节点对象
12              var btn = document.getElementById("btn"); // 返回null(因为代码执行到此处的时候id="btn"的元素还没有加
    载到内存)
13              // 第二步:给节点对象绑定事件
14              btn.onclick = function(){
15                  alert("hello js");
16              }
17              */
18              function ready(){
19                  var btn = document.getElementById("btn");
20                  btn.onclick = function(){
21                   alert("hello js");
22                  }
23              }
24          </script>
25          <input type="button" value="hello" id="btn" />
26      </body>
27  </html>
```

```
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <meta charset="utf-8">
5           <title>JS代码的执行顺序</title>
6       </head>
7       <body>
8           <script type="text/javascript">
9               // 页面加载的过程中,将a函数注册给了load事件
10              // 页面加载完毕之后,load事件发生了,此时执行回调函数a
11              // 回调函数a执行的过程中,把b函数注册给了id="btn"的click事件
12              // 当id="btn"的节点发生click事件之后,b函数被调用并执行.
13              window.onload = function(){ // 这个回调函数叫做a
14                  document.getElementById("btn").onclick = function(){ // 这个回调函数叫做b
15                      alert("hello js.........");
16                  }
17              }
18          </script>
19
20          <input type="button" value="hello" id="btn" />
21      </body>
22  </html>
```

2.1.16 JS代码设置节点的属性

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JS代码设置节点的属性</title>
    </head>
    <body>
        <script type="text/javascript">
            window.onload = function(){
                document.getElementById("btn").onclick = function(){
                    var mytext = document.getElementById("mytext");
                    // 一个节点对象中只要有的属性都可以"."
                    if(mytext.type == "text"){
                        mytext.type = "checkbox";
                    }else if(mytext.type == "checkbox"){
                        mytext.type = "text";
                    }
                }
            }
        </script>
        <input type="text" id="mytext"/>
        <input type="button" value="将文本框修改为复选框" id="btn"/>
    </body>
</html>
```

2.1.17 JS代码捕捉回车键

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JS代码捕捉回车键</title>
    </head>
    <body>
        <script type="text/javascript">
            window.onload = function(){
                var usernameElt = document.getElementById("username");
                // 回车键的键值是13
                // ESC键的键值是27
                /*
                usernameElt.onkeydown = function(a, b, c){
                    // 获取键值
                    // alert(a); // [object KeyboardEvent]
                    // alert(b);
                    // alert(c);
                }
                */
                /*
                usernameElt.onkeydown = function(event){
                    // 获取键值
                    // 对于“键盘事件对象"来说,都有keyCode属性用来获取键值.
                    alert(event.keyCode);
                }
                */
                usernameElt.onkeydown = function(event){
                    if(event.keyCode === 13){
                        alert("正在进行验证....");
                    }
                }
            }
        </script>

        <input type="text" id="username"/>
    </body>
</html>
```

2.1.18 JS的void运算符

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JS的void运算符</title>
    </head>
    <body>
        <!--
            void运算符的语法：void(表达式)
            运算原理：执行表达式，但不返回任何结果。
                javascript:void(0)
                其中javascript:作用是告诉浏览器后面是一段JS代码。
```

```
                以下程序的javascript:是不能省略的。
        -->
        <a href="javascript:void(0)" onclick="window.alert('test code')">
            既保留住超链接的样式，同时用户点击该超链接的时候执行一段JS代码，但页面还不能跳转。
        </a>

        <br>

        <a href="javascript:void(100)" onclick="window.alert('test code')">
            既保留住超链接的样式，同时用户点击该超链接的时候执行一段JS代码，但页面还不能跳转。
        </a>

        <br>

        <!--void() 这个小括号当中必须有表达式-->
        <a href="javascript:void()" onclick="window.alert('test code')">
            既保留住超链接的样式，同时用户点击该超链接的时候执行一段JS代码，但页面还不能跳转。
        </a>

        <br><br><br>
    </body>
</html>
```

2.1.19 JS的控制语句

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JS的控制语句</title>
    </head>
    <body>
        <script type="text/javascript">
        /*
            1、if
            2、switch

            3、while
            4、do .. while..
            5、for循环

            6、break
            7、continue

            8、for..in语句（了解）
            9、with语句（了解）
        */
            // 创建JS数组
            var arr = [false,true,1,2,"abc",3.14]; // JS中数组中元素的类型随意.元素的个数随意.
            // 遍历数组
            for(var i = 0; i < arr.length; i++){
                alert(arr[i]);
            }
            // for..in
            for(var i in arr){
                //alert(i);//下标
                alert(arr[i]);
            }

