web技术

课外学习

TypeScript入门
Vue3选项式API+TS

课本内容

第一章 Web前端开发技术综述

1.1 Web概述

1.1.1 web的特点

1. 易导航和图形化的界面：Web之前的因特网上的信息只有文本形式，Web具有可以将图形、音频、视频等集于一体的特性；此外Web导航非常方便，只需从一个链接跳转到另一个链接
2. 与平台无关性：无论计算机系统是什不同的站点上都可以通过因特网访问WWW
3. 分布式结构：对于Web来说，信息可以放在不同的站点上
4. 动态性：各Web站点的信息包含站点本身的信息，信息的提供者可以经常对站上的信息进行更新和维护
5. 交互性：
    - 超链接使得用户的浏览顺序和所访问的站点完全由用户决定
    - 用户可以通过表单Form的形式可以从服务器获得动态的信息

1.1.3 Web工作原理

用户通过客户端浏览器访问到因特网上的网站或者其他网络资源时，通常经历以下过程：

1. 在客户端的浏览器的地址栏中输入需要访问网站的统一资源定位符(URL,Uniform Resource Locator)
2. 通过域名服务器进行全球域名解析，并根据解析结果决定访问指定IP地址的网站或网页
3. 客户端获得IP地址后，向指定IP地址上的Web服务器发送一个HTTP(Hypertext Transfer Protocal,超文本传输协议)请求

4. Web服务器通常情况下会快速地响应客户端的请求，将客户需要的HTML文本，图片和构成网页的一切文件发送回用户
   此外，如果需要访问数据库系统中的数据时，Web服务器会将控制权转让给应用服务器，根据Web服务器的数据请求读写数据库，并进行相关数据库的访问操作，应用服务器将数据查询响应发送给Web服务器，由Web服务器将查询结果再传发给客户端的浏览器

大多数网页中包含很多超链接，有内链接和外链接。通过超链接可以设置资源下载、页面浏览及链接其他网络资源，通过超链接，将有用的资源组织在一起，就形成了一个所谓的信息的"网"

1.2 Web前端开发技术

1.2.1 HTML

HTML(Hypertext Markup Language)超文本标记语言，是一种标记语言，而非编程语言
HTML是Web页面的结构，使用标记来描述网页，网页内容包括：

- 标题
- 副标题
- 段落
- 无序列表
- 定义列表
- 表格
- 表单
  SGML(Standard Generalized Markup Language,标准通用标记语言)
  一种定义电子文档结构和描述其内容的国际标准语言，是所有电子文档标记语言的起源
  HTML是SGML的一个应用(一个子集)
  Web浏览器可以读取HTML文档，并以网页的形式显示出它们

1.2.2 CSS

级联样式表(Cascading Style Sheet,CSS)，也称为层叠样式表
在设计Web页面时，采用CSS技术，可以有效地对页面的布局，字体，颜色，背景和其他效果实现更加精密的控制
解决了网页内容与表现分离的问题
其本质上是一种标记语言，不需要编译，可以有浏览器直接解释执行，属于**浏览器解释语言**

1.2.3 JavaScript

JavaScript的概念
JS是一种基于对象和事件驱动，具有相对安全性的客户端脚本语言，也是一种广泛用于客户端Web开发的脚本语言，常用于给HTML网页添加动态的功能，例如响应用户的各种操作
JavaScript的组成

- 核心(ECMAScript)
- 文档对象模型(Document Object Model,DOM)
- 浏览器对象模型(Browser Object Model,BOM)

1.2.4 HTML DOM

HTML DOM(Document Object Model) 是HTML文档对象模型的缩写
DOM是一种与浏览器、平台语言无关的接口，使得用户可以访问页面上的其他标准组件
DOM和JavaScript结合起来实现了Web页面的行为与结构的分离

1.3.5 BOM

BOM(Browser Object Model) 浏览器对象模型，定义了JS可以及进行操作的浏览器的各个功能部件的接口，提供访问文档各个功能部件(窗口、屏幕部件功能、浏览历史记录)的途径和操作方法

1.3.6 AJAX

AJAX (Asynchronous JavaScript and XML) 异步JavaScript和XML
是多种技术的综合，使用XHTML和CSS标准化呈现，使用DOM实现动态显示和交互，使用XML和XSTL进行数据交换与处理，使用XMLHttpRequest对象进行异步数据读取，使用JavaScript绑定和处理所有数据
AJAX可以部分更新页面内容

1.3.7 jQuery

跨浏览器的JavaScript库，简化HTML与JavaScript之间的操作

第二章 HTML基础

2.1 HTML文档结构

2.1.1 基本结构

HTML文档由头部head和主体body两个部分组成
头部<head>标记中，可定义标题、样式
主体<body>标记中，可定义段落、标题字、超链接、脚本、表格、表单等元素，主体内容是网页要显示的信息
HTML文档的头部标记主要包含页面标题标记、元信息标记、样式标记、脚本标记、链接标记等,一般不显示在页面上

2.2.2 元信息<meta>

META标记用来描述一个HTML网页文档的属性，也称为元信息(meta-information),这些信息不会显示在浏览器的页面中，包括:

- 作者
- 日期和时间
- 网页描述
- 关键词
- 页面刷新
  该标记位于文档的头部(其属性形式是“名称/值”对)

<meta>标记

```
<meta name="" content="">
<meta http-equiv="" content="">
```

name属性用于描述网页,name属性的值所描述的内容(值)通过content属性表示
便于搜索引擎机器人查找、分类
其中最重要的是description、keywords和robots

http-equiv属性用于提供HTTP协议的响应头报文,http-equiv属性的值所描述的内容(值)通过content属性表示，通常为网页加载前提供给浏览器等设备使用

- content-type charset提供编码信息
- refresh刷新与跳转页面
- no-cache页面缓存

- expires网页缓存过期时间

属性	值	描述
content	some_text	定义与http-equiv或name属性相关的元信息
http-equiv	content-type	内容类型
	expires	网页缓存过期时间
	refresh	刷新与跳转(重定向)页面
	set-cookie	如果网页过期，那么存盘的cookie将被删除
name	author	定义网页作者
	description	定义网页简短描述
	keywords	定义网页关键词
	generator	定义编辑器
scheme	some_text	定义用于翻译content属性值的格式。

```
<meta name="keywords" content="信息参数" />
<meta name="description" content="信息参数" />
<meta http-equiv="content-type" content="text/html; charset=信息参数" />
<meta name="generator" content="信息参数" />
<meta name="author" content="信息参数">
<meta http-equiv="refresh" content="时间;  url=网址参数">
<meta name="robots" contect="信息参数">
```

2.3 主体body

主体body是一个Web页面的主要部分，其设置内容是读者实际看到的信息(图片、图像、表格、文字、超链接等元素)

2.3.1 body标记

```
<body>…      </body>
```

＜body＞是开始标记，＜/body＞是结束标记。两者之间所包括的内容为网页上显示的信息。

2.3.2 body标记属性

设置body标记属性可以改变Web页面显示效果
body标记主要属性如下

属性	值	描述
text	• rgb(R,G,B) grb(R%,G%,B%) • #RRGGBB |#RGB • Colorname	规定文档中所有文本的颜色。 不赞成使用。请使用样式取代它。
bgcolor	• 同上	规定文档的背景颜色。不赞成使用。
alink	• 同上	规定文档中活动链接的颜色。
link	• 同上	规定文档中未访问链接的默认颜色。
vlink	• 同上	规定文档中已被访问链接的颜色。
background	URL	规定文档的背景图像。
topmargin	• Pixel	规定文档中上边距的大小
leftmargin	• pixel	规定文档中左边距的大小

示例

```
<body leftmargin="50px" topmargin="50px"
          text="#000000" bgcolor="#339999"
           link="blue" alink="white" vlink="red"
           background="body_image.jpg">
```

颜色表示方法

- RGB（R，G，B），其中R、G、B为是0～255的整数
- RGB（R%，G%，B%），其中R、G、B为0～100的整数
- 3位或6位十六进制数#RGB或#RRGGBB，R、G、B为十六进制数，取值范围：0～9、A～F
- #RGB可以转换为#RRGGBB
- 例如红色分别表示为#F00、#FF0000

2.4 HTML基本语法

2.4.1 标记语法

HTML标记是由尖括号包围的关键词，用于说明指定内容的外貌和特征，也称为标签（Tag）
<html> <head> <body>等都是标记。标记通常分为单个标记和成对标记两种类型
单个标记
<标记名称>或<标记名称/>

最常用的单标记有
、<hr>。
、
表示换行，<hr>、

成对标记
成对标记由开始标记和结束标记两部分组成，必须成对使用。开始标记也称为首标记，告诉Web浏览器从此处开始执行该标记所表示的功能；结束标记也称为尾标记，告诉Web浏览器在这里结束该标记。
<标记名称>内容</标记名称>

2.4.2 属性语法

<标记名称 属性1="属性值1" 属性2="属性值2" … 属性n="属性值n">
属性应在开始标记（首标记）内定义，并且和标记名之间有一个空格分隔。
属性值可以直接书写，也可以使用双引号""括起来

2.5 注释

两种注释方法:

- <!-- 注释信息 -->
- <comment>注释信息</comment>

comment标记是成对标记，以<comment>开始，以</comment>结束
标记包围的信息为注释内容,但在高版本的浏览器中均显示在页面上，建议不使用此注释标记

2.6 HTML文档编写规范

2.6.1 HTML页面编码基本规范

1. 所有标记均以“<”开始、以“>”结束。
2. 根据标记类型，正确输入标记，单个标记最好在右尖括号前加1个斜杠“/”，如换行标记是单标记
   ，成对标记最好同时输入起始标记和结束标记，以免忘记。
3. 标记可以嵌套使用，但不能交叉使用。
4. 在HTML代码中不区分大小写。
5. 标记中可以设置各种属性，属性值建议用双引号标注起来

6. 书写开始与结束标记时，在左尖括号与标记名或与斜杠“/”之间不能留有多余空格，否则浏览器标记不能识别，导致错误标记直接显示在页面上，影响页面美观效果。
7. 编写HTML代码时，应该使用锯齿结构，即采用缩进风格，使代码结构清晰，便于理解和分析页面的结构，便于代码后期阅读和维护。

2.6.2 HTML文档命名规范

1. 文档的扩展名为html或者htm，建议统一用html作为文件名的后缀。
2. 文档名中只可由英文字母、数字或下划线组成，建议以字母或下划线开始。
3. 文档名中不能包含特殊符号，如空格、$、&等。
4. 文档名区分大小写。
5. Web服务器主页一般是index.html或default.html

2.7 HTML文档类型

2.7.1 <!DOCTYPE>标记

```
<!DOCTYPE element-name DTD-type DTD-name DTD-url>
```

<!DOCTYPE >表示开始声明DTD（Document Type Definition文档类型定义），其中DOCTYPE是关键字。
element-name指定该DTD的根元素名称。
DTD-type指定该DTD是属于标准公用的还是私人制定的。设置为PUBLIC则表示该DTD是标准公用的，设置为SYSTEM则表示私人制定的。
DTD-name指定该DTD的文件名称。
DTD-url指定该DTD文件所在的URL地址。
>是指结束DTD的声明。

2.7.2 DTD类型

HTML 4.01 规定了三种DTD类型：严格Strict、过渡Transitional以及框架Frameset。

- <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
- <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
- <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN " "http://www.w3.org/TR/html4/frameset.dtd">
- HTML5的DTD定义：<!doctype html>

第三章 格式化文字与段落

3.1 Web页面初步设计

Web页面设计原则

- 简洁
  满足人们的实用和需求为目标，要求简练，准确。
- 一致性
  网站中各个页面使用相同的页边距，页面中的每个元素与整个页面以及站点的色彩和风格上的一致性。
- 好的对比度
  对比度的目的强调突出关键内容，以吸引浏览者，鼓励他们去发掘更深层次的内容。

3.1.1 向Web页面添加文件信息

```
<body>向这里添加内容</body>
```

- body 元素定义文档的主体。
- body 元素包含文档的所有内容（比如文本、超链接、图像、表格和列表等）。
- body标记所包含的内容会显示在页面上。

3.1.2标题字标记

标题是通过 <h1> -<h6> 等标记进行定义的

```
<h1 align=“left|center|right| justify " >1号标题字</h1>
```

3.1.3 添加空格与特殊符号

在HTML文件中，添加空格的方式与其他文档添加空格的方式不同，网页中通过代码控制来添加空格

```
<body>
        &lt;&reg;&times;
</body>
```

在body标记内无论输入多少空格、回车符，页面在显示时都会忽略。

在HTML文件中，插入特殊字符（与空格相同）。

显示结果	说明	Entity Name	Entity Number
	显示一个空格		
<	小于	<	<
>	大于	>	>
&	&符号	&	&
"	双引号	"	"
©	版权	©	©
®	注册商标	®	®
×	乘号	×	×
÷	除号	÷	÷

3.2 格式化文本标记

3.2.1 文本修饰标

标记	说明
软件工程专业！	定义粗体
软件工程专业！	定义斜体
<u>软件工程专业！</u>	定义下划线
~~软件工程专业！~~	定义删除线
^{软件工程专业！}	定义上标
_{软件工程专业！}	定义下标
软件工程专业！	定义着重文字，与效果相同
软件工程专业！	定义加重语气，与效果相同
软件工程专业！	变小字号
<big>软件工程专业！</big>	变大字号

3.2.2 计算机输出标记

标签	说明
<code></code>	定义计算机代码
<kbd></kbd>	定义键盘码
<samp></samp>	定义计算机代码样本
<tt></tt>	定义打字机代码
<var></var>	定义变量
<pre></pre>	定义预格式文本

3.2.3 引用和术语标记

标记	主要用途
<abbr>etc.</abbr>	定义缩写
<address>江苏南京市</address>	定义地址
<blockquote>长的引用</blockquote>	定义长的引用
<cite>引用、引证</cite>	定义引用、引证
<q>引用短语</q>	定义短的引用语，IE看不到引号，其余可以
<dfn>定义项目</dfn>	定义一个定义项目

3.2.4 字体font标记

font标记素用来改变默认的字体、颜色、大小等属性，这些更改分别通过不同的属性定义完成

```
<font  face=""  size=""  color="" ></font>
```

属性	值	说明
size	+1～+7，-1～-7	数字越大字号越大。
color	•rgb(r,g,b) grb(r%,g%,b%) •#rrggbb |#rgb colorname	规定文本的颜色。可以使用rgb函数、十六进制数、颜色英文名称来表达。
face	字体1，字体2，…，字体n	face属性可以有多个值，用逗号分隔。字体使用方式为从左向右依次选用。只要前面的字体不存在，则使用后一个字体。都不存在，使用默认"宋体"

3.3 段落与排版标记

3.3.1 段落标记<p>

段落<p>标记用来起始一个段落，它是一个块级元素 。段落p标记会自动在其前后创建一些空白

```
<p align="left|center|right">段落正文内容</p>
```

3.3.2 换行标记br

 换行
文档中，
元素属于单个标志，表示插入换行符

3.3.3 水平分割标记<hr>

```
<hr size=""  color=""   width=""  align=""    >
```

属性	值	说明
width	像素px或百分比	设置水平线宽度。
size	整数，单位px	设置水平线高度
color	rgb函数、十六进制数，颜色英文名称	设置水平线颜色
align	left center right	设置水平线对齐方式

3.3.4 拼音/音标注释ruby标记和rt/rp标记

```
<ruby>
      中<rt><rp>(</rp>zhong<rp>)</rp></rt>
      国<rt><rp>(</rp>guo<rp>)</rp></rt>
</ruby>
```

ruby 标记定义ruby注释（中文注音或字符）。ruby标记与rt标记一同使用
ruby标记由一个或多个字符（需要一个解释/发音）和一个提供该信息的rt 标记组成，还包括可选的rp标记，定义当浏览器不支持ruby 标记时显示的内容
注意有些浏览器当定义rp标记后，将不起效果

3.3.5 段落缩进标记<blockquote>

```
<body>
        <blockquote>需要缩进的内容</blockquote>
</body>
```

段落缩进(也称为“块引用”)blockquote标记引用的内容必须是块级标记，浏览器在blockquote标记前后添加了换行，并增加了外边距。
一对blockquote标记能够向右缩进5个西文字符的位置