            // for..in语句可以遍历对象的属性
            User = function(username,password){
                this.username = username;
                this.password = password;
            }

            var u = new User("张三", "444");
            alert(u.username + "," + u.password);//张三,444
            alert(u["username"] + "," + u["password"]);//张三,444
            for(var shuXingMing in u){
                //alert(shuXingMing)
                //alert(typeof shuXingMing) // shuXingMing是一个字符串
                alert(u[shuXingMing]);//属性名
            }

            alert(u.username);//张三
            alert(u.password);//444

            with(u){
                alert(username + "," + password);//张三,444
            }
        </script>
    </body>
</html>
```

```
<!--
public class Test{
    public static void main(String[] args){
        int[] arr = {1,2,3,4,5,6};
        int[] arr2 = new int[5]; // 等同于：int[] arr2 = {0,0,0,0,0};
        String[] arr3 = {"a","b","c"};
        String[] arr4 = new String[3]; // 等同于：String[] arr4 = {null,null,null};
    }
}
-->
```

2.2 DOM编程

2.2.1 获取文本框的value

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>DOM编程-获取文本框的value</title>
    </head>
    <body>
        <script type="text/javascript">
            /*
                1、JavaScript包括三大块：
                    ECMAScript：JS的核心语法（ES规范 / ECMA-262标准）
                    DOM：Document Object Model（文档对象模型：对网页当中的节点进行增删改的过程。）HTML文档被当做一棵DOM树来看待。
                        var domObj = document.getElementById("id");
                    BOM：Browser Object Model（浏览器对象模型）
                        关闭浏览器窗口、打开一个新的浏览器窗口、后退、前进、浏览器地址栏上的地址等，都是BOM编程。
                2、DOM和BOM的区别和联系？
                    BOM的顶级对象是：window
                    DOM的顶级对象是：document
                    实际上BOM是包括DOM的！
             */
            /*
            window.onload = function(){
                //var btnElt = window.document.getElementById("btn");
                var btnElt = document.getElementById("btn");
                alert(btnElt); // object HTMLInputElement
            }
            */

            window.onload = function(){
                var btnElt = document.getElementById("btn");
                btnElt.onclick = function(){
                    // 获取username节点
                    var usernameElt = document.getElementById("username");
                    var username = usernameElt.value;
                    alert(username);

                    // alert(document.getElementById("username").value);

                    // 可以修改它的value
                    document.getElementById("username").value = "zhangsan";
                }
            }
        </script>
        <input type="text" id="username" />
        <input type="button" value="获取文本框的value" id="btn"/>
        <!--
        <input type="button" id="btn" value="hello" />
        -->


        <hr>
        <script type="text/javascript">
            window.onload = function(){
                document.getElementById("setBtn").onclick = function(){
                    document.getElementById("username2").value = document.getElementById("username1").value;
                }
            }
        </script>
        <input type="text" id="username1" />
        <br>
        <input type="text" id="username2" />
        <br>
        <input type="button" value="将第一个文本框中的value赋值到第二个文本框上" id="setBtn" />

        <hr >
```

```
        <!--blur事件：失去焦点事件-->
        <!--以下代码中的this代表的是当前input节点对象,this.value就是这个节点对象的value属性。-->
        <input type="text" onblur="alert(this.value)" />
    </body>
</html>
```

2.2.2 innerHTML和innerText操作div和span

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>DOM编程-innerHTML和innerText操作div和span</title>
        <style type="text/css">
            #div1{
                background-color: aquamarine;
                width: 300px;
                height: 300px;
                border: 1px black solid;
                position: absolute;
                top: 100px;
                left: 100px;
            }
        </style>
    </head>
    <body>
        <!--
            innerText和innerHTML属性有什么区别？
                相同点：都是设置元素内部的内容。
                不同点：
                    innerHTML会把后面的“字符串”当做一段HTML代码解释并执行。
                    innerText，即使后面是一段HTML代码，也只是将其当做普通的字符串来看待。
        -->
        <script type="text/javascript">
            window.onload = function(){
                var btn = document.getElementById("btn");
                btn.onclick = function(){
                    // 设置div的内容
                    // 第一步:获取div对象
                    var divElt = document.getElementById("div1");
                    // 第二步:使用innerHTML属性来设置元素内部的内容
                    // divElt.innerHTML = "fjdkslajfkdlsajkfldsjaklfds";
                    divElt.innerHTML = "<font color='red'>用户名不能为空！</font>";//用户名不能为空！
                    // divElt.innerText = "<font color='red'>用户名不能为空！</font>";//<font color='red'>用户名不能为空！</font>
                }
            }
        </script>