3.3.6 预格式化标记<pre>

<pre></pre>标记对网页中的文字段落进行预格式化，浏览器会完整保留设计者在源文件中所定义的格式，包括各种空格、缩进以及其他特殊格式

```
<pre>预格式化文本 </pre >
 <pre>
        春 晓
              孟浩然
        春眠不觉晓，
                  处处闻啼鸟。
        夜来风雨声，
                  花落知多少。
  </pre>
```

第四章 列表

4.1 列表简介

列表能对网页中的相关信息进行合理的布局，将项目有序或无序地罗列在一起，从而方便用户浏览和操作
HTML中列表一共有5种，分别是无序列表、有序列表、定义列表、菜单列表和目录列表。常用的列表有无序列表、有序列表、定义列表等3种

4.2 无序列表

在HTML文件中插入成对的标记
，完成无序列表的插入
列表项(List Items)li标记用于定义 一个列表项

```
<ul type=“disc | circle | square">
        <li type="">项目名称</li>
        <li type="">项目名称</li>
        <li type="">项目名称</li>
        …
</ul>
```

> disc - ●； circle -○； square -■

4.3 有序列表

有序列表(Ordered List)ol标记是成对标记
在HTML文件中插入成对的标记
，完成有序列表的插入

4.4 列表嵌套

在HTML文件中，列表嵌套使用不仅使网页的内容布局更加美观，而且使显示的内容更加清晰、明白

```
<ul>
       <li >项目名称
         <ol>
             <li >项目名称 </li>
              …
         </ol>
   </li>
    …
</ul>
```

4.5 定义列表

定义列表(Definition List)dl标记是成对标记。在HTML文件中插入成对的标记<dl></dl>，完成定义列表的插入。

```
<dl>
        <dt>项目1</dt>
             <dd>说明1<dd>
             <dd>说明2<dd>
        …
</dl>
dt标记的由来:definition term
dd标记的由来: definition description
```

第五章 超链接

5.1 超链接概述

超链接是指从**一个网页**指向**一个目标**的连接关系

> 目标可以是：一个网页、图片、一个电子邮件地址、一个文件或是一个应用程序

网页中超链接的对象是一段文本或者是一个图片
超链接在本质上属于一个网页的一部分，它是一种允许我们同其他网页或站点之间进行连接的元素

5.2 超链接语法、路径及分类

5.2.1 超链接a(Anchor 锚)标记语法

```
<a href="url"  name="" title="" target="">超链接标题</a>
```

1. href （href-Hypertext reference）：链接指向的目标文件
2. title：指向链接的提示信息
3. target：指定打开的目标窗口
    - _parent -上一级窗口；
    - _blank-新窗口；
    - _self - 同一窗口，默认值；
    - _top - 整个窗口打开；
    - framename-框架或浮动框架名

5.2.2 超链接路径

超链接路径：绝对路径、相对路径、根路径

绝对路径有2种:

1. 从盘符开始定义的文件路径，如E:\web\index.html
2. 从协议开始定义的URL网址，例如中国教育与科研计算机网的网址 http://www.edu.cn。
   相对路径:相对于当前文件的路径，从当前文件所在位置指向目的文件的路径

根路径:
从网站的最底层开始起，一般网站的根目录就是域名下对应的文件夹。以一个斜杠“/”开头，代表根目录，然后书写文件夹名，最后书写文件名。
例如/download/index.html。

5.2.3 超链接分类

超链接可以分为内部链接和外部链接两种
内部链接是指网站内部文件之间的链接
外部链接是指网站内的文件链接到站点内容外的文件

5.3 超链接的应用

5.3.1 创建HTTP文件下载超链接

网站提供软件、文件等资料下载，下载文件的链接指向文件所在的相对路径或绝对路径，文件类型：*.doc/*.pdf/*.exe/*.rar等。

5.3.2 创建FTP站点访问超链接

FTP服务器链接和网页链接区别在于所用协议不同，浏览网页采用http协议，而FTP服务器采用FTP协议连接

```
<a href = "ftp://服务器IP地址或域名">链接的文字</a>
```

5.3.3 创建图像链接

链接标题是一个图像，浏览时单击链接图像时，可以打开超接href所设置的URL

```
<a href=“#”>
<img src=“url” width=“” height=“” align=“” border=“” alt=“”>
</a>
```

5.3.4 创建电子邮件超链接

一般网站上都会设置“联系我们”这样的栏目或超链接，目的是方便用户及时与网站管理员进行沟通与联系--电子邮件超链接

```
<a href=“mailto:E-mail地址[ ?subject=邮件主题[&参数=参数值]]”>
链接内容
</a>
```

5.3.5 创建页面书签链接

书签是指到文章内部的链接，可是实现段落间的任意跳转。实现这样的链接要先定义书签名称和书签链接
步骤:

1. 定义书签名: 书签标题
2. 定义书签链接
   书签标题 同一页面内

书签标题 不同页面间

5.4 浮动框架

浏览器窗口含有孤立的子窗口称为浮动框架。在浏览器窗口中使用<iframe></iframe>标记，可以嵌入浮动框架

```
<iframe name=" name " src="url"  width="“" height=" " ></iframe>
   <a href="target.html" target= " name" >链接标题</a>
```

属性	说明	属性	说明
src	设置源文件属性	scrolling	设置框架滚动条
name	设置框架名称	frameborder	设置框架边框
width	设置浮动框架窗口宽度	marginwidth	设置框架左右边距
height	设置浮动框架窗口高度	marginheight	设置框架上下边距
align	设置框架对齐方式		

第六章 图像与多媒体文件

6.1 图像

网页上插入图像的方法就是使用<img>标记。它的众多属性可以控制图像的路径、尺寸和替换文字等各种功能

```
<img src=“url” width=“” height=“” hspace=“” vspace=“” align=“” border=“” a
```

属性说明:

- src(source):图像的 URL路径。
- alt ： 添加图片的替代文字。
- width/height：设置图片的宽度和高度(px,%)。
- border：设置图片边框(px);
- align:设置图片对齐方式（水平/垂直两个方向）;
- hspase/vspase:设置图片的间距设置(单位不加px)
  设置图像对齐方式及间距

```
<img src=" URL " align="value">
    <img src="URL" hspace="水平间距数值" vspace="垂直间距数值">
```

取值	说明
top	图像的顶端和当前行的文字顶端对齐，当前行高度相应扩大
middle	图像水平中线和当前行的文字中线对齐，当前行高度相应扩大
bottom	图像的底端和当前行的文字底端对齐，当前行高度相应扩大
left	图像左对齐，浮动游离于文字之外，文字环绕图像周围，文字行高度没有任何变化
center	图像中线和当前行的文字中线对齐，当前行高度相应扩大
right	图像右对齐，浮动游离于文字之外，文字环绕图像周围，文字行高度没有任何变化

设置图像热区链接

```
 <img src="图片地址" usemap="#映射图片名称">
<map name="映射图片名称">
      <area  shape="热区形状" coords="热区坐标" href="URL">
</map>
```

- shape:rect(矩形)、 circle (圆形)、 poly (多边形)
- coords与shape对应的坐标值：
  - rect: x1,y1,x2,y2 (4 个值);
  - circle: center-x、center-y、radius(3个值)
    - poly:(x1,y1,x2,y2 ,…,xi,yi,…,xn,yn,x1,y1)((n+1) *2个值，多边形的顶点数)。

6.2 滚动文字

6.2.1 添加滚动文字

通过marquee标记可以添加滚动文字(内容)，增加动态效果，丰富网页的内容

```
<marquee width="" height="" bgcolor="" direction="up|down|left|right" beha
```

6.2.2 设置滚动文字背景颜色和滚动循环

```
<marquee bgcolor="" loop="5" >滚动内容</marquee>
```

文字背景颜色采用5种方法，最常用的设置方法是十六进制和rgb()函数。
默认情况下，滚动文字将会不停地循环滚动，但使用loop属性就可以设置滚动文字的滚动循环次数。循环次数直接使用数字表示。一般为整数，-1表示无限循环。

6.2.3 设置滚动方向与滚动方式

```
<marquee direction="滚动方向" behavior="滚动方式">滚动内容</marquee>
```

- direction:up|down|left(向左滚动，默认值)|right
- behavior: scroll(循环往复滚动，为默认值)| slide(滚动一次就停止)| alternate(来回交替滚动)

6.2.4 设置滚动速度和滚动时延

```
<marquee scrollamount="20px" scrolldelay="100ms"> 滚动 内容</marquee>
```

语法说明:

- 滚动速度是滚动文字每次移动的长度，长度用数字表示，单位为像素px。
- 延迟时间以毫秒ms为单位，延迟时间越小滚动速度越快，延迟时间越大即会出现走走停停的效果。

6.2.5 设置滚动范围和滚动空白空间

```
    <marquee width="" height="" hspace="" vspace="" >滚动内容</marquee>
```

- 宽度值和高度值均用数字表示，单位为像素
- hspace、vspace属性值是整数，单位为像素,但不要加单位

6.3 音频、视频及Flash文件

使用<embed></embed>标记，可以播放的文件类型有Midi、Mav、AIFF、SWF、AV、MP3、MOV、AVI 等。

```
<embed src="多媒体文件" width="界面的宽度" height="界面的高度"autostart="true
```

- width、height：整型值，单位为像素。
- 设置宽度和高度会出现播放界面，否则不显示播放界面。一些高版本浏览器不设置宽度和高度也可以出现播放界面。如果播放声音、音乐文件作为背景音乐时，必须同时将宽度和高度属性的值设置为0。
- src：设置媒体文件的路径。
- autostart：逻辑值。true 为自动播放；false 为不自动播放。
- loop：逻辑值。规定音频或视频文件是否循环。属性值为true 时，音频或视频文件循环；属性值为false 时，音频或视频文件不循环

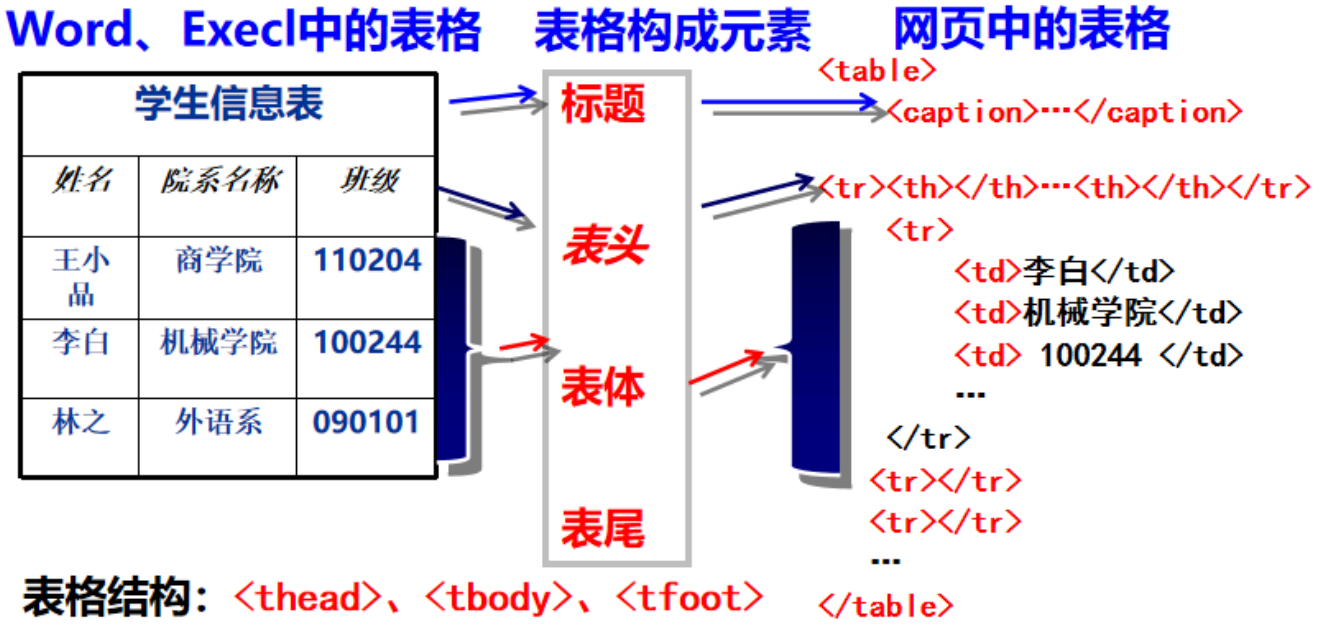
第十一章 表格

11.1 表格

表格是网页设计制作不可缺少的元素，它以简洁明了和高效快捷的方式将图片、文本、数据和表单的元素有序的显示在页面上，让我们可以设计出漂亮的页面

标记	说明	标记	说明
<table></table>	表格标记	<thead></thead>	定义表格的表头
<caption></caption>	表格标题标记	<tbody></tbody>	定义表格的主体
<th></th>	表格表头标记	<tfoot></tfoot>	定义表格的页脚
<tr></tr>	表格的行标记		
<td></td>	表格的列标记		

11.2 表格标记



在HTML中，使用<table></table>标记可以创建表格。

```
 <table>
      <caption>插入表格标题</caption>
     <tr>
         <th></th> ><th></th> ><th></th> …
    </tr>
    <tr>
         <td ></td> <td ></td> <td ></td> …
    </tr>
</table>
```

语法说明:

- <table></table>标记表示插入表格
- <tr></tr>表示插入一行
- <td></td>表示插入一列
- <th></th>插入表头

11.3 表格属性设置

表格是网页文件中布局的重要元素，制作网页的过程中常常需要对网页中的表格做一些设置，对表格的设置实质是对表格标记属性的一些设置

```
<table border"" bordercolor="" bordercolorlight""  align="center" borderco
```

1. 表格的边框属性--border
2. 边框的颜色--bordercolor
3. 亮边框的颜色--bordercolorlight
4. 暗边框的颜色– bordercolordark
5. 背景颜色—bgcolor
6. 背景图像—background
7. 表格宽度与高度—width/height
8. 表格对齐属性-align

表格边框样式属性

```
<table  frame="" rules=""> </table>
```

frame: 设置表格边框样式； rules: 设置表格内部边框样式

frame属性值	说明	rules属性值	说明
above	显示上边框	all	显示所有内部边框
below	显示下边框	none	不显示内部边框
hsides	显示上下边框	rows	仅显示行边框
vsides	显示左右边框	cols	仅显示列边框
lhs	显示左边框	groups	显示介于行列间边框
rhs	显示右边框		
border	显示上下、左右边框		
void	不显示边框		

表格单元格间距、单元格边距属性

```
<table cellspacing=“5px” cellpadding=“5px”>  </table >
```

cellspacing:设置表格的单元格和单元格之间的间距，默认值是2px，使得表格布局不会显得过于紧凑
cellpadding:设置单元的内容与边框的距离

表格水平对齐
align="center"

设置表格行属性
表格行tr标记的属性用于设置表格某一行的样式，其属性设置如下表所示。

属性值	说明	属性	说明
align	行内容水平对齐	bordercolor	行的边框颜色
valign	行内容垂直对齐	bordercolorlight	行的亮边框颜色
bgcolor	行的背景颜色	bordercolordark	行的暗边框颜色

```
<tr align="left | center | right"> </tr>
<tr valign="top | middle | bottom"></tr>
```

设置单元格的属性

表格列标记td的属性可以设置表格单元格的显示风格。常用的属性如下表所示。单元格的颜色、边框和对齐属性与行tr标记一样

属性值	说明	属性	说明
align	单元格内容水平对齐	bordercolorlight	单元格的亮边框颜色
valign	单元格内容垂直对齐	bordercolordark	单元格的暗边框颜色
bgcolor	单元格的背景颜色	rowspan	单元格跨行
background	单元格背景图像	colspan	单元格跨列
bordercolor	单元格的边框颜色	width	单元格宽度
		height	单元格高度

单元格跨行、列

<td>的属性用于设定表格中某一单元格的属性。

单元格跨行rowspan(跨行合并-纵向合并)

单元格跨列colspan(跨列合并-横向合并)

基本语法：

< td rowspan=“3”>…</td>

<td colspan=“3”>…</td>

第十二章 表单

12.1 表单概述

表单是较为复杂的HTML元素，经常与脚本、动态网页、后台数据处理等结合在一起使用，是设计动态网页的必备元素。

利用表单可以在HTML页面中插入一些表单控件(元素)，如文本框、提交按钮、重置按钮、单选按钮、复选框、下拉列表框等，完成各类信息的采集。

```
    <form method="post" action="">
        <input type="text" name="">
```

```
          <textarea name="" rows="" cols=""></textarea>
         <select name="">
             <option value="" selected> </option >
        </select>
    </form>
```

表单标记

- name:给定表单名称，表单命名之后就可以用脚本语言(如VBScript,JavaScript)对它进行控制
- action:指定处理表单信息的服务器端应用程序
- method:用于指定表单处理表单数据方法，method的值（get、post，默认get)。
- enctype:规定表单数据在发送到服务器之前进行编码。有三种取值，分别如如下：
  - application/x-www-form-urlencoded (在发送前编码所有字符,默认)
  - multipart/form-data(不对字符编码)
  - text/plain(空格转换为 “+” 加号，但不对特殊字符编码) 。

12.2 定义域和域标题

利用<fieldset> </fieldset>域标记可将表单内的相关元素进行分组
当一组表单元素放到fieldset标记内时，浏览器会以特殊方式来显示它们，它们可能有特殊的边界、3D效果，或者可创建一个子表单来处理这些元素。<legend> </legend>标记定义域标题。

```
<form>
    <fieldset>
       <legend align=“left | center | right”>域标题</legend>
            <input name="" type="radio" value="" checked>
        ……
    </fieldset>
</form>
```

12.3 表单信息输入

表单的主要功能是为用户提供输入信息的接口，将输入信息发送请求到服务器并等待服务器响应

```
<form><input name="" type=" "    ></form>
```

属性	值	说明
name	name	定义input元素的名称。
type	text |password | checkbox |radio |image |submit |reset |button |file |hidden	规定input元素的类型。 text为单行文本输入框，password为密码输入框，checkbox为复选框，radio为单选按钮，image为图像按钮，submit为提交按钮，reset为重置按钮，button为普通按钮，file为文件选择框，hidden为隐藏框。

单行文本输入框、密码框文本框

```
<input name="" type="text" maxlength="" size="" value="“  readonly >
<input  name=" “   type="password"  maxlength=""  size="">
```

- maxlength：设置单行输入框输入的最大字符数；
- size：设置单行输入框可显示的最大字符数；
- value：文本框的值，指定输入框中初始值；
- readonly:只读,文本框不可编辑。

复选框、单选按钮

```
<input name="" type="checkbox" value="" checked>
<input name="" type="radio"  value="" checked>
```

checked表示预选中。每一个复选框name、value属性都是不同的。
每组单选按钮的name值必须相同，而value属性值必须不同

图像按钮

```
<input name="" type="image" src="" width="" height=" "  >
```

提交按钮、重置按钮和普通按钮
提交按钮

```
<input name="" type="submit"  value="">
```

value: 指定显示在提交按钮上的文字，默认值“提交查询内容”，需要给value赋个初值。点击提交按钮后，将表单数据提交给服务器

重置按钮

```
<input name=" "    type="reset"    value="">
```

value值默认为“重置”，不需要定义，但可以改变。点击该按钮可将表单域的内容清空。

普通按钮

```
<input name="" type="button" value="" onclick="">
```

普通按钮需要定义onclick属性，才能进行表单处理

文本选择框和及隐藏框
文本选择框

```
<input  name=""  type="file">
```

选择文件后并不能真正打开，只是将文件名回填到文件输入框内

隐藏框

```
<input name="" type="hidden" value="">
```

隐藏框不显示在表单中，随用户表单一起提交给服务器

12.4 多行文本输入框

```
<textarea name="" rows="" cols="" wrap=""> </textarea>
```

rows:指定输入的行数；
cols:指定输入的列数；
wrap:指定文本自动换行，值分别为：off|physical|virtual

12.5 下拉列表框

```
<select name="" size="" multiple>
      <option value=""     selected  >选项内容</option>
     <option      value=""  >选项内容</option>
    <option value=""   >选项内容</option>
 </select>
```

注：
size定义下拉列表的大小；
multiple设置列表框支持多选；
selected设置选项为预选状态

第十三章 HTML5基础与CSS3应用

13.1 HTML5概述

13.1.1 HTML5的八个特性

1. 语义特性（Semantic）。HTML5赋予网页更好的意义和结构。
2. 离线与存储特性（Offline & Storage）。HTML5开发的网页APP，启动时间更短，联网速度更快。由于有HTML5 APP Cache、本地存储功能、Indexed DB和File API说明文档。
3. 设备访问特性 (Device Access)。HTML5提供了前所未有的数据与应用接入开放接口。使外部应用可以直接与浏览器内部的数据直接相连，例如视频影音可直接与麦克风及摄像头相联。
4. 多媒体特性(Multimedia)。 支持网页端的Audio、Video等多媒体功能，与网站自带的APPS、摄像头、影音功能相得益彰。\
5. 三维、图形与特效特性（3D、Graphics & Effects）。基于SVG、Canvas、WebGL及CSS3的3D功能，用户会惊叹于在浏览器中，所呈现的惊人视觉效果。
6. 性能与集成特性（Performance & Integration）。 HTML5会通过Web Workers和XMLHttpRequest2等技术，帮助您的Web应用和网站在多样化的环境中更快速的工作。
7. 连接特性（Connectivity）。HTML5拥有更有效的服务器推送技术(Server-Sent Event和WebSockets)，能够帮助我们实现服务器将数据“推送”到客户端的功能。
8. CSS3特性(CSS3)。CSS3中提供了更多的风格和更强的效果。

13.1.2 HTML5的优势

1. 摆脱对平台的依赖。打开浏览器，直接就可以访自己的应用。

2. 实时更新。
3. 离线使用。用户可以离线使用，更新下载量较少。
4. 代码更安全。HTML5可以将Web代码全部加密，本地应用解密后再运行，大大的提供了代码的安全性。
5. 跨平台。JavaScript的代码可以在许多地方使用，包括移动应用、移动网站、PC网站、各种浏览器插件，甚至可以用WebKit封装作为跨平台的应用程序。
6. 可以充分利用Native。HTML5可以通过浏览器作为中介充分利用Native的好处(使用GPS、照相机、本地相册、读取本地联系人等)。某些Web无法实现的功能，可以利用Native来实现。

13.1.3 HTML5新增结构元素及页面元素

1. HTML5中新增加结构元素。例如Header页眉、nav导航、section节、article文章、aside侧栏、footer页脚。
2. HTML5 中新增页面元素。例如video、audio、embed、progress、time、mark、ruby、rt 、rp 、canvas 、command、datalist、output、wbr 、source、menu、details等

13.1.4 HTML5废除的元素与属性

(1) 纯表现的元素。如font、basefont、center、big、s、u、strike、tt。
(2) 对可用性产生负面影响的元素。如frameset、frame、noframes等元素。HTML5只支持浮动框架（内联框架）iframe元素。
(3) 易产生混淆的元素。如acronym、applet、isindex、dir等元素。
(4) 废除只有部分浏览器支持的元素。如blink、bgsound、marquee等元素。(5) 其它被废除的元素。如废除rb，使用ruby替代；废除listing使用pre替代；废除xmp使用code替代；废除nextid使用guids替代；废除plaintex使用“text/plian”MIME类型替代。

13.1.5 浏览器支持与选择

一些低版本的浏览器并不支持HTML5，如IE6～IE8浏览器。所有新、旧浏览器，对无法识别的元素均会视作内联(inline)元素来自动处理。可以通过其它方法让这些浏览器能够处理“未知”的HTML元素。使用http://html5test.com来测试浏览器的支持
html5shiv是针对IE浏览器比较好的解决方案。html5shiv主要解决HTML5提出的不被IE6～IE8识别新的元素，不能作为父节点包裹子元素，不能应用CSS样式。从指定网站上直接下载并保存到本地项目目录中（https://github.com/aFarkas/html5shiv/）

13.2 HTML5文档结构

HTML5 文档结构同样是由头部和主体两部分组成，只是新增了一些结构元素，如header、nav、article、section、aside、footer 六个结构元素，这些元素都是块级元素

13.2.1 HTML5页面结构

图13-2 HTML4.01页面布局

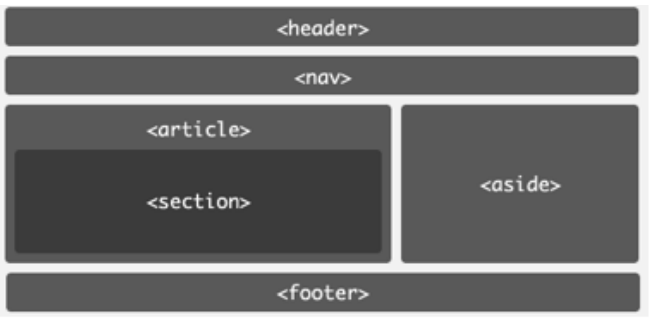


图13-3 HTML5结构元素布局

13.2.2 HTML5 新增结构元素

header 标记

header 标记定义文档和区域的页眉，通常是一些引导和导航信息

它不局限于写在网页头部，也可以写在网页内容里面。通常<header>标记至少包含（但不局限于）一个标题标记（h1～h6），也可以包括hgroup（标题组合）标记、表格标识、搜索表单、导航

nav 标记

nav 标记代表页面的一个部分，是一个可以作为页面导航的链接组。建议不要在footer元素中使用nav 元素，否则易造成页面显示不正确。配置相应的CSS 代码可以实现水平导航

article标记

article标记是一个特殊的section标记，它比section具有更明确的语义，它代表一个独立的、完整的相关内容块，可独立于页面其它内容使用

例如论坛帖子、博客文章、新闻故事、评论

一般来说，article会有标题部分，通常包含在header内，有时也会包含footer。article标记可以嵌套，内层的article对外层的article标记有隶属关系

section标记

section 标记定义文档中的节。例如章节、页眉、页脚或文档中的其他部分

一般用于成节的内容，会在文档流中开始一个新的节

它用来表现普通的文档内容或应用区块，通常由内容及其标题组成

section 元素不是一个普通的容器元素，它表示一段专题性的内容，可以带有标题。如果描述一件具体的事物，建议使用article 来代替section；如果使用section，仍可以使用h1 作为标题，而不用担心它所处的位置。如果一个容器需要定义样式或定义行为，建议用div 元素

aside标记

aside（侧栏，也称为旁注）标记用来说明其所包含的内容与页面主要内容相关，但不是该

页面的一部分，类似于使用括号对正文进行注释（就像这样）。括号中的内容提供关于该元素的一些附加信息，例如广告、成组的链接、侧栏等

footer标记

footer 标记定义section 或文档的页脚，包含了与页面、文章或部分内容有关的信息，例如文章的作者或者日期

作为页面的页脚时，一般包含了版权、相关文件和链接。它与页眉header 标记用法相同，在一个页面中可以多次使用，若在一个区段的最后使用footer标记，那么它就相当于该区段的页脚。

13.3 HTML5新增页面元素

13.3.1 hgourp标记

标题组合hgroup 标记是对网页或区段section 的标题元素（h1～h6）进行组合

```
    <hgroup>
          <h1>标题1</h1>
          <h2>标题2</h2>……
      </hgroup>
```

13.3.2 figure标记与figcaption标记

figure标记用于对元素进行组合，常用于图像与图像描述组合。

figcaption（图题）标记用于定义figure元素的标题（caption），可以给一组图像标记定义标题，但figcaption 标记不是必需的。如果包含了figcaption元素，那么它必须放置在figure元素的第一个或最后一个子元素的位置上。

```
<figure>
     <p>图像文件说明内容。</p>
     <img src=""   width=""  alt="" title="" />
     <figcaption >为图添加标题)</figcaption>
</figure>
```

13.3.3 mark标记与time标记

记号mark 标记用来定义带有记号的文本。在需要突出显示文本时可以使用mark 标记。此标记对关键字做高亮处理（黄底色标注），突出显示，标注重点，在搜索方面可以应用。

时间time 标记用来定义公历的时间（24 小时制）或日期，时间和时区偏移是可选的。该标

记能够以机器可读的方式对日期和时间进行编码。该标记不会在任何浏览器中呈现任何特殊效果

```
<mark>重点标注的内容</mark>
<time>9:00</time> <!-- 定义时间 -->
<time datetime="2017-05-01" pubdate="pubdate">国际劳动节</time>
```

time标记的pubdate属性：指示该标记中的日期/时间是文档（或最近的article标记）的发布日期。
time标记的datetime属性：规定日期/时间

13.3.4 details标记与summary标记

细节details标记是一个开关式、交互式控件，用来定义用户可见的或者隐藏的需求补充细节，任何形式的内容都能被放在该标记中。该元素的内容对用户是不可见的，除非设置了open属性。
摘要summary 标记配合使用可以为details定义标题，summary元素应该是details元素的第一个子元素。标题是可见的，用户单击标题时，会显示出details。只有Chrome、Safari 6以上支持summary标记。

```
<details open>
        <summary> details的标题</summary>
        details的详细内容
</details>
```

13.3.5 progress标记与meter标记

进度Progress标记用来定义运行中的任务进度（进程）。该标记有两个属性：max表示规定需要完成的值；value规定进程的当前值
度量meter 标记定义已知范围或分数值内的标量测量，也被称为gauge(尺度)。如磁盘用

量、CPU使用率等等

表13-4 meter标记的属性、值及描述表

属性名	值	描述
form	form_id	规定meter元素所属的表单。
high	number	规定被界定为高值的范围。
low	number	规定被界定为低值的范围。
max	number	规定范围的最大值。
min	number	规定范围的最小值。
optimum	number	规定度量的最优值。
value	number	必需。规定度量的当前值。

13.3.6 input标记与detalist标记

input 标记用于搜集用户信息，此处仅介绍通过input 标记的list 属性与datalist 标记的id 属性进行关联，即将此两个属性的值设置为相同的值，通过datalist 标记列出所有合法的输入值列表。

选项列表datalist 标记用来定义input 标记可能的选项列表。一般与input 标记配合使用，主要用来定义input 可能的值，提供“自动完成”的功能，方便用户输入。datalist 标记及其选项不会被显示出来，只有当用户鼠标盘旋在input 标记域时，才能看到“▼”，然后单击“▼”弹出一个下拉列表，提供用户选择作为用户的输入数据。

```
<input list="courese" placeholder="请选择课程" />
<datalist id="courese">
<option value="HTML5移动应用开发">
<option value=".NET应用开发">
<option value="JavaEE应用开发">
<option value="PHP+MySQL应用开发">
</datalist>
```

13.4 HTML5表单

HTML5 对表单系统做了彻底的改造，以适应当前的应用

13.4.1 HTML5 新增的表单属性

HTML5表单新增一些新属性。这些属性是autocomplete、novalidate。

form标记的新属性

autocomplete属性

autocomplete：on|off。属性规定form标记或类型为text、search、url、telephone、email、password、date pickers、range、color的input标记是否具有自动完成的功能。当

表单元素设置了自动完成功能后，会记录用户输入过的内容，双击表单元素会显示历史输入。

novalidate属性

novalidate：true|false。属性规定在提交表单时不进行验证form或类型为text、search、url、telephone、email、password、date pickers、range、color的input标记。

input标记的新属性

`height 和width 属性`

height 和width 属性规定只适用于image 类型的input 标记的图像高度和宽度

`form 属性`

form 属性规定输入域所属的一个或多个表单。form 属性必须引用所属表单的id

`list 属性`

list属性规定输入域的datalist。datalist标记是输入域的选项列表。list属性适用于类型为text、search、url、telephone、email、password、date pickers、range、color 的input标记。

`placeholder 属性。`

placeholder属性提供一种提示，描述输入域所期待的值,属性支持类型为text、search、url、telephone、email、password的input标记。