        <input type="button" value="设置div中的内容" id="btn"/>
        <div id="div1"></div>
    </body>
</html>
```

2.2.3 正则表达式

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>DOM编程-关于正则表达式</title>
    </head>
    <body>
        <script type="text/javascript">
        /*
            1、什么是正则表达式，有什么用？
                正则表达式：Regular Expression
                正则表达式主要用在字符串格式匹配方面。
            2、正则表达式实际上是一门独立的学科，在Java语言中支持，C语言中也支持，javascript中也支持。
            大部分编程语言都支持正则表达式。正则表达式最初使用在医学方面，用来表示神经符号等。目前使用最多
            的是计算机编程领域，用作字符串格式匹配。包括搜索方面等。
            3、正则表达式，对于我们javascript编程来说，掌握哪些内容呢？
                第一：常见的正则表达式符号要认识。
                第二：简单的正则表达式要会写。
                第三：他人编写的正则表达式要能看懂。
                第四：在javascript当中，怎么创建正则表达式对象！（new对象）
                第五：在javascript当中，正则表达式对象有哪些方法！（调方法）
                第六：要能够快速的从网络上找到自己需要的正则表达式。并且测试其有效性。
            4、常见的正则表达式符号？
                .   匹配除换行符以外的任意字符
                \w  匹配字母或数字或下划线或汉字
                \s  匹配任意的空白符
```

```
                \d  匹配数字
                \b  匹配单词的开始或结束
                ^   匹配字符串的开始
                $   匹配字符串的结束

                *      重复零次或更多次
                +      重复一次或更多次
                ?      重复零次或一次
                {n}    重复n次
                {n,}   重复n次或更多次
                {n,m}  重复n到m次

                \W         匹配任意不是字母，数字，下划线，汉字的字符
                \S         匹配任意不是空白符的字符
                \D         匹配任意非数字的字符
                \B         匹配不是单词开头或结束的位置
                [^x]       匹配除了x以外的任意字符
                [^aeiou]   匹配除了aeiou这几个字母以外的任意字符

                正则表达式当中的小括号()优先级较高。
                [1-9]          表示1到9的任意1个数字（次数是1次。）
                [A-Za-z0-9]    表示A-Za-z0-9中的任意1个字符
                [A-Za-z0-9-]   表示A-Z、a-z、0-9、- ，以上所有字符中的任意1个字符。

                |     表示或者
            5、简单的正则表达式要会写
                QQ号的正则表达式：^[1-9][0-9]{4,}$
            6、他人编写的正则表达式要能看懂？
                email正则：^\w+([-+.]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$
            7、怎么创建正则表达式对象，怎么调用正则表达式对象的方法？
                第一种创建方式：
                    var regExp = /正则表达式/flags;
                第二种创建方式:使用内置支持类RegExp
                    var regExp = new RegExp("正则表达式","flags");

                关于flags：
                    g：全局匹配
                    i：忽略大小写
                    m：多行搜索（ES规范制定之后才支持m。）当前面是正则表达式的时候，m不能用。只有前面是普通字符串的时候，m才可以使用。

                正则表达式对象的test()方法？
                    true / false = 正则表达式对象.test(用户填写的字符串);
                    true : 字符串格式匹配成功
                    false: 字符串格式匹配失败
        */
            window.onload = function(){
                // 给按钮绑定click
                document.getElementById("btn").onclick = function(){
                    var email = document.getElementById("email").value;
                    var emailRegExp = /^\w+([-+.]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$/;
                    var ok = emailRegExp.test(email);
                    if(ok){
                        //合法
                        document.getElementById("emailError").innerText = "邮箱地址合法";
                    }else{
                        // 不合法
                        document.getElementById("emailError").innerText = "邮箱地址不合法";
                    }
                }
                // 给文本框绑定focus
                document.getElementById("email").onfocus = function(){
                    document.getElementById("emailError").innerText = "";
                }
            }
        </script>

        <input type="text" id="email" />
        <span id="emailError" style="color: red; font-size: 12px;"></span>
        <br>
        <input type="button" value="验证邮箱" id="btn" />
    </body>
</html>
```

2.2.4 扩展字符串的trim()

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>扩展字符串的trim()</title>
    </head>
    <body>
```

```
        <script type="text/javascript">
            // 低版本的IE浏览器不支持字符串的trim()函数,怎么办?
            // 可以自己对String类扩展一个全新的trim()函数!
            String.prototype.trim = function(){
                // alert("扩展之后的trim方法");
                // 去除当前字符串的前后空白
                // 在当前的方法中的this代表的就是当前字符串.
                //return this.replace(/^\s+/, "").replace(/\s+$/, "");
                return this.replace(/^\s+|\s+$/g, "");
            }