`autofocus属性`

autofocus属性规定在页面加载时，该域自动地获得焦点

`required 属性`

required 属性规定必须在提交之前填写输入域（不能为空）

`min、max 和step 属性`

min、max 和step 属性用于为包含数字或日期的input 类型规定限定

`multiple 属性`

规定输入域中可选择多个值。适用于类型为email、file 的input 标记

`form overrides 表单重写属性（form override attributes）`

允许重写form 元素的某些属性设定

`pattern属性`

一般为正则表达式 规定用于验证input域的模式。适用于text、search、 url、tel、email、password等类型的input标记。

13.4.2 新增的表单元素

标记名称	标记功能描述
<output></output>	定义不同类型的输出，比如脚本的输出。
<keygen></keygen>	规定用于表单的密钥对生成器字段。
<datalist></datalist>	定义选项列表。与input元素配合使用该元素，来定义input可能的值。

output 标记

output 标记定义不同类型的输出。该标记有for、form、name 三个属性。for 属性用于描述计算中使用的元素与计算结果之间的关系，其值为每一元素的id，多个id 之间用空格分隔。form 属性用于定义输入字段所属的一个或多个表单。name 属性用于定义对象的唯一名称（表单提交时使用）。

keygen标记

keygen 标记用来提供一种验证用户的可靠方法.keygen 元素是密钥对生成器（key-pairgenerator）。当提交表单时，会生成两个键：一个是私钥（private key），一个公钥（public key）。

私钥存储于客户端，公钥则被发送到服务器。公钥可用于之后验证用户的客户端证书（clientcertificate）。目前，浏览器对此元素的糟糕的支持度不足以使其成为一种有用的安全标准。

datalist标记

datalist 标记规定了input 标记可能的选项列表。datalist 标记被用来为input 标记提供“自动完成”的特性。使用input 标记的list 属性来绑定datalist 元素（list 属性=datalist 的id 属性值）。

13.4.3 HTML5新增的input类型

HTML5 增加很多新的表单输入类型，分别为color、date pickers（日期选择器，包括date、month、week、datetime、time、datetime-local）、email、number、range、search、tel、url。目前所有的主流浏览器一般都支持新的input 类型，即使不被支持，仍可以显示为常规的文本域。

Input 类型——Date Pickers（日期选择器）

HTML5 提供多个可供选取日期和时间的新输入类型：

（1）date——选取日、月、年。

（2）month——选取月、年。

（3）week——选取周和年。

（4）time——选取时间（小时和分钟）。

（5）datetime——选取时间、日、月、年（UTC 时间）。

（6）datetime-local——选取时间、日、月、年（本地时间）。

input 类型：color。

```
<input type="color" name="favcolor">
```

input 类型：tel 定义输入电话号码字段

```
<input type="tel" name="usrtel">
```

input 类型：email

email 类型用于包含e-mail 地址的输入域。在提交表单时，会自动验证email 域的值是否合法有效。

```
<input type="email" name="useremail">
```

input 类型：number

```
<input type="number" name="mynumber" min="0" max="100">
```

属性	描述
disabled	规定输入字段是禁用的。
max	规定允许的最大值。
maxlength	规定输入字段的最大字符长度。
min	规定允许的最小值。
pattern	规定用于验证输入字段的模式。
readonly	规定输入字段的值无法修改(只读)。
required	规定输入字段的值是必需的。
size	规定输入字段中的可见字符数。
step	规定输入字段的合法数字间隔。
value	规定输入字段的默认值。

input 类型：range

range 类型用于包含一定范围内数字值的输入域。range 类型显示为滑动条。

```
<input type="range" name="money" min="1" max="1000" step="5">
```

input 类型：search

search 类型用于搜索域，例如站点搜索或Google 搜索。

```
<input type="search" name="websidesearch">
```

input 类型：url

url 类型用于包含URL 地址的输入域。在提交表单时，会自动验证url 域的值

```
    <input type="url" name="homepage">
```

13.5 HTML5视频与音频

HTML5 提供了video 标记和audio 标记

13.5.1 video标记及属性

HTML5 规定了一种通过video 元素来包含视频的标准方法。Video 标记支持三种视频格式，分别为MP4、WebM、Ogg。

- Ogg：带有Theora 视频编码和Vorbis 音频编码的Ogg 文件。
- MPEG4：带有H.264 视频编码和AAC 音频编码的MPEG 4 文件。
- WebM：带有VP8 视频编码和Vorbis 音频编码的WebM 文件。

```
<video src="movie.ogg" width="320" height="240" controls="controls">
    您的浏览器不支持 video标记。
</video>
```

width 和height 属性：控制视频的尺寸。使用时需要设置视频的高度和宽度，便于视频播放。如果不设置宽度和高度，页面就会根据原始视频的大小而改变。
src 属性：规定要播放的视频的url。
loop：设置该属性，则当媒体文件完成播放后再次开始播放。
preload：设置该属性，则视频在页面加载时进行加载，并预备播放。
如果使用autoplay，则忽略该属性。
该属性有三种值：auto（一旦页面加载，则开始加载音频/视频）
metadata（当页面加载后仅加载音频/视频的元数据）
none（页面加载后不应加载音频/视频）。格式如下：

```
<video preload="auto|metadata|none">
```

poster 属性:
用于在视频下载时显示的图像（海报图片），或者在用户点击播放按钮前显示的图像。如果未设置该属性，则使用视频的第一帧来代替。赋值方法：poster=“url”
video 标记支持多个source 标记。可以使用source 标记为video 标记和audio 标记提供多个不同的音频、视频文件，以解决浏览器支持

13.5.2 audio标记及属性

HTML5 使用audio 标记能够播放声音文件或者音频流。同样可以使用source 标记给audio 标记提供不同格式的音频文件，浏览器将使用第一个支持的音频文件

```
<audio width="320" height="240" controls="controls">
<source src="horse.ogg" type="audio/ogg">
<source src="horse.mp3" type="audio/mpeg">
您的浏览器不支持 video标记。
</ audio >
```

第十七章 HTML5高级应用

自己学

第十四章 JavaScript基础

14.1.1 JavaScript简介

JavaScript是一种基于对象和事件驱动、安全性、轻量级、解释型、弱类型的客户端脚本语言。 决定WEB页面的行为，具有客户端数据验证、用户交互等功能。
JavaScript具有如下特点

1. 简单性(小程序、无须编译、解释性、弱数据类型)
2. 安全性(Browser无法访问本地硬盘数据/写入到数据库)
3. 动态性（JS可以直接对用户提交的信息作出回应）
4. 跨平台性（支持JS的Browser）

14.1.2 第一个JS程序

```
<script type=“text/javascript [src=“外部JS文件”]>js语句块;</script>
<script language=“javascript [src=“外部JS文件”]>js语句块;</script>
```

14.1.3 JS放置的位置

JavaScript代码放置的位置：

1. 头部
2. 主体
3. 单独的js文件

4. 直接在事件处理代码中
   JavaScript程序本身不能独立存在，它是依附于HTML代码，经浏览器解释执行
   可将JavaScript函数写成一个独立的js文件，在HTML文档中引用该js文件，引用时必须使用src属性。JavaScript文件的扩展名为*.js。格式如下：

```
<script type=“text/javascript” src=“外部JS文件”></script>
```

此时在标记之间的所有JS语句都被忽略，不会执行

头部
JS脚本插入在头部时，通常需要定义为函数格式,格式:

```
function 函数名(参数1,参数2,…, 参数n){函数体语句;}
```

主体
JS脚本插入在主体时，JavaScript语句能够被立即执行。也可以定义成函数，但必须引用才能执行

外部JS
外部JS文件需要引用到HTML文件中才能被执行。编写外部JS文件时不需要使用、<script></script>标记

事件处理代码

```
<!-- edu_14_1_4.html -->
<html>
  <head>
  <title>直接在事件处理代码中加入JavaScript代码</title>
   </head>
  <body>
   <form>
    <input type="button" onclick="alert('直接在事件处理代码中加入JavaScript代
    </form>
   </body>
</html>
```

JS代码直接放置在事件处理的代码中，可以直接运行。也可以加上“javascript:alert(‘信息’);”

14.2 JS程序

JavaScript程序由语句、语句块、函数、对象、方法、属性等构成，通过顺序、分支和循环三种基本程序控制结构来进行编程

14.2.1 语句和语句块

JavaScript语句是发送给浏览器的命令，这些命令的作用是告诉浏览器要做的事情

```
alert(“这是告警消息框!”); //弹出告警消息框
```

JavaScript语句可以分批组合起来形成语句块，语句块以左花括号

```
{var s=0;document.write(“S的值=”+s);}   //赋值，并输出到页面
```

14.2.2 代码

JavaScript代码是由若干条语句或语句块构成的执行体

```
<script type="text/javascript">
    var color="red";
    if(color=="red")
    {
      document.write("颜色是红色!");
      alert("颜色是红色!");
    }
</script>
```

14.2.3 消息对话框

1. 警告框
   alert (message) ;
2. 确认框
   var yn=confirm (message);
3. 提示框
   var s1=prompt (text, defaultText);

14.2.4 JS注释

单行注释：使用“//”作为注释标记，可以单独一行或跟在代码末尾，放在同一行中，“//”后为注释内容部分。
多行注释：以“/*”标记开始，以“*/”标记结束，两个标记之间所有的内容都是注释文本

14.3 标识符和变量

14.3.1 命名规范

JavaScript关于标识符的规定如下：
(1)必须使用字母或者下划线和$开始。
(2)必须使用英文字母、数字、下划线组成，不能出现空格或制表符。
(3)不能使用JavaScript关键字与JavaScript保留字。
(4)不能使用JavaScript语言内部的单词，比如Infinity，NaN，undefined等。
(5)大小写敏感，如name和Name是不同的两个标识符。

JS关键字

表14-1 JavaScript的关键字

break	case	catch	continue	default
delete	do	else	finally	for
function	if	in	instanceof	new
return	switch	this	throw	try
typeof	var	void	while	with

JS保留字

表14-2 JavaScript的保留字

abstract	boolean	byte	char	class
const	debugger	double	enum	export
extends	final	float	goto	implements
import	int	interface	long	native
package	private	protected	public	short
static	super	synchronized	throws	transient
volatile				

14.3.2 数据类型

JavaScript中的数据类型可分为字符型、数值型、布尔型、Null、Undefined和对象六种类型

字符型
字符型数据又称为字符串，由若干个字符组成，并且用单引号或双引号封装起来
在使用字符串的过程中，需要注意单引号、双引号必须成对使用相互包含，但不能交叉

数值型
是JavaScript中最基本的数据类型之一，分为整型、浮点型、内部常量以及特殊值
整数表示方法有十进制表示、八进制(0开头)和十六进制(0x开头)的方式表示
浮点数:5.434 3.5E15
内部常量:Math.E、Math.PI
特殊值:Infinity NaN-Not a Number

Boolean（布尔型）
JavaScript中，通常采用true和false表示布尔型数据，但也可将他们转换为其他类型的数据，例如可将值为true的布尔型数据转换为整数1，而将值为false的布尔型数据转换为整数0。
Null
null,表示空,不是0，0是有值的。
Undefined
变量创建后未赋值（数字：NaN；字符串：Undefined；Boolean:false)

Object
对象也是JS的重要组成部分，如date、window、document等，在后面介绍

数据类型之间可以通过函数进行转换
转换函数如下：
Nnumber(value):把值转换成数字（整型或浮点数）
String(value):把值转换成字符串
Boolean(value):把值转换成Boolean类型

14.3.3 变量

变量:可以保存执行时变化的值的名字，称为“变量”，变量是存储信息的容器
使用 var 声明变量

14.3.4 转义字符

转义字符	代表含义	转义字符	代表含义
\b	退格符	\t	水平制表符
\f	换页符	\'	单引号
\n	换行符	\"	双引号
\r	回车符	\\	反斜线

14.4 运算符和表达式

JavaScript运算符主要有：算术运算符、关系运算符、逻辑运算符、赋值运算符、自增自减运算符、逗号运算符和位运算符等

根据操作数的个数，将运算符分为一元运算符、二元运算符和三元运算符。

由操作数（变量、常量、函数调用等）和运算符结合在一起构成的式子称为“表达式”，最简单的表达式可以是常量名称

对应的表达式包括：算术表达式、关系表达式、逻辑表达式、赋值表达式、自增、自减表达式、逗号表达式、条件表达式、位表达式。

14.4.1 算术运算符和表达式

算术运算符负责算术运算，用算术运算符和运算对象(操作数)连接起来符合规则的式子，称为算术表达式

- 双元运算符
- 单元运算符

表14-6 算术运算符

运算符	描述	例子（假定a = 2)	结果
+	加	b = a+2	b = 4
-	减	b = a -1	b = 1
*	乘	b = a * 2	b = 4
/	除	b = a / 2	b = 1
%	取模	b = a%2	b = 0
++	自增	b = a++ （后置，先使用再运算)	b = 2
--	自减	b = - -a （前置，先运算再使用)	b = 1

14.4.2 关系运算符和表达式

运算符	>	<	>=	<=	!=	==	===	!==
名称	大于	小于	大于 或等于	小于 或等于	不等于	等于	全等于	非全等于
表达式	6>5	6<5	6>=5	6<=5	6!=5	6==5	5==="5"	5!=="5"
结果	true	false	true	false	true	false	false	true
判断内容	数值	数值	数值	数值	数值	数值	数值与类型	数值与类型

14.4.3 逻辑运算符和表达式

表14-8 逻辑运算符				
a	b	!a（非）	a||b或	a&&b与
true	true	false	true	*true*
true	false	false	true	false
false	true	true	true	false
false	false	true	*false*	false

14.4.4 赋值运算符和表达式

基本语法：

- ■简单赋值运算：<变量> = <变量> operator <表达式>
- ■复合赋值运算：<变量> operator = <表达式>

运算符	=	+=	-=	*=	/=	%=
名称	赋值	加法赋值	减法赋值	乘法赋值	除法赋值	模赋值(求余赋值)
表达式	i=6	i+=5	i-=5	i*=5	i/=5	i%=5
示例	var i=6;	i+=5;	i-=5;	i*=5;	i/=5;	i%=5;
i的结果	6	11	1	30	1.2	1
等价于		i=i+5;	i=i-5;	i=i*5;	i=i/5;	i=i%5;
位移	a=4	<<=	>>=	>>>=		
表达式		a<<=1	a>>=1	a>>>=1		
a结果		a=8	a=2	a=2		

14.4.5 位运算符和表达式

位运算符是对二进制表示的整数进行按位操作的运算符

位运算符：&-按位与；~ -按位非；|-按位或；^-按位异或

14.4.6 条件运算符和表达式

条件运算符是一个3元运算符，也就是该运算涉及3个操作数
变量=布尔表达式？真值表达式 : 假值表达式

其他表达式
1.逗号运算符（,）
var x=1 , y=2 , z=3;
x=y+z , y=x+z;
2.新建对象运算符（new）
var str1=new String();var stu=new Array();
3.删除运算符（delete）：是一个一元运算符，用于删除一个对象的属性或某个数组的元素。
delete array[30],delete object.height
4.类型运算符（typeof）
typeof(300),typeof("Welcome to You!")

14.5 JS程序控制结构

14.5.1 顺序结构

代码顺序执行

14.5.2 分支结构

在 JavaScript 中，可以使用下面几种条件语句：
if 语句（单条件单分支）：在一个指定的条件成立时执行代码。
if...else 语句（单条件双分支）：在指定的条件成立时执行代码，当条件不成立时执行另外的代码。
if...else if....else 语句（多条件多分支）：使用这个语句可以选择执行若干块代码中的一个。
switch 语句（单条件多分支）：使用这个语句可以选择执行若干块代码中的一个。

14.5.3 循环结构

for循环、while循环、do-while循环

for-in循环

```
for (变量 in 对象){  执行代码；  }
```

14.6 JS函数

JavaScript函数分为系统内部函数和系统对象定义的函数及用户自定义函数

常用系统函数分全局函数和对象定义的函数
全局函数它不属于任何一个内置对象，使用不需要加任何对象名称，直接使用

14.6.1 常用的系统函数

全局函数

1. 计算表达式的结果函数：eval（字符串表达式）
   返回值：表达式的值或“undefined”
2. 编码函数escape()：escape(字符串)
   escape() 函数将参数字符串中的特定字符(ISO-Latin-1 字符集)进行编码，并返回一个编码后的字符串。它可以对空格、标点符号及其他非ASCII字母表的字符进行编码，除了以下字符：“* @ - _ + . / ”
3. 解码函数:unescape(string)
   unescape 函数返回的字符串是 ISO-Latin-1 字符集的字符。参数string包含形如“%xx”的字符的字符串，此处xx为两位十六进制数值。
4. 字符型转换成数值型函数:parseFloat(string)
5. 字符型转换成数值型函数:parseInt(numbestring , radix)
   以“0x”开始-16进制；以“0“开始--8进制；其他--10进制
6. 判断是否是NaN()函数:isNaN(testValue)

对象的函数

1. toString(radix)。将Number型数据转换为字符型数据，并返回指定的基数的结果
   其中radix范围2～36，若省略该参数，则使用基数10
2. toFixed(n)。将浮点数转换为固定小数点位数的数字
   n是整数，设置小数的位数，如果省略了该参数，将用0代替
3. 字符串查找和提取常用函数

方法	说明
indexOf(searchvalue,fromindex)	从前向后搜索字符串。返回某个指定的字符串值在字符串中首次出现的位置,如果没有发现，返回-1
lastIndexOf(searchvalue,fromindex)	从后向前搜索字符串。返回一个指定的字符串值最后出现的位置，如果没有发现，返回-1
charAt(index)	返回在指定位置的字符
substring(start,stop)	用于提取字符串中介于两个指定下标之间的字符

14.6.2 自定义函数

```
function functionname(argument1,argument2, ... , argumentn){函数体；}
```

14.6.4 函数变量的作用域

变量分为局部变量和全局变量
局部变量是指在函数内部声明的变量，只在一段程序中起作用的变量；全局变量是指在函数之外声明的变量，在整个JavaScript代码中都可起作用的变量，全局变量的生命周期从声明开始，在页面关闭时结束
局部变量和全局变量可以重名,在函数体内部，局部变量的优先级高于全局变量

专用于函数体内部的变量一定要用var关键字声明，否则该变量将被定义成全局变量

第十五章 JS事件分析

15.1 JS事件概述

事件编程：让用户不仅能够浏览页面中的内容，而且还可以和页面元素进行交互
事件-事件是可以被JavaScript侦测到的行为(ACTION)

15.1.1 事件类型

1. 鼠标事件
   例如单击button、选中checkbox和radio等元素；鼠标进入、悬浮或退出页面的某个热点：例如鼠标停在一个图片上方或者进入table的范围
2. 键盘事件
   当按下按键或释放按键时
3. HTML事件
   例如页面body被加载时；在表单中选取输入框或改变输入框中文本的内容：例如选中或修改了文本框中的内容

4. 突变事件
   主要指文档底层元素发生改变时触发的事件，如DomSubtreeModified(DOM子树修改)

15.1.2 事件句柄

事件句柄是事件发生要进行的操作

```
<标记 事件句柄=“JavaScript代码”…></标记>
如:<body onload=“show()”>…    </body>
```

事件分类	事件名称	事件句柄	事件
窗口事件	load	onLoad	当文档载入时执行JS代码
	unload	onUnload	当文档卸载时执行JS代码
表单元素事件	change	onChange	当元素改变时执行JS代码
	submit	onSubmit	当表单被提交时执行JS代码
	reset	onReset	当表单被重置时执行JS代码
	select	onSelect	当元素被选取时执行JS代码
	blur	onBlur	当元素失去焦点时执行JS代码
	focus	onFocus	当元素获得焦点时执行JS代码
鼠标事件	click	onClick	当鼠标被单击时执行JS代码
	dblclick	onDblclick	当鼠标被双击时执行JS代码
	mousedown	onMouseDown	当鼠标按钮被按下时执行JS代码
	mousemove	onMouseMove	当鼠标指针移动时执行JS代码
	mouseout	onMouseOut	当鼠标指针移出某元素时执行JS代码
	mouseover	onMouseOver	当鼠标指针悬停于某元素之上时执行JS代码
	mouseup	onMouseUp	当鼠标按钮被松开时执行JS代码
键盘事件	keydown	onKeyDown	当键盘被按下时执行JS代码
	keypress	onKeyPress	当键盘被按下后又松开时执行JS代码
	keyup	onKeyUp	当键盘被松开时执行JS代码

15.1.3 事件处理

当一个事件发生时，如果需要截获并处理该事件，只需要定义该事件的事件句柄所关联的事件处理代码，具体的处理方式有以下3种：

1. 静态指定

```
<HTML标记 ... 事件句柄1="事件处理程序" [事件句柄2 = "事件处理程序" ...]></H
```

2. 动态指定

```
<事件主角-对象>.<事件句柄>=<事件处理程序>
```

3. 特定对象特定事件的指定

```
   <script type=“text/javascript”for=“对象” event=“事件”>
  //事件处理程序代码
</script>
```

动态指定

事件处理程序在JavaScript中动态指定(分配)

```
<body>
<form name="myform" method="post" action="" >
   <input id=“input” type=“button” name=“mybutton" value="提交" >
</form>
 <script type="text/javascript">
  function clickHandler() {alert("即将提交表单！ "); return true;}
  //动态分配一个事件句柄
  document.getElementById(‘input’).onclick=function(){return clickHandler(
  myform.mybutton.onclick();
  </script>
 </body
```

特定对象的特定事件指定

```
<!doctype html>
<html lang="en">
 <head>
  <meta charset="UTF-8">
<title>给特定对象指定特定事件处理程序</title>
</head>
<body>
<h4>给特定对象指定特定事件处理程序</h4>
<script type="text/javascript" for="window" event="onload">
    alert("网页读取完成，欢迎光临！ ");
</script>
body>
</html>
```

15.1.4 事件处理程序的返回值

在JavaScript中通常事件处理程序不需要有返回值，这时浏览器会按默认方式进行处理
返回值类型：boolean布尔型值
浏览器根据返回值的类型决定下一步如何操作。当返回值为true，进行默认操作； 当返回值为 false，阻止浏览器的下一步操作
基本语法：事件句柄=“return 函数名（参数）;"

15.2 表单事件

事件分类	事件句柄	事件
表单元素事件	onchange	当元素改变时执行脚本
	onsubmit	当表单被提交时执行脚本
	onreset	当表单被重置时执行脚本
	onselect	当元素被选取时执行脚本
	onblur	当元素失去焦点时执行脚本
	onfocus	当元素获得焦点时执行脚本

15.2.1 获得及失去焦点事件

当表单中的元素获得焦点时会触发Focus获得事件，当表单中的元素失去焦点时会触发Blur失去焦点事件

15.2.2 提交及重置事件

表单的Submit事件触发后会将表单中的数据提交到服务器端，Reset事件触发后会将表单中的数据重置为初始值

15.3 鼠标事件

事件分类	事件句柄	事件
鼠标事件	onclick	当鼠标被单击时执行脚本
	ondblclick	当鼠标被双击时执行脚本
	onmousedown	当鼠标按钮被按下时执行脚本
	onmousemove	当鼠标指针移动时执行脚本
	onmouseout	当鼠标指针移出某元素时执行脚本
	onmouseover	当鼠标指针悬停于某元素之上时执行脚本
	onmouseup	当鼠标按钮被松开时执行脚本

15.4 键盘事件

事件分类	事件句柄	事件
键盘事件	onkeydown	当键盘被按下时执行脚本
	onkeypress	当键盘被按下后又松开时执行脚本
	onkeyup	当键盘被松开时执行脚本

15.5 窗口事件

第十六章 DOM和BOM

16.1 常用对象

JavaScript的对象类型（分为4类）：本地对象、内建对象、宿主对象、自定义对象

1. 本地对象就是ECMA-262定义的类（引用类型）包括：
   Object、Function、Array、String、Boolean、Number、Date、RegExp、Error、EvalError、RangeError、ReferenceError、SyntaxErro、TypeError、URIError等。
   这些对象独立于宿主环境，先定义对象，实例化后再通过对象名来使用
2. 内置对象(built-in object)。由ECMAScript实现提供的、不依赖与宿主环境的对象
   这意味着开发者不必明确实例化内置对象，它已被实例化了。ECMA-262只定义了两个内置对象，即Global和Math。Global是全局对象，全局对象只是一个对象，而不是类。既没有构造函数，也无法实例化一个新的全局对象
   isNam(),isFinite(),parseInt()和parseFloat()等，都是Global对象的方法。Math对象直接使用，如Math.Random()、Math.round(20.5)等
3. 宿主对象(host object)。ECMAScript实现的宿主环境提供的对象。所有BOM和DOM对象都是宿主对象。通过它可以与文档和浏览器环境进行交互，如document、window和frames等
4. 自定义对象。根据程序设计需要，由编程人员自行定义的对象。例如定义一个person对象，它有4个属性分别是firstName、lastName、age、eyeColor，同时给属性赋值。定义格式如下：

```
   var person=new Object();    /* 这是一种方法*/
    person.firstname="Bill";
    person.lastname="Gates";
    person.age=56;
    person.eyecolor="blue";
var person={firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};/*另
```

16.1.1 Array

Array对象用于在单个的变量中存储多个相同类型的值,但由于JavaScript是弱类型的脚本语言主，所以数组元素也可以不一致

1. 创建Array对象

```
var stu1=new array();
var stu2=new Array(size);
var stu3=new Array(element0, element1, ..., elementn);
var str4=[“C++程序设计”,“HTML开发基础”,“数据库原理””];
```

参数size定义数组元素的个数。返回的数组的长度stu2.length等于size
2. 数组的返回值
数组变量stu1、stu2、stu3返回新创建并被初始化了的数组。如果调用构造函数Array()时没有使用参数，那么返回的数组为空，数组的length为0。
3. 数组元素初始化与修改指定数组元素

```
   var stu = new Array();          /*先定义数组*/
stu[0] = "计算机组成原理";       /*给数组元素赋值*/
 stu[1] = "Java程序设计";        /*给数组元素赋值*/
 var len=stu.length;             /*len的值为2*/
 stu[1] = "创新创业教育";      /* 修改数组中第2个元素 */
```

4. 数组对象的属性和方法
    - 数组的下标从0开始，到数组的长度-1为止。
    - 如访问第一个元素的代码可以这样写：var cn = stu[0];
    - 数组下标可以用变量替代，但变更的取值范围必须符合数组下标的边界，否则返回undefined 。例如：var i = 3;var cn = stu[i];
    - 可以用再赋值的方式来修改数组对应位置的元素，如： var stu1[2] = “王建伟”;
      使用数组对象的属性和方法
      length：数组对象长度；
      join(separator)：把数组各个项用某个字符(串)连接起来，但并不修改原来的数组，默认用逗号分隔;
      pop()：删除并返回数组的最后一个元素
      push(newelement1,.,newelementX):可向数组的末尾添加一个或多个元素，并返回新的长度

shift()：用于把数组的第一个元素从其中删除，并返回第一个元素的值
unshift(newelement1,newelement2,....,newelementX)： 向数组的开头添加一个或更多元素，并返回新的长度
sort()、reverse()、toString()

16.1.2 Date对象

JavaScript脚本内置对象Date用于处理日期和时间。Date对象有很多方法，可以提取时间和日期

1. 创建日期对象

```
   var today=new Date();
var today=new Date(毫秒数);
var today=new Date(标准时间格式字符串);
var today=new Date(年,月,日,时,分,秒,毫秒);
```

提取日期字段方法

方法名	说明
getDate()	从Date对象返回一个月中的某一天(1～31)。
getDay()	从Date对象返回一周中的某一天(0～6)。
getMonth()	从Date对象返回月份(0～11)。
getFullYear()	从Date对象以四位数字返回年份。
getHours()	返回Date对象的小时(0～23)。
getMinutes()	返回Date对象的分钟(0～ 59)。
getSeconds()	返回Date对象的秒数(0～ 59)。
getMilliseconds()	返回Date对象的毫秒(0～999)。
getTime()	返回至今的毫秒数。

日期转换与调整
将日期转化为字符串

```
today.toString(); //把Date对象转换为字符串
today.toLocaleString();  //转换为本地时间串
today.toDateString();  //日期部分转为字符串
today.toTimeString();//时间部分转为字符串
```

调整日期对象的日期和时间

```
var today = new Date();
today.setDate(today.getDate()+5);  //将日期调整到5天以后，如果碰到跨年月，自动
today.setFullYear(2009,11,11);  //调整today对象到2009-11-11，月和日期参数可以
```

16.1.3 Math

```
var area=Math.PI*radius*radius ;//计算圆的面积
var r= Math.random();   //生成介于 0.0 - 1.0之随机数
var s3=sqrt(10); //10的平方根，值小于0，则返回 NaN。其他函数：
Math.max(x,y):返回 x 和 y 中的最高值。
var max=Math.max(100,200,300);//max=300
Math.min(x,y):返回 x 和 y 中的最低值。
var min=Math.min(1,45,67);//min=1
Math.pow(x,y):返回 x 的 y 次幂(xy)
ceil()：对数进行上舍入。   var x=Math.ceil(10.5);//x=11
floor()：对数进行下舍入。var x=Math.floor(10.5);//x=10
round()：把数四舍五入为最接近的整数。
      var x=Math.round(10.5); //x=10,
      var y=Math.round(-10.5);//y=-10
exp() ：返回 e 的指数。
      var x=Math.exp(1);//x=2.718
log()：返回数的自然对数（底为e）。
pow(x,y) ：返回 x 的 y 次幂。
产生某一区间数据方法：[m,n]
      var num=Math.floor(Math.random()*(n-m+1)+m)
      var num=Math.round(Math.random()*(n-m)+m)
      var num=Math.ceil(Math.random()*(n-m)+m)
```

16.1.4 Number

使用强制类型转换函数Number(value)可以把给定的值转换成数字

```
var ss=Number(false)              //返回值为0
var ss=Number(true)               //返回值为1
var ss=Number(null)               //返回值为0
var ss=Number(100)               //返回值为100
var ss=Number("5.5 ")             //返回值为5.5
var ss=Number("56 ")              //返回值为56
var ss=Number(undefined)         //返回值为NaN
```

```
var ss=Number("5.6.7 ")     //返回值为NaN，与parseFloat(“5.6.7”)不同
var ss=Number(new Object())      //返回值为NaN
```

16.1.5 String

String对象属于JavaScript核心对象之一。主要提供诸多方法实现字符串检查、抽取子串、字符串连接、字符串分割等字符串相关.
创建String对象:

```
var  s1 = "Welcome to you!";
var  s2 = new String("Welcome to you!");
```

强制类型转换String(Value)

```
var s1=String("100")        //返回值为字符串100
var s1=String("acdd")       //返回值为字符串acdd
var s1=String("false")       //返回值为字符串false
var s1=String(true)          //返回值为字符串true
var s1=String(null)          //返回值为字符串null
var s1=new Array("111","222","333");String(s1)   //返回值为111,222,333
var s1=String(new Object())        //返回值为字符串[object,Object]
```

1. 获取String对象长度属性
   String对象常用的属性有length，返回目标字符串中字符数目。

```
var s1 = "hello,world";
var len = s1.length; //s1.length返回11，s1所指向的字符串有11个字符
```

2.连接两个字符串
String对象的concat( )方法能将作为参数传入的字符串加入到调用该方法的字符串的末尾，并将结果返回给新的字符串。

```
var targetString=new String("Welcome to ");
var strToBeAdded=new String("the world!");
var finalString=targetString.concat(strToBeAdded);
```

3.把字符串分割为字符串数组
split()方法可以把字符串分割成字符串数组。

```
<script type=text/javascript>
var str1 = " How are you doing today?”
var subarray =str1.split(" ");  //subarray是一个数组
for(var i=0;i<subarray.length;i++)
{     document.write(subarray [i]);
      document.write("<br>");  }
 </script>
```

4. String对象的显示风格方法
   big()：变大些； small()：变小些；
   fontsize()：字体大小； fontcolor()：字体颜色；
   bold()：变粗些；italics()：斜体；
   sub()：下标； up()：下标
5. 字符串的大小写转换
   toLowerCase()：把字符串转换为小写
   toUpperCase()：把字符串转换为大写

16.1.6 Boolean

Boolean对象
var bo=new Boolean(value);
var bo1=Boolean(value);
参数为下列情况时返回true
1、true、”true”、”false”、”dfaf a”
参数为下列情况时返回flase
0、null、false、NaN、“”、空