            window.onload = function(){
                document.getElementById("btn").onclick = function(){
                    // 获取用户名
                    var username = document.getElementById("username").value;
                    // 去除前后空白
                    username = username.trim();
                    // 测试
                    alert("--->" + username + "<----");
                }
            }
        </script>
        <input type="text" id="username" />
        <input type="button" value="获取用户名" id="btn" />
    </body>
</html>
```

2.2.5 表单验证

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>表单验证</title>
        <style type="text/css">
            span{
                color: red;
                font-size: 12px;
            }
        </style>
    </head>
    <body>
    <!--
    表单验证:
         (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
    -->
        <script type="text/javascript">
            window.onload = function(){
                //获取用户名以及用户名的span标签
                var usernameElt = document.getElementById("username");
                var usernameErrorSpan = document.getElementById("username_error");
                //给username这个文本框绑定失去焦点blur事件
                usernameElt.onblur = function(){
                    //获取用户名
                    var username = usernameElt.value;
                    //去除前后空白
                    username = username.trim();
                    //判断用户名是否为空
                    if(username === ""){
                        //用户名为空
                        usernameErrorSpan.innerText = "用户名不能为空";
                    }else{
                        //用户名不为空
                        //继续判断长度[6-14]
                        if(username.length < 6 || username.length > 14){
                            //用户名长度非法、
                            usernameErrorSpan.innerText = "用户长度必须在[6-14]之间";
                        }else{
                            //用户名长度合法
                            //必须判断是否含有特殊符号
                            var regExp = /^[A-Za-z0-9]+$/;
                            if(!regExp.test(username)){
                                //用户名中含有特殊符号
                                usernameErrorSpan.innerText = "用户名只能由数字和字母组成";
                            }else{
                                //用户名最终合法
```

```
                        }
                    }
                }
            }
            //给username这个文本框绑定获得焦点focus事件
            usernameElt.onfocus = function(){
                if(usernameErrorSpan.innerText != ""){
                    //清除用户名文本框不合法的value
                    usernameElt.value = "";
                }
                //清空usernameErrorspan
                usernameErrorSpan.innerText = "";
            }

            //获取确认密码框对象
            var userpwd2Elt = document.getElementById("userpwd2");
            //获取密码错误提示的span标签
            var userpwdErrorSpan = document.getElementById("pwd_error");
            //绑定blur事件
            userpwd2Elt.onblur = function(){
                //获取密码和确认密码
                var userpwdElt = document.getElementById("userpwd");
                var userpwd = userpwdElt.value;
                var userpwd2 = userpwd2Elt.value;
                if(userpwd != userpwd2){
                    //密码不一致
                    userpwdErrorSpan.innerText = "密码不一致";
                }else{
                    //密码一致
                }
            }
            //绑定focus事件
            userpwd2Elt.onfocus = function(){
                if(userpwdErrorSpan.innerText != ""){
                    userpwd2Elt.value = "";
                }
                userpwdErrorSpan.innerText = "";
            }

            //给email绑定blur事件
            var emailElt = document.getElementById("email");
            //获取email的span
            var emailErrorSpan = document.getElementById("email_error");
            emailElt.onblur = function(){
                //获取email
                var email = emailElt.value;
                //编写email的正则表达式
                var emailRegExp = /^\w+([-+.]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*$/;
                if(!emailRegExp.test(email)){
                    //email不合法
                    emailErrorSpan.innerText = "邮箱地址不合法";
                }else{
                    //email合法
                }
            }
            //给email绑定focus事件
            emailElt.onfocus = function(){
                if(emailErrorSpan.innerText != ""){
                    emailElt.value = "";
                }
                emailErrorSpan.innerText = "";
            }

            //给提交按钮绑定鼠标单击事件
            var submitBtnElt = document.getElementById("submitBtn");
            submitBtnElt.onclick = function(){
                // 触发username的blur、userpwd2的blur、email的blur
                // 不需要人工操作,使用纯JS代码触发事件.
                usernameElt.focus();
                usernameElt.blur();

                userpwd2Elt.focus();
                userpwd2Elt.blur();

                emailElt.focus();
                emailElt.blur();

                //当所有表单项都是合法的时候，提交表单
                if(usernameErrorSpan.innerText == "" && userpwdErrorSpan.innerText == "" &&
emailErrorSpan.innerText == ""){
                    //获取表单对象
                    var userFormElt = document.getElementById("userform");
```

```
                        //可以在这里设置action，也可以不在这里设置action
                        userFormElt.action = "http://localhost:8080/jd/save";
                        //提交表单
                        userFormElt.submit();
                    }
                }

            }
        </script>

        <!--这个表单提交应该使用post，这里为了检测，所以使用get。-->
        <!-- <form id="userForm" action="http://localhost:8080/jd/save" method="get"> -->
        <form id="userform" method="get">
            用户名<input type="text" name="username" id="username" />
            <span id="username_error"></span>
            <br>

            密码<input type="password" name="userpwd" id="userpwd"/><br>
            确认密码<input type="password" id="userpwd2" />
            <span id="pwd_error"></span>
            <br>

            邮箱<input type="text" name="email" id="email" />
            <span id="email_error"></span>
            <br><br>

            <!-- <input type="submit" value="注册" /> -->
            <input type="button" value="注册" id="submitBtn"/>    
            <input type="reset" src="重置" />
        </form>
    </body>
</html>
```