16.2 HTML DOM

document对象是客户端JavaScript最为常用的对象之一，在浏览器对象模型中，它位于window对象的下一层级。
document对象包含一些简单的属性，提供了有关浏览器中显示文档的相关信息，例如：该文档的URL、字体颜色，修改日期等。
document对象还包含一些引用数组的属性，这些属性可以代表文档中的表单、图象、链接、锚以及applet。

同其他对象一样，document对象还定义了一系列的方法，通过这些方法，可以使JavaScript在解析文档时动态地将HTML文本添加到文档中。

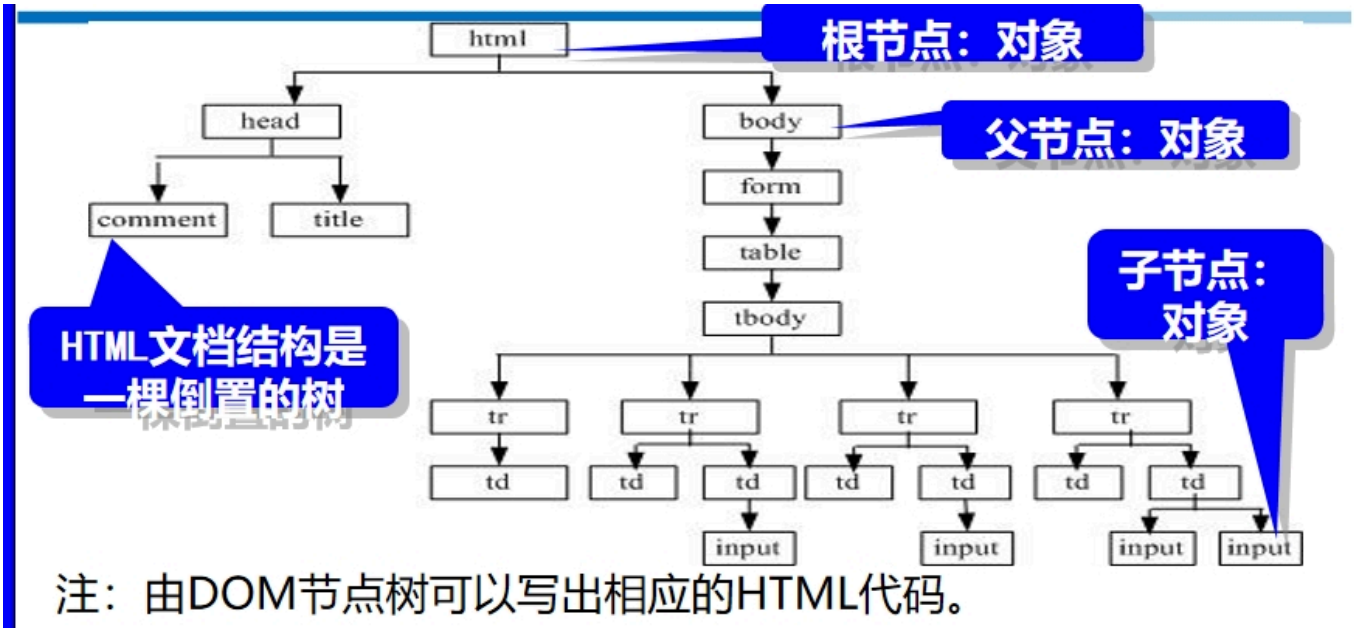
16.2.1 DOM简介

DOM模型

HTML DOM定义了访问和操作HTML文档的标准方法。

DOM将HTML文档表达为树结构。HTML文档结构好像倒垂的一棵树一样，其中<html>标记就是树的根节点，<head>、<body>是树的两个子节点。这种描述页面标记关系的树型结构称为DOM节点树（文档树）。

16.2.2 DOM节点树



16.2.3 DOM节点

根据HTML DOM规范，HTML文档中的每个成分都是一个节点。具体的规定如下：

- 整个文档是一个文档节点；
- 每个 HTML 标签是一个元素节点；
- 包含在 HTML 元素中的文本是文本节点；
- 每一个 HTML 属性是一个属性节点；
- 注释属于注释节点。

  根节点

```
var root=document.documentElement;
```

子节点

```
var aNodeList=root.childNodes;  //对象数组
```

节名（如根节点root）.firstChild、节名（如根节点root）. lastChild可获取一个节点的第一和最后一个子节点。节点有三个属性分别是nodeName、nodeType、nodeValue。
父结点

```
var parentNode=aNode.parentNode;
```

兄弟节点

```
var preNode=aNode.previousSibling;//返回前个节点引用
var nextNode=aNode.nextSibling;//返回后一个节点引用
```

关系
HTML文档来主要节点：

- 元素节点 (Element Node),
  <html>,<head>、<body>、<h1>、<p>和<ul>等标签都是元素节点。
- 文本节点(Text Node)
  文本节点包含在元素节点内，如h1、p、li等节点就可以包含一些文本节点。
- 属性节点(Attribute Node)
  属性节点总是被包含在元素节点当中，可以通过元素节点对象调用getAttribute( )方法来获取属性节点。

16.2.4 DOM节点访问

访问节点的方式很多种，可以通过document对象的方法来访问节点，也可以通过元素节点的属性来访问节点。

- 通过getElementById( )方法访问节点

```
var s=document.getElementById(id);
```

```
- 最好为需要交互的元素设定一个唯一的id，以便查找；
- getElementById()返回的是一个页面元素的引用.
- 元素ID引用时，必须加引号，如“userName”，否则易引起语法错误，缺少对象。
```

- 通过getElementsByName ( )方法访问节点

```
var s=document.getElementByName(name);
```

```
-该方法返回一个数组，引用元素必须通过下标进行；
- 如果返回一个长度为0的数组，没有元素；可以通过对象的length属性来判断。
```

- 通过getElementsByTagName ( )方法访问节点

```
var s=document.getElementByTagName(tagname);
```

- 通过form元素访问节点

```
var loginform = document.myform;//myform为form对象的名称
var username1=loginform.elements[0];//通过elements属性来访问
var username2=loginform.username;//通过name属性来访问
```

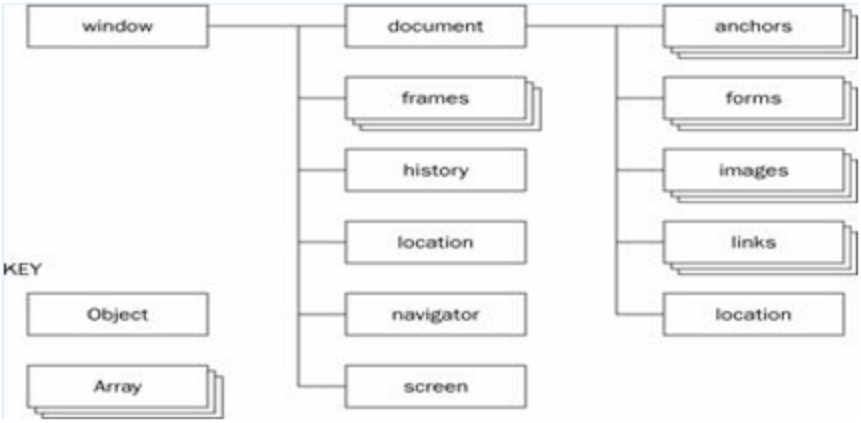
16.2.5 DOM节点操作

创建与修改节点

方法名	说明
createElement(tagname)	创建标记名为tagname的节点。
createTextNode(text)	创建包含文本text的文本节点。
createDocumentFragment()	创建文档碎片。
createAttribute()	创建属性节点。
createComment(text)	创建注释节点。
removeChild(node)	删除一个名为node的子节点。
appendChild(node)	添加一个名为node的子节点。
insertBefore(nodeB,nodeA)	在名为nodeA节点前插入一个名为nodeB的节点
replaceChild(nodeB,nodeA)	用一个名为nodeB节点替换另一个名为nodeA节点
cloneNode(boolean)	克隆一个节点，它接收一个boolean参数，为true时表示该节点带文字；false表示该节点不带文字

16.3 BOM

客户端浏览器这些预定义的对象统称为浏览器对象，它们按照某种层次组织起来的模型统称为浏览器对象模型（BOM-Browser Object Model）。浏览器对象模型 (BOM) 使 JavaScript 有能力与浏览器“对话”。
BOM提供了独立于内容而与浏览器窗口进行交互的对象。BOM由一系列相关的对象构成，并且每个对象都提供了很多方法与属性。



16.3.1 window对象

window对象位于浏览器对象模型的顶层，是document、frame、location等其他对象的父类。window对象的常用方法如下表

方法名	说明
open(url,name,features,replace)	打开新的浏览器窗口或查找一个已命名的窗口。
prompt(“提示信息”，默认值)	显示可提示用户输入的对话框。
blur()	把键盘焦点从顶层窗口移开。
close()	关闭浏览器窗口。
focus()	把键盘焦点给予一个窗口。
setInterval(code,interval)	按照指定的周期（以毫秒计）来调用函数或计算表达式。
setTimeout(code,delay)	在指定的毫秒数后调用函数或计算表达式。
clearInterval(intervalID)	取消由setInterval()设置的timeout。
clearTimeout(timeoutID)	取消由setTimeout()方法设置的timeout。

16.3.2 navigator对象

navigator对象用于获取用户浏览器的相关信息
navigator属性

属性名	说明
appName	返回浏览器的名称。
appVersion	返回浏览器的平台和版本信息。
platform	返回运行浏览器的操作系统平台。
systemLanguage	返回操作系统使用的默认语言。
userAgent	返回由客户机发送服务器的user-agent头部的值。
appCodeName	返回浏览器的代码名。
appName	返回浏览器的名称。
appVersion	返回浏览器的平台和版本信息。

16.3.3 screen对象

screen对象用于获取用户屏幕设置的相关信息，具有height、width、availHeight、availWidth等属性。

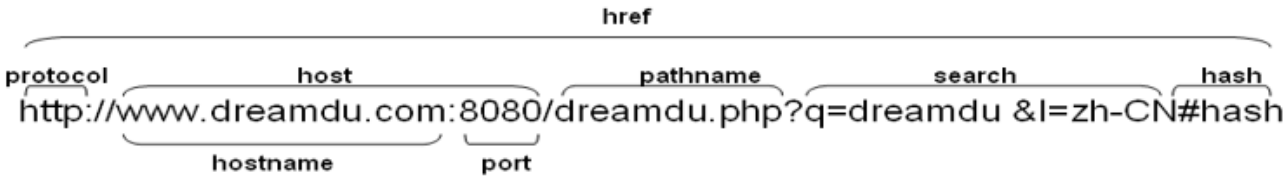
16.3.4 history对象

history对象表示窗口的浏览历史，并由window对象的history属性引用该窗口的history对象。

方法名	说明
forward()	加载history列表中的下一个 URL。
back()	加载history列表中的前一个 URL。
go(number|URL)	加载history列表中的某个具体页面。URL参数指定要访问的URL，number参数指定要访问的URL在history的URL列表中的位置。

16.3.5 location对象

location对象用来表示浏览器窗口中加载的当前文档的URL，该对象的属性说明了URL中的各个部分



属性名	说明
hash	设置或返回从井号(#)开始的URL(锚)
href	设置或返回完整的URL
hostname	设置或返回URL中的主机名
protocol	设置或返回当前URL的协议
port	设置或返回当前URL的端口号
Pathname	设置或返回当前URL的路径部分
Host	设置或返回URL中的主机名和端口号的组合
Search	设置或返回从问号(?)开始的 URL(查询部分)

方法名	说明
reload()	重新加载当前文档。
assign()	加载新的文档。
replace()	用新的文档替换当前文档。

第七章 CSS基础

7.1 CSS概念

7.1.1 CSS的基本概念

CSS（Cascading Style Sheet）层叠样式表，也称为级联样式表，用来设计网页风格
在网页制作时采用CSS技术，可以有效地对页面的布局、字体、颜色、背景和其他效果实现更加精确的控制
只要对相应的代码做一些简单的修改，就可以改变同一页面的不同部分，或者不同网页的外观和格式

7.1.2 传统HTML的缺点

维护困难；标记不足；网页过“胖”；定位困难

7.1.3 CSS的特点

CSS 通过定义标记或标记属性的外在表现，对页面结构风格进行控制，分离文档的内容和表现，克服了传统HTML 的缺点。将CSS 嵌入在页面中，通过浏览器解释执行，而且CSS文件是文本文件，只要理解了HTML 就可以掌握它

7.1.4 CSS的优势

1. 表现和内容相分离
   CSS通过定义HTML标记的样式，使得页面 内容和显示相分离，简化了网页格式设计，也使得对网页格式的修改更方便。
2. 加强了网页的表现力
   CSS样式属性提供了比HTML更多的格式设计功能。例如，去掉网页超级链接的下划线、给文字添加阴影等
3. 增强了网站风格的一致性
   CSS样式定义到样式表文件中，在多个网页中同时应用样式表文件中的样式，就确保了多个网页具有一致的格式。可以随时更新样式表文件，改变网站风格。大大降低了网站的开发与维护工作

7.1.5 CSS编辑方法

1. 写在HTML文件里；
2. 写在独立的*.CSS文件里。

7.2 使用CSS控制Web页面

CSS控制页面是通过CSS规则实现的，**CSS规则由选择器和声明组成，声明由属性和属性值对组成**
CSS提供了丰富的选择器类型，包括标记选择器、类选择器、id选择器及伪类选择器等，能够灵活地对整个页面、页面中的某个标记或一类标记进行样式设置
此外，在HTML页面中应用CSS规则的方式也比较灵活，包括行内(内联)样式表、内部样式表、链入外部样式表及导入外部样式表

7.2.1 CSS基本语法

CSS是一个由包含一个或多个规则的文本文件
规则： 选择符Selector+ 声明部分Declaration
选择器通常是需要改变样式的HTML标记。
声明由一个或多个属性名称与属性值对组成。

```
<style type="text/css">
   /*  定义body样式  */
    body{background:black;color:red;}
    .div{padding:50px;}
    .pic{float:right;   padding:20px;}
</style>
```

CSS注释方法
注释不能嵌套。
注意与HTML注释方法不同。

7.2.2 CSS选择器类型

CSS选择器类型：5种

- 标记选择器
  标记选择符----对HTML的标记重定义。该样式立即生效

```
p,h1{font-size:30px;color:blue;font-family:黑体;}
```

- 类选择器
  以点号“.”开头，并可以任意命名，如.div1、.files等，该样式应用后生效，有些标记的样式相同时，可以定义成选择符组

```
 .div1,.file{background:red;color:white;}
```

联合选择器---标记+类选择器（p.c3{color:red;}）

- id选择器

```
#div1{background:red;color:white;}
```

类选择器和ID选择器的区别
ID选择符与类选择符的区别：
（1）类选择符可以给任意多的标记定义样式，但ID选择符在页面中标记中只能使用一次；
（2）ID选择符样式比类选择符样式优先级高。ID选择符局限性大，只能单独定义某个元素的样式（特殊情况下使用）。

- 伪类选择器
  一种特殊的类选择符，最大的作用就是对链接<a>的不同状态定义不同的样式效果

```
    a:link{color:#339999;text-decration:none;}
    a:visited{color:#33cc00;text-decration:none}
```

```
a:hover{color:red;text-decration:underline;}
a:active{color:blue;text-decration:underline;}
```

- CSS属性选择器

选择器	描述
[attribute]	用于选取带有指定属性的标记。
[attribute=value]	用于选取带有指定属性和值的标记。
[attribute~=value]	用于选取属性值中包含指定词汇的标记(用空格分隔的字词列表)。
[attribute|=value]	用于选取带有以指定值开头的属性值的标记(属性值是value或者以“value-”开头的值)。
[attribute^=value]	匹配属性值以指定值开头的每个标记。
[attribute$=value]	匹配属性值以指定值结尾的每个标记。
[attribute*=value]	匹配属性值中包含指定值的每个标记。

7.2.3 CSS选择器声明

1. 集体声明

```
h1,h2,h3,h4,h5,h6,p,h2.special,#one{color:red;font-family:黑体;}
```

2. 全局声明 –通配符

```
*{color:purple;font-size:16px;margin:0 auto;padding:0;}
```

3. 派生选择符

```
li strong{ font-style:italic; font-weight:normal;}
   strong{font-weight:bold;}
```

7.2.4 CSS定义与引用

CSS样式表类型：4种

- 内联样式表（Inline Style Sheet）

```
<标记 style="属性: 属性值; 属性: 属性值;…">
```

```
  标记：HTML标记，如body、table、p等；
  标记的style定义只能影响标记本身；
  style的多个属性之间用分号分割；
  标记本身定义的style优先于其他所有样式定义。
  行内样式表只影响单个元素（标记）

- 内部样式表（Internal Style Sheet）
  Style标记是双标记，有type属性,必须放在头部。
  内部样式表只影响单个文件
'''css
<head>
<style type="text/css">
.div1,.div3{background:#99ffff;width:200px;
                            height:100px;}
         #div2{background:#00cc00;width:200px;
                               height:100px;}
  p,h1{font-size:18px; color:#003366;}
</style>
</head>
```

- 链接外部样式表（Link External Style Sheet）
  import语句后的“;”号，一定要加上！
  “外部样式表的文件名称”是要嵌入的样式表文件名称，含路径，后缀为.css；
  @import应该放在style元素的最前面。

```
    <style type="text/css">
       @import url("外部样式表的文件名称");
       p,p1{font-size:18px; color:blue}
   </style>
```

- 导入外部样式表（Import External Style Sheet）
  <link>标记是单标记，放在头部，不使用style标记。
  外部样式表的文件名称必须带后缀名.css。
  CSS文件一定是纯文本格式。
  外部样式表修改后所有引用的页面样式自动地更新；

外部样式表优先级低于内部样式表；
同时链接几个外部样式表时按“最近优先的原则”。

```
<link type="text/css" rel="stylesheet" href="外部样式表的文件名称"/ >
```

7.3 CSS继承与层叠

样式表的继承规则是子标记继承父标记的样式
CSS规定样式的优先级(从高到低)如下
行内样式 > id样式 > 类样式 > 标记样式

第八章 DIV与SPAN

8.1.1 DIV(divsion/section)定义

是一个块级(block-level)元素，其前后均有换行符，可定义文档中的分区或节

```
<div id="layer1" class=“” style ="position:absolute;
      left:29px; top:12px; width:135px; height:24px;
      background:#99cccc; border:2px dashed #ffff00;">
  块包含的内容</div>
```

图层CSS属性

- position:定位，static|absolute|relative |fixed
- width|:height图层宽度|图层高度
- left| top:左边距|顶部距离
- border:边框，“线粗细 线形状 线颜色”
- z-index:图层层叠，子层永远在父层之上，值越大越在上层,前提条件是position属性值为“absolute”。
- clear:left|right|both|none，清除左、右、两边及允许浮动。
- float:left|right|none，允许左、右、不浮动

8.2 图层嵌套与层叠

图层包含其它图层，称为图层的嵌套
图层嵌套经常需要与CSS样式一起使用，达到更加精确控制页面显示效果

```
<div id="wrap">
     <div id="d1" class="inline_div">div1</div>
     <div id="d2" class="inline_div">div2</div>
     <div id="d3">div3</div>
</div>
```

层叠

```
<!-- edu_8_2_2.html -->
<!doctype html>
<html lang="en">
<head>
<meta charset="UTF-8">
<style type="text/css">
body{margin:0;  /*margin表示边距，在8.5CSS盒模型介绍*/}
div{position:absolute; /* 定位方式为绝对定位 */
width:200px;height:200px;}
#d1{background-color:black;
z-index:0;    /* 该图层在最下面 */color:white;}
#d2{background-color:red;top:25px;left:50px;z-index:1; /*图层在中间 */}
#d3{background-color:yellow;top:50px;left:100px;
z-index:2;    /* 该图层在最上面 */}
</style>
</head>
<body>
<div id="d1" >div1</div>
<div id="d2" >div2</div>
<div id="d3" >div3</div>
</body>
</html>
```

8.3 div标记和span标记

span标记—行内元素

```
<span id="" class="">行内内容</span>
```

Div与span标记使用区别

div和span标记默认情况下都没有对标记内的内容进行格式化或渲染，只有使用CSS来定义

相应的样式时才会显示出不同

1. 是否是块标记。div标记是块标记，一般包含较大范围，在区域的前后会自动换行；而span标记是行内标记，一般包含范围较窄，通常在一行内，在此区域的范围外不会自动换行
2. 是否可以互相包含。一般来说，div标记可以包含span标记，但span标记不可能包含div标记
3. 但是块标记和行标记不是绝对的，通过定义CSS的display属性可以相互转化

CSS中的display属性
display： 规定元素应该生成的显示框的类型
这个属性用于定义建立布局时元素生成的显示框类型。对于 HTML 等文档类型，如果使用 display 不谨慎会很危险，因为可能违反 HTML 中已经定义的显示层次结构

none	此元素不会被显示。
block	此元素将显示为块级元素，此元素前后会带有换行符。
inline	默认。此元素会被显示为内联元素，元素前后没有换行符。
inherit	规定应该从父元素继承 display 属性的值。

第九章CSS 样式属性

9.1 CSS属性值中的单位

绝对单位
绝对单位在网页中很少使用，一般多用在传统平面印刷中，但在特殊场合使用绝对单位是很必要时的
绝对单位包括：英寸、厘米、毫米、磅和pica（皮卡），其对应的英文单位分别是in（1in=2.54cm）、cm、mm、 pt（1pt=1/72in）、pica（pc，1pc=12pt）
相对单位
相对单位与绝对单位相比显示大小不是固定的，它所设置的对象受屏幕分辨率、或视觉区域、浏览器设置以及相关元素的大小等因素影响
经常使用的相对单位包括：em、ex、px、%。

9.2 CSS字体样式

CSS属性命名规范：CSS样式属性由多个单词构成时，单词之间使用“-”连字符连接

- font-size设置字号

```
.p1 {font-size:2cm}
 p{font-size:14pt;}
 .p4 {font-size:medium}
 .p6 {font-size:x-large}
 p{font-size:1.5em;}
.b{font-size:200%;}
```

绝对大小：in、cm、mm、pt、pc为单位，如16pt。
相对大小：em、ex、px、%。
例如font-size：1.5em|150%
使用关键字来指定大小：如xx-small | x-small | small | medium | large | x-large | xx-large。

- font-family设置字体

```
font-family:字体1,字体2,…,字体n
```

浏览器依次查找字体，只要存在就使用该字体，不存在将会继续找下去，以此类推，直到最后一种字体，仍不存在则使用默认字体（宋体）
如果字体名称中出现空格，必须使用双引号将字体括起来

- font-style设置字体样式

```
  font-style: normal | italic | oblique
```

```
normal
italic (斜体显示)
oblique(倾斜字体显示)
```

- font-weight设置字体加粗

```
font-weight: normal | bold | bolder | lighter |100|200|…|900
```

100-900(9个层次,数字越小字体越细、数字越大字体越粗)

- font-variant设置字体变体
  font-variant属性用于设置字体变体，主要用于设置英文字体,实际上是设置文本字体是

否为小型的大写字母

```
font-variant: normal | small-caps
```

- font设置综合字体属性
  font属性是复合属性，一次完成多个字体属性的设置,包括字体粗细、风格、字体变体、大小/行高及字体名称

```
font:font-style font-weight font-variant font-size/line-height font-famil
```

利用font属性一次完成多个字体属性的设置，属性值与属性值之间必须使用空格隔开。
前三个属性值可以不分先后顺序，默认为normal。
大小和字体名称系列必须显式指定，先设置大小，再设置字体系列。需要设置行高时，可以写在字体大小的后面，中间用“/”分隔，行高为可选的属性。font属性可以继承。

9.3 CSS文本样式

- **letter-spacing**:
  - 用于设置字母之间的间距。
  - 取值：
    - `normal`: 默认间距。
    - 长度单位（如 `px`, `em` 等）：指定具体的间距。
- **line-height**:
  - 控制文本行与行之间的高度。
  - 取值：
    - `normal`: 默认行高。
    - 比例（如 `1.5`）：相对于字体大小的倍数。
    - 长度单位（如 `20px`）：指定固定的行高。
    - 百分比（如 `150%`）：相对于字体大小的百分比。
- **text-indent**:
  - 定义段落首行的缩进。
  - 取值：
    - 长度单位（如 `20px`）：指定具体的缩进。
    - 百分比单位（如 `50%`）：相对于包含块的宽度。
- **text-decoration**:
  - 用于设置文本的装饰效果。

    - 取值：
        - `none`: 无装饰。
        - `underline`: 下划线。
        - `overline`: 上划线。
        - `line-through`: 删除线。
- **text-transform**:
    - 控制文本的大小写转换。
    - 取值：
        - `uppercase`: 全部大写。
        - `lowercase`: 全部小写。
        - `capitalize`: 首字母大写。
        - `none`: 不进行任何转换。
- **text-align**:
    - 定义文本的对齐方式。
    - 取值：
        - `left`: 左对齐。
        - `right`: 右对齐。
        - `center`: 居中对齐。
        - `justify`: 两端对齐。
- **vertical-align**:
    - 控制元素的垂直对齐方式，通常用于表格单元格或行内元素。
    - 取值：
        - `top`: 顶部对齐。
        - `middle`: 垂直居中对齐。
        - `bottom`: 底部对齐。

9.4 CSS颜色与背景

9.4.1 颜色color属性

color属性用于设置元素字体的色彩，该属性的语法比较简单，但取值比较多样，可以是颜色名称、函数、十六进制数等形式
颜色名称。使用red、blue、yellow等CSS预定义的表示颜色的参数。CSS预定义17种颜色，如下表所示。
RGB()函数。使用rgb（RRR, GGG, BBB）或rgb（r%, g%, b%）,字母R或r、G或g、B或b分别表示颜色分量红色、绿色、蓝色，前者参数的取值为0到255，后者参数的取值为0到

100。
十六进制数。使用“#RRGGBB”或者“#RGB”的形式，其中每个位十六进制数从0到F取值，比如#FFC0CB表示pink。

9.4.2 背景background属性

- **background-color**:
  - 设置元素的背景颜色。
  - 取值：
    - 关键字（如 `red`, `blue` 等）。
    - RGB 值（如 `rgb(255, 0, 0)`）。
    - `transparent`: 透明背景。
- **background-image**:
  - 用于设置元素的背景图像。
  - 取值：
    - `url(*.jpg)`: 引用图像文件（可以是 JPEG、PNG 等格式）。
    - `none`: 不使用背景图像。
- **background-attachment**:
  - 控制背景图像的滚动行为。
  - 取值：
    - `scroll`: 背景图像随页面滚动。
    - `fixed`: 背景图像固定在视口，滚动时不会移动。
- **background-repeat**:
  - 定义背景图像的重复方式。
  - 取值：
    - `repeat`: 背景图像在水平和垂直方向上都重复。
    - `repeat-x`: 背景图像仅在水平方向上重复。
    - `repeat-y`: 背景图像仅在垂直方向上重复。
    - `no-repeat`: 背景图像不重复，仅显示一次。
- **background-position**:
  - 设置背景图像的位置。
  - 取值：
    - 百分比（如 `50% 50%`）：相对于元素的宽高定位，`50%` 表示居中。
    - 长度（如 `10px 20px`）：指定具体的 x 和 y 位置。
    - 关键字（如 `top left`, `center`, `bottom right`）：使用预定义位置。

9.5 CSS列表样式

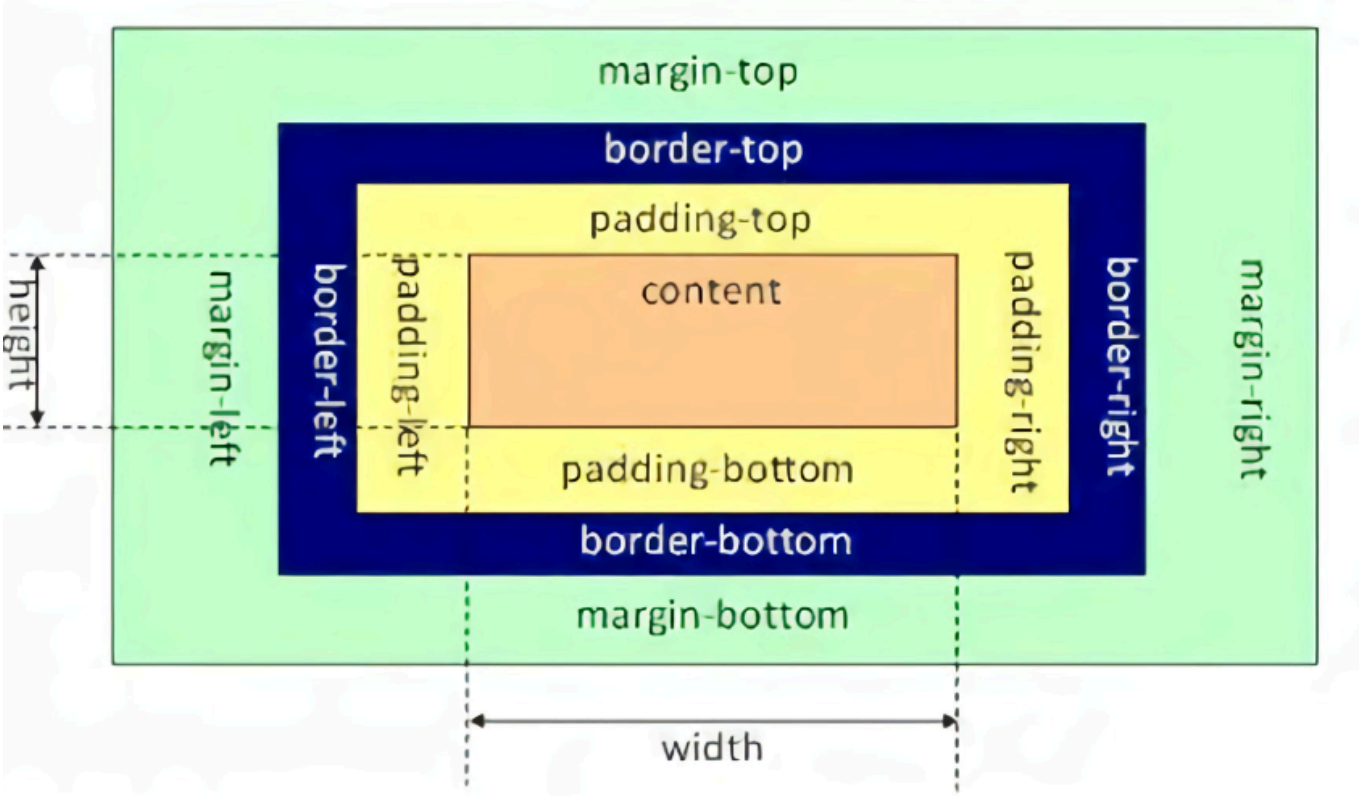
- **list-style-type**:
  - 用于定义列表项的标记样式，适用于有序列表（`<ol>`）和无序列表（`<ul>`）。
  - 取值：
    - `disc`: 实心圆点（无序列表默认样式）。
    - `circle`: 空心圆点。
    - `square`: 实心方块。
    - `decimal`: 数字（1, 2, 3...，有序列表默认样式）。
    - `lower-roman`: 小写罗马数字（i, ii, iii...）。
    - `upper-roman`: 大写罗马数字（I, II, III...）。
    - `lower-alpha`: 小写字母（a, b, c...）。
    - `upper-alpha`: 大写字母（A, B, C...）。
    - `none`: 不显示标记。
- **list-style-image**:
  - 用于设置列表项的标记为自定义图像。
  - 取值：
    - `url(*.gif)`: 指定图像文件作为列表标记（可以是 GIF、PNG 等格式）。
    - `none`: 不使用自定义图像，使用默认标记。
- **list-style-position**:
  - 控制列表项标记的位置。
  - 取值：
    - `inside`: 标记在列表项内容的内部，内容会环绕标记。
    - `outside`: 标记在列表项内容的外部，内容不会环绕标记（默认值）。

9.6 CSS盒模型

9.6.1 CSS盒模型结构

W3C组织就建议把所有网页上的对象都放在一个盒(box)中，设计师可以通过创建定义来控制这个盒的属性，这些对象包括段落、列表、标题、图片以及层
盒模型主要定义四个区域MBPC：边界(margin) 、边框(border) 、填充(padding)和内容