2.2.6 复选框的全选和取消全选

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>复选框的全选和取消全选</title>
    </head>
    <body>
        <script type="text/javascript">
            window.onload = function(){
                var aihaos = document.getElementsByName("aihao");
                var firstChk = document.getElementById("firstChk");

                firstChk.onclick = function(){
                    for(var i = 0; i < aihaos.length; i++){
                        aihaos[i].checked = firstChk.checked;
                    }
                }

                // 对以上数组进行遍历
                var all = aihaos.length;
                for(var i = 0; i < aihaos.length; i++){
                    aihaos[i].onclick = function(){
                        var checkedCount = 0;
                        // 总数量和选中的数量相等的时候,第一个复选框选中.
                        for(var i = 0; i < aihaos.length; i++){
                            if(aihaos[i].checked){
                                checkedCount++;
                            }
                        }
                        firstChk.checked = (all == checkedCount);
                        /*
                        if(all == checkedCount){
                            firstChk.checked = true;
                        }else{
                            firstChk.checked = false;
                        }
                        */
                    }
                }
            }
        </script>
        <input type="checkbox" id="firstChk"/><Br>
        <input type="checkbox" name="aihao" value="movie" />电影<Br>
        <input type="checkbox" name="aihao" value="comic" />动漫<Br>
        <input type="checkbox" name="aihao" value="tv" />剧集<Br>
    </body>
</html>
```

2.2.7 获取下拉列表选中项的value

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>获取下拉列表选中项的value</title>
    </head>
    <body>
        <script type="text/javascript">
            window.onload = function(){
                var provinceListElt = document.getElementById("provinceList");
                provinceListElt.onchange = function(){
                    // 获取选中项的values
                    alert(provinceListElt.value);
                }
            }
        </script>
        <select id="provinceList">
            <option value ="">--请选择省份--</option>
            <option value ="001">河北省</option>
            <option value ="002">河南省</option>
            <option value ="003">山东省</option>
            <option value ="004">山西省</option>
        </select>
    </body>
</html>

<!--
省份和市区的关系是：1对多
省份表t_province
id      pcode      pname
----------------------------
1       001        河北省
2       002        河南省
3       003        山东省
4       004        山西省

市区表t_city
id      ccode      cname       pcode(fk)
-------------------------------------------
1       101        石家庄         001
2       102        保定           001
3       103        邢台           001
4       104        承德           001
5       105        张家口         001
6       106        邯郸           001
7       107        衡水           001

前端用户选择的假设是河北省，那么必须获取到河北省的pcode，获取到001
然后将001发送提交给服务器，服务器底层执行一条SQL语句：
    select * from t_city where pcode = '001';

    返回一个List集合，List<City> cityList;

    cityList响应浏览器，浏览器在解析cityList集合转换成一个新的下拉列表。
-->
```

2.2.8 显示网页时钟

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>显示网页时钟</title>
    </head>
    <body>
        <script type="text/javascript">
            /* 关于JS中内置的支持类：Date，可以用来获取时间/日期。*/
            // 获取系统当前时间
            var nowTime = new Date();
            // 输出
            document.write(nowTime);//Fri Dec 10 2021 19:43:35 GMT+0800 (中国标准时间)

            // 转换成具有本地语言环境的日期格式.
            nowTime = nowTime.toLocaleString();
            document.write(nowTime);//2021/12/10 下午7:43:35
            document.write("<br>");
            document.write("<br>");

            // 当以上格式不是自己想要的,可以通过日期获取年月日等信息,自定制日期格式.
            var t = new Date();
            var year = t.getFullYear(); // 返回年信息,以全格式返回.
```

```
            var month = t.getMonth(); // 月份是:0-11
            // var dayOfWeek = t.getDay(); // 获取的一周的第几天(0-6)
            var day = t.getDate(); // 获取日信息.
            document.write(year + "年" + (month+1) + "月" + day + "日");//2021年12月10日
            document.write("<br>");
            document.write("<br>");

            // 重点:怎么获取毫秒数?(从1970年1月1日 00:00:00 000到当前系统时间的总毫秒数)
            //var times = t.getTime();
            //document.write(times); // 一般会使用毫秒数当做时间戳. (timestamp)
            document.write(new Date().getTime());//1639136615761
        </script>

        <script type="text/javascript">
            function displayTime(){
                var time = new Date();
                time = time.toLocaleString();
                document.getElementById("timeDiv").innerHTML = time;
            }

            // 每隔1秒调用displayTime()函数
            function start(){
                // 从这行代码执行结束开始,则会不间断的,每隔1000毫秒调用一次displayTime()函数.
                v = window.setInterval("displayTime()", 1000);
            }
            function stop(){
                window.clearInterval(v);
                document.getElementById("timeDiv").innerText = "";
            }
        </script>
        <br><br>
        <input type="button" value="显示系统时间" onclick="start();"/>
        <input type="button" value="系统时间停止" onclick="stop();" />
        <div id="timeDiv"></div>
    </body>
</html>
```

2.2.9 内置支持类Array

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>内置支持类Array</title>
    </head>
    <body>
        <script type="text/javascript">
            // 创建长度为0的数组
            var arr = [];
            alert(arr.length);//0

            // 数据类型随意
            var arr2 = [1,2,3,false,"abc",3.14];
            alert(arr2.length);//6

            // 下标会越界吗
            // arr2 = [1,2,3,false,"abc",3.14,undefined,"test"]
            arr2[7] = "test"; // 自动扩容.

            document.write("<br>");
            // 遍历
            for(var i = 0; i < arr2.length; i++){
                document.write(arr2[i] + "<br>");
            }

            // 另一种创建数组的对象的方式
            var a = new Array();
            alert(a.length); // 0

            var a2 = new Array(3); // 3表示长度.
            alert(a2.length); // 3

            var a3 = new Array(3,2);//多个参数表示存入数组的元素
            alert(a3.length); // 2

            var a = [1,2,3,9];
            var str = a.join("-");
            alert(str); // 1-2-3-9

            // 在数组的末尾添加一个元素(数组长度+1)
            a.push(10);
            alert(a.join("-")); // 1-2-3-9-10

```

```
            // 将数组末尾的元素弹出(数组长度-1)
            var endElt = a.pop();
            alert(endElt);//10
            alert(a.join("-"));//1-2-3-9

            // 注意:JS中的数组可以自动模拟栈数据结构:后进先出,先进后出原则.
            // push压栈
            // pop弹栈

            // 反转数组.
            a.reverse();
            alert(a.join("="));// 9=3=2=1
        </script>
    </body>
</html>
```

2.3 BOM编程

2.3.1 open和close

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>BOM编程-open和close</title>
    </head>
    <body>
        <script type="text/javascript">
            /*
                1、BOM编程中，window对象是顶级对象，代表浏览器窗口。
                2、window有open和close方法，可以开启窗口和关闭窗口。
            */
        </script>

        <input type="button" value="开启百度(新窗口)" onclick="window.open('http://www.baidu.com');" />
        <input type="button" value="开启百度(当前窗口)" onclick="window.open('http://www.baidu.com',
'_self');" />
        <input type="button" value="开启百度(新窗口)" onclick="window.open('http://www.baidu.com',
'_blank');" />
        <input type="button" value="开启百度(父窗口)" onclick="window.open('http://www.baidu.com',
'_parent');" />
        <input type="button" value="开启百度(顶级窗口)" onclick="window.open('http://www.baidu.com',
'_top');" />

        <input type="button" value="打开新窗口"  onclick="window.open('02.html')"/>
    </body>
</html>
```

```
<!--02.html-->
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title></title>
    </head>
    <body>
        <input type="button" value="关闭当前窗口" onclick="window.close();" />
    </body>
</html>
```

2.3.2 弹出消息框和确认框

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>弹出消息框和确认框</title>
    </head>
    <body>
        <script type="text/javascript">
            function del(){
                /*
                var ok = window.confirm("亲，确认删除数据吗？");
                if(ok){
                    alert("delete data ....");
                }
                */
                if(window.confirm("亲，确认删除数据吗？")){
                    alert("delete data ....");
                }
            }
        </script>
```

```
        <input type="button" value="弹出消息框" onclick="window.alert('消息框!')" />
        <!--删除操作的时候都要提前先得到用户的确认。-->
        <input type="button" value="弹出确认框(删除)" onclick="del();" />
    </body>
</html>
```

2.3.3 当前窗口设置为顶级窗口

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>当前窗口设置为顶级窗口</title>
        <!--窗体-->
        <!--
        <frameset cols="30%,*">
            <frame src="http://www.baidu.com" />
            <frame src="005-child-window.html" />
        </frameset>
        -->
    </head>
    <body>
        <!--在当前窗口中隐藏的内部窗体。-->
        <!-- <iframe src="http://www.baidu.com"></iframe> -->
        <iframe src="05.html"></iframe>
    </body>
</html>
```

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>05.html</title>
    </head>
    <body>
        child window.
        <script type="text/javascript">
            window.onload = function(){
                var btn = document.getElementById("btn");
                btn.onclick = function(){
                    if(window.top != window.self){
                        //window.top = window.self;
                        window.top.location = window.self.location;
                    }
                }
            }
        </script>
        <input type="button" value="将当前窗口设置为顶级窗口" id="btn" />
    </body>
</html>
```

2.3.4 历史记录-history

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title></title>
    </head>
    <body>
        <a href="07.html">07.html页面</a>
        <input type="button" value="前进" onclick="window.history.go(1)"/>
    </body>
</html>
```

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>07.html</title>
    </head>
    <body>
        07 page!
        <input type="button" value="后退" onclick="window.history.back()" />
        <input type="button" value="后退" onclick="window.history.go(-1)" />
    </body>
</html>
```

2.3.5 设置浏览器地址栏上的URL

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>设置浏览器地址栏上的URL</title>
    </head>
    <body>
        <script type="text/javascript">
            function goBaidu(){
                //var locationObj = window.location;
                //locationObj.href = "http://www.baidu.com";

                // window.location.href = "http://www.jd.com";
                // window.location = "http://www.126.com";

                //document.location.href = "http://www.sina.com.cn";
                document.location = "http://www.tmall.com";
            }
        </script>

        <input type="button" value="天猫" onclick="goBaidu();"/>
        <input type="button" value="baidu" onclick="window.open('http://www.baidu.com');" />
    </body>
</html>

<!--
总结，有哪些方法可以通过浏览器往服务器发请求？
1、表单form的提交。
2、超链接。<a href="http://localhost:8080/oa/save?username=zhangsan&password=123">用户只能点击这个超链接</a>
3、document.location
4、window.location
5、window.open("url")
6、直接在浏览器地址栏上输入URL，然后回车。（这个也可以手动输入，提交数据也可以成为动态的。）

以上所有的请求方式均可以携带数据给服务器，只有通过表单提交的数据才是动态的。
-->
```

2.4 JSON

2.4.1 JSON概述

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JSON</title>
    </head>
    <body>
        <script type="text/javascript">
            /*
                1、什么是JSON，有什么用？
                    JavaScript Object Notation（JavaScript对象标记），简称JSON。(数据交换格式)
                    JSON主要的作用是：一种标准的数据交换格式。（目前非常流行，90%以上的系统，系统A与系统B交换数据的话，都是采用JSON。）
                2、JSON是一种标准的轻量级的数据交换格式。特点是：
                    体积小，易解析。
                3、在实际的开发中有两种数据交换格式，使用最多，其一是JSON，另一个是XML。
                    XML体积较大，解析麻烦，但是有其优点是：语法严谨。（通常银行相关的系统之间进行数据交换的话会使用XML。）
                4、JSON的语法格式：
                    var jsonObj = {
                        "属性名" : 属性值,
                        "属性名" : 属性值,
                        "属性名" : 属性值,
                        "属性名" : 属性值,
                        ....
                    };
                5、XML的语法格式：
                <?xml version="1.0" encoding="GBK"?>
                <!--
                    HTML和XML有一个父亲：SGML（标准通用的标记语言。）
                    HTML主要做页面展示：所以语法松散。很随意。
                    XML主要做数据存储和数据描述的：所以语法相当严格。
                -->
                <students>
                    <student sno="110">
                        <sname>张三</sname>
                        <sex>男</sex>
                    </student>
                    <student sno="120">
                        <sname>李四</sname>
                        <sex>男</sex>
```

```
                    </student>
                    <student sno="130">
                        <sname>王五</sname>
                        <sex>男</sex>
                    </student>
                </students>
            */
            // 创建JSON对象(JSON也可以称为无类型对象。轻量级，轻巧。体积小。易解析。)
            var studentObj = {
                "sno" : "110",
                "sname" : "张三",
                "sex" : "男"
            };
            // 访问JSON对象的属性
            alert(studentObj.sno + "," + studentObj.sname + "," + studentObj.sex);

            // 之前没有使用JSON的时候,定义类,创建对象,访问对象的属性.
            Student = function(sno,sname,sex){
                this.sno = sno;
                this.sname = sname;
                this.sex = sex;
            }
            var stu = new Student("111","李四","男");
            alert(stu.sno + "," + stu.sname + "," + stu.sex);

            // JSON数组
            var students = [
                {"sno":"110","sname":"张三","sex":"男"},
                {"sno":"120","sname":"李四","sex":"男"},
                {"sno":"130","sname":"王五","sex":"男"}
            ];
            // 遍历
            for(var i = 0; i < students.length; i++){
                var stuObj = students[i];
                alert(stuObj.sno + "," + stuObj.sname + "," + stuObj.sex);
            }
        </script>
    </body>
</html>
```

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>复杂一些的JSON对象</title>
    </head>
    <body>
        <script type="text/javascript">
            var user = {
                "usercode" : 110,
                "username" : "张三",
                "sex" : true,
                "address" : {
                    "city" : "北京",
                    "street" : "大兴区",
                    "zipcode" : "12212121",
                },
                "aihao" : ["movie","comic","tv"]
            };
            // 访问人名以及居住的城市
            alert(user.username + "居住在" + user.address.city);

            //请自行设计JSON格式的数据，这个JSON格式的数据可以描述整个班级中每一个学生的信息，以及总人数信息。
            var jsonData = {
                "total": 3,
                "students": [
                    {"name":"zhangsan","birth":"1980-10-20"},
                    {"name":"lisi","birth":"1981-10-20"},
                    {"name":"wangwu","birth":"1982-10-20"}
                ]
            };
        </script>
    </body>
</html>
```

2.4.2 eval函数

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>eval函数</title>
```

```
    </head>
    <body>
        <!--
            JSON是一种行业内的数据交换格式标准。
            JSON在JS中以JS对象的形式存在。
        -->
        <script type="text/javascript">
            /*
            eval函数的作用是：
                将字符串当做一段JS代码解释并执行。
            */
           window.eval("var i = 100;");//相当于i = 100;
           alert("i = " + i); // i = 100

           // java连接数据库,查询数据之后,将数据在java程序中拼接成JSON格式的“字符串”,将json格式的字符串响应到浏览器
           // 也就是说java响应到浏览器上的仅仅是一个"JSON格式的字符串",还不是一个json对象.
           // 可以使用eval函数,将json格式的字符串转换成json对象.
           var fromJava = "{\"name\":\"zhangsan\",\"password\":\"123\"}"; //这是java程序给发过来的json格式的"字符串"
           // 将以上的json格式的字符串转换成json对象
           window.eval("var jsonObj = " + fromJava);
           // 访问json对象
           alert(jsonObj.name + "," + jsonObj.password); // 在前端取数据.

           /*
            面试题：
                在JS当中：[]和{}有什么区别？
                    [] 是数组。
                    {} 是JSON。

                java中的数组：int[] arr = {1,2,3,4,5};
                JS中的数组：var arr = [1,2,3,4,5];
                JSON：var jsonObj = {"email" : "zhangsan@123.com","age":25};
           */

           var json = {
               "username" : "zhangsan"
           };
           // JS中访问json对象的属性
           alert(json.username);
           // JS中访问json对象的属性
           alert(json["username"]);
        </script>
    </body>
</html>
```

2.4.3 设置table的tbody

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>设置table的tbody</title>
    </head>
    <body>
        <script type="text/javascript">
            // 有这些json数据
            var data = {
                "total" : 4,
                "emps" : [
                    {"empno":7369,"ename":"SMITH","sal":800.0},
                    {"empno":7361,"ename":"SMITH2","sal":1800.0},
                    {"empno":7360,"ename":"SMITH3","sal":2800.0},
                    {"empno":7362,"ename":"SMITH4","sal":3800.0}
                ]
            };

            // 希望把数据展示到table当中.
            window.onload = function(){
                var displayBtnElt = document.getElementById("displayBtn");
                displayBtnElt.onclick = function(){
                    var emps = data.emps;
                    var html = "";
                    for(var i = 0; i < emps.length; i++){
                        var emp = emps[i];
                        html += "<tr>";
                        html += "<td>"+emp.empno+"</td>";
                        html += "<td>"+emp.ename+"</td>";
                        html += "<td>"+emp.sal+"</td>";
                        html += "</tr>";
                    }
                    document.getElementById("emptbody").innerHTML = html;
                    document.getElementById("count").innerHTML = data.total;
```

```
                }
            }
        </script>

        <input type="button" value="显示员工信息列表" id="displayBtn" />
        <h2>员工信息列表</h2>
        <hr>
        <table border="1px" width="50%">
            <tr>
                <th>员工编号</th>
                <th>员工名字</th>
                <th>员工薪资</th>
            </tr>
            <tbody id="emptbody">
                <!--
                <tr>
                    <td>7369</td>
                    <td>SMITH</td>
                    <td>800</td>
                </tr>
                <tr>
                    <td>7369</td>
                    <td>SMITH</td>
                    <td>800</td>
                </tr>
                <tr>
                    <td>7369</td>
                    <td>SMITH</td>
                    <td>800</td>
                </tr>
                -->
            </tbody>
        </table>
        总共<span id="count">0</span>条数
    </body>
</html>
```