(content)



9.6.2 边界属性设置

边界属性是margin，也称为外边距，表示盒子边框与页面边界或其他盒子之间的距离，属性值为长度值、百分数或auto，属性效果是围绕元素边框的“空白”

```
margin-top: 20px; margin-right : 20px;
 margin-bottom : 20px; margin-left : 20px;
margin:10px; /* 4个边均为10px */
margin:10px 20px ; /* 上下|左右*/
margin:10px 20px 30px; /* 上|右左|下*/
margin:10px 20px 30px 40px; /* 上|右|下|左*/
```

可以通过单边属性进行设置

9.6.3 边框属性设置

- **border-width**:
  - 用于设置边框的宽度，可以通过关键字或具体长度来定义。
  - 取值：
    - `thin`: 细边框。

        - `medium`: 中等宽度边框（默认值）。
        - `thick`: 粗边框。
        - 长度单位（如 `1px`, `2em` 等）：指定具体的边框宽度。
    - 该属性为复合属性，可拆分为四个子属性：
        - `border-top-width`: 上边框宽度。
        - `border-right-width`: 右边框宽度。
        - `border-bottom-width`: 下边框宽度。
        - `border-left-width`: 左边框宽度。
- **border-style**:
    - 用于设置边框的样式。
    - 取值：
        - `none`: 无边框。
        - `dotted`: 虚线边框（点状）。
        - `dashed`: 虚线边框（短划线）。
        - `solid`: 实线边框。
        - `double`: 双线边框。
        - `groove`: 凹型边框，表示立体效果。
        - `ridge`: 凸型边框，表示立体效果。
        - `inset`: 嵌入式边框，给人一种深度效果。
        - `outset`: 突出式边框，给人一种浮起的效果。
    - 该属性也是复合属性，分为四个子属性：
        - `border-top-style`: 上边框样式。
        - `border-right-style`: 右边框样式。
        - `border-bottom-style`: 下边框样式。
        - `border-left-style`: 左边框样式。
- **border-color**:
    - 用于设置边框的颜色。
    - 取值：
        - 颜色关键字（如 `red`, `blue` 等）。
        - RGB 值（如 `rgb(255, 0, 0)`）。
- **border**:
    - 该属性为复合属性，简写形式，允许同时设置边框的宽度、样式和颜色。
    - 语法示例：
        - `border: 2px solid #ff33ee;`

- 这里 `2px` 是边框宽度，`solid` 是边框样式，`#ff33ee` 是边框颜色。

9.6.4 填充属性设置

元素内边界主要是指边框和内部元素之间的空白距离，利用padding属性设置元素内的边界时，也包括5个属性，同样也有四种设置方法

```
padding:长度 | 百分比
padding-top、padding-right、padding-bottom:同上
padding:20px 30px 40px 60px;/**/
padding:20px 30px 40px; /* 上|右|下|左*/
padding:20px 30px; /* 上|左右|下*/
padding:20px; /* 上右下左均相同*/
```

第十章 CSS+DIV页面布局

10.1 页面布局设计

现在所有的主流的、大型的IT企业的网站布局几乎都采用DIV、CSS技术，有些甚至采用DIV、CSS、表格混合进行页面布局
CSS布局的步骤大致为：
首先整体上对页面进行分块
接着按照分块设计使用div标记，并理清div标记的嵌套和层叠关系
然后对各div标记进行CSS定位，最后在各个分块中添加相应的内容

10.1.1 “三行模式”或“三列模式”

10.1.2 “三行二列”、“三行三列”模式

10.1.3 多行多列复杂模式

没啥意思，自己学

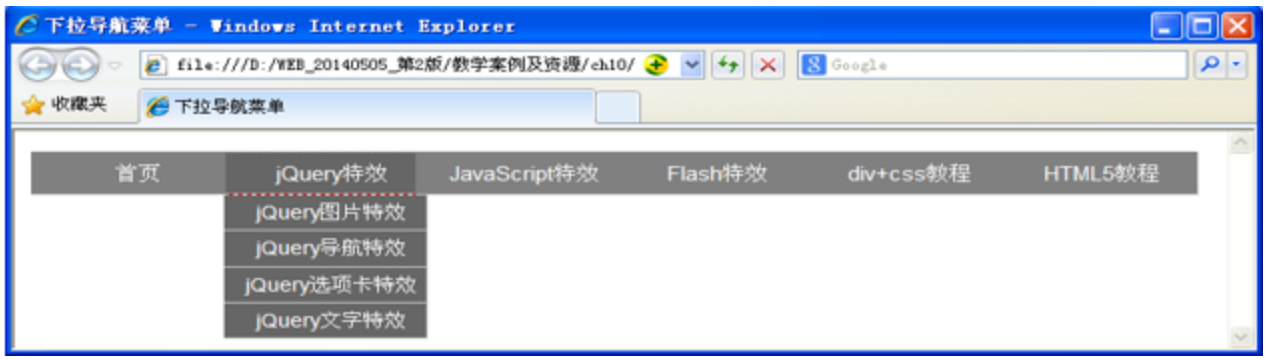
10.2 导航菜单设计

10.2.1 一级水平导航菜单

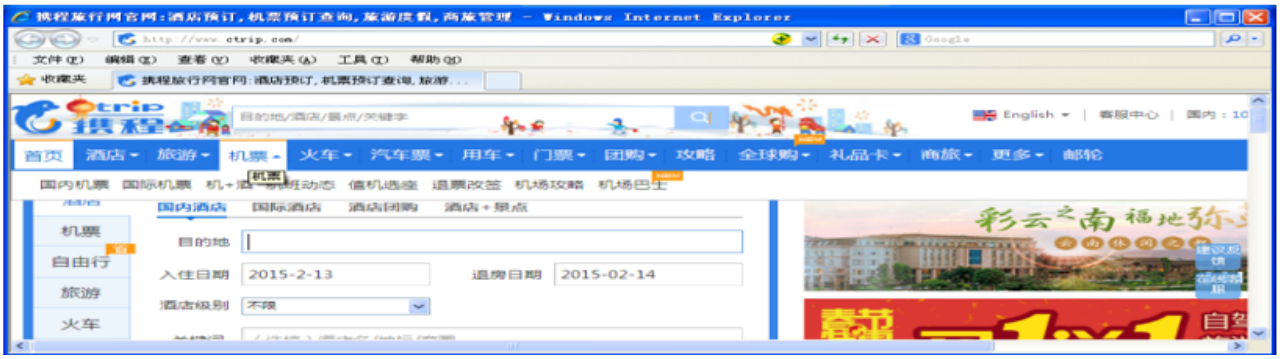
1. 采用“表格+超链接”来设计
2. 采用“无序列表+超链接”来设计

10.2.2 二级水平导航菜单

1. 下拉导航菜单
   采用
、
、<a>等标记和CSS样式定义来实现一个简单的二级下拉菜单的设计过程



2. 横向二级导航菜单
   所谓横向二级导航菜单，即一层主菜单是水平排列、二层子菜单也是水平排列，各占一行，其中二层子菜单可能会占多行，取决于子菜单的数量



CSS3

CSS3 浏览器兼容性

常用的浏览器属性前缀

为了让CSS样式能够满足不同浏览器版本的需要，需要在样式属性前面增加一些区分不同浏览器的前缀

-webkit-：适用于webkit 核心浏览器，包含Safari、Chrome 等。

-moz-：适用于Firefox 浏览器等。

-ms-：适用于IE 浏览器。

-o-：适用于Opera 浏览器

为了满足不同浏览器对CSS3新特性的支持，需要如下声明：

```
Div{-moz-animation: myfirst 5s;         /* Firefox */
      -webkit-animation: myfirst 5s;      /* Safari 和 Chrome */
```

```
    -o-animation: myfirst 5s;              /* Opera */
    -ms-animation: myfirst 5s;            /* IE */
    animation: myfirst 5s;                   /* 标准属性写在最后，*/
}
```

CSS3 前缀解决方案

为了简化开发过程和相关的代码冗余问题，在页面中引入了-prefix-free 这个类库，可以自动帮助在CSS 中添加相关的浏览器特有的前缀属性。

-prefix-free 是一个JavaScript 工具库，用户再也不需要编写带有浏览器前缀的CSS代码，在需要的时候，-prefix-free 会自动帮助添加当前浏览器需要的前缀。引用方式如下：

```
    <script src="http://cdn.gbtags.com/prefixfree/1.0.7/prefixfree.min.j
<script src=”http://leaverou.github.com/prefixfree/prefixfree.min.js“></sc
    <script src="js/prefixfree.min.js"></script>
```

CSS 样式重置方案

Normalize.css 是一个很小的CSS 文件，但它在默认的HTML 元素样式上提供了跨浏览器的高度一致性。相比于传统的CSS reset，Normalize.css 是一种现代的、为HTML5 准备的优质替代方案。

CSS3 边框

CSS3具有3个边框属性

属性	描述
border-image	设置所有border-image-*属性的复合属性。
border-radius	设置所有四个border-*-radius属性的复合属性。
box-shadow	向矩形方框添加一个或多个阴影。

1. border-radius圆角边框

```
border-radius: 水平半径(1-4个值)px|%/垂直半径(1-4个值)px|%;
            border-radius:2em;                  /* 等同于下列4行定义  */
            border-top-left-radius:2em;           /* 定义左上角半径  */
    border-top-right-radius:2em;         /*定义右上角半径*/
    border-bottom-right-radius:2em;     /*定义右上角半径  */
    border-bottom-left-radius:2em;       /*定义左下角半径  */
```

2. box-shadow 边框阴影

值	描述
h-shadow	必需。水平阴影的位置。允许负值。
v-shadow	必需。垂直阴影的位置。允许负值。
blur	可选。模糊距离。
spread	可选。阴影的尺寸。
color	可选。阴影的颜色。请参阅 CSS 颜色值。
inset	可选。将外部阴影 (outset) 改为内部阴影。

```
box-shadow: h-shadow v-shadow blur spread color inset;
            box-shadow: 0 0 30px 20px #6699ff inset;     /*内部阴影 */
            box-shadow: 0px 0px 45px 10px #9999ff;      /*外部阴影 */
            box-shadow: 20px 20px 35px 15px #99ff33; /* 外部阴影 */
```

3. border-image 边框图像

值	描述
border-image-source	规定边框中图像的路径。
border-image-slice	规定图像边框向内偏移，可以是数字或百分比。
border-image-width	规定图像边框的宽度。
border-image-outset	规定边框图像区域超出边框的量。
border-image-repeat	规定图像边框是否应平铺(复制)、铺满(环绕)或拉伸。

```
   border-image:border-image-source
    border-image-slice/border-image-width/ border-image-outset border-imag
    border-image: url("border.png") 27 27 27 27 fill/27 27 27 27/27px 27px
```

- border-image-source 属性（边框图像）
  - 默认无边框图像，如果设置边框图像，则使用绝对或相对url 地址指定边框图像
- border-image-slice 属性（图像切片/剪裁）
  - 该属性规定图像边框向内偏移，可以是数字或百分比。可以1～4 个值，类似于padding属性的设置方法

```
      border-image-slice: number |% |fill;
      border-image-slice:27 27 27 27; /* 边框图像切9块，每个角为27px*27px*/
```

- border-image-repeat: stretch | round | repeat
  - stretch（拉伸）、round（环绕）、repeat（复制）
- border-image-width 属性（边框图像宽度）

```
border-image-width: number |% ;   /* 可以有1～4个值*/
   border-image-width:27px 1 10% 27px;
```

- border-image-outset 属性（图像外凸）

```
border-image-outset: length | number | percentage | auto; /* 1～4个值 */
```

CSS3 转换transform 属性

CSS3 2D 转换常用方法有translate()、rotate()、scale()、skew()、matrix()。

- 位移translate(x,y)
  translate(x,y)方法的作用是将元素从当前位置根据给定的x 轴坐标和y 轴坐标进行移动。x 表示left，父元素的左边界；y 表示top，父元素的上边界。translate()方法还提供根据单一轴移动的方法，分别是translateX()和translateY()。使用方法如下：

```
transform:translate(50px,50px); /* 向右移动50px，向下移动50px */
transform:translate(50px,0); /* 向右移动50px */
transform:translateX(50px); /* 向右移动50px */
transform:translate(0,50px); /* 向下移动50px */
transform:translateY(50px); /* 向下移动50px */
```

- 旋转rotate(deg)
  可以对元素旋转给定的角度，正值为顺时针，负值为逆时针

```
transform:rotate(deg);          /* 基本语法 */
transform:rotate(10deg);      /* 旋转10° */
transform:rotate(120deg);    /* 旋转120° */
```

- 缩放scale(x, y)
  参数x 表示元素宽度的缩放倍数，参数y 表示元素高度的缩放倍数
  scale 方法也可以接受负值，当参数x 为负值时，元素内容会横向倒置；当参数y 为负值时，元素内容会纵向倒置

```
transform:scale(x,y);
transform:scale(1,4);
```

```
transform:scale(2,2);
```

- 扭曲skew(deg, deg)
  kew(x,y)方法的作用是将元素翻转（扭曲）给定的角度，参数x、y 分别表示围绕x轴翻转给定的角度、围绕y 轴翻转给定的角度

```
transform: skew(deg, deg);
transform: skew(30deg, 30deg); /*围绕x轴翻转30°，围绕y轴翻转30° */
transform: skew(15deg, 65deg); /*围绕x轴翻转15°，围绕y轴翻转65° */
```

- 综合转换matrix(n,n,n,n,n,n)
  matrix()方法是一个综合性的方法，它综合了上述的移动、旋转、缩放等功能。
  matrix()方法有六个参数，包含旋转、缩放、移动（平移）和倾斜功能。语法如下，参数的作用如下

```
transform:matrix(scaleX, skewX, skewY, scaleY, translateX, translateY);/*
   transform:matrix(0.866,0.5,-0.5,0.866,20,20)
```

CSS3 3D 转换

SS3 可以使用3D 转换来对元素进行格式化。常用的3D 转换方法有rotateX()、rotateY()
旋转rotateX()方法。
通过rotateX()方法，元素围绕其X 轴以给定的度数进行旋转。
旋转rotateY()方法。
通过rotateY()方法，元素围绕其Y 轴以给定的度数进行旋转。

```
transform: rotateX(angle); /* X轴方向旋转一定角度 */
transform: rotateY(angle); /* Y轴方向旋转一定角度 */
#div1{transform:rotateX(120deg);}
#div2{transform:rotateY(120deg);margin:10px auto;}
```

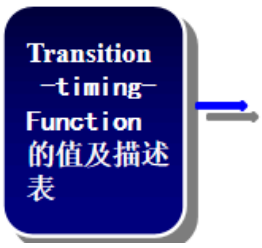
CSS3 过渡transition 属性

transition 属性是一个复合属性，它有四个过渡属性。语法如下

```
    transition: property duration timing-function delay;
    transition: width 2s;   /* 宽度上过渡2s */
    transition-property: none|all| property;
```

```
    transition-property: width;      /* width属性上转场 */
    transition-duration: time;
    transition-duration: 3s;
    transition-timing-function: linear|ease|ease-in|ease-out|ease-in-out|
  transition-timing-function: ease-in-out;
      transition-delay: time;
      transition-delay: 2s;
```

值	描述
transition-property	规定设置过渡效果的CSS属性的名称。
transition-duration	规定完成过渡效果需要多少秒或毫秒。
transition-timing-function	规定速度效果的速度曲线。
transition-delay	定义过渡效果何时开始。



值	描述
linear	规定以相同速度从开始至结束的过渡效果(cubic-bezier(0,0,1,1))。
ease	规定以慢速开始、变快、慢速结束的过渡效果。 类似于cubic-bezier(0.25,0.1,0.25,1)。
ease-in	规定以慢速开始的过渡效果(cubic-bezier(0.42,0,1,1))。
ease-out	规定以慢速结束的过渡效果(cubic-bezier(0,0,0.58,1))。
ease-in-out	规定以慢速开始和结束的过渡效果(cubic-bezier(0.42,0,0.58,1))。
cubic-bezier(n,n,n,n)	在cubic-bezier函数中定义自己的值。可能的值是0～1 之间。

CSS3 动画animation

SS3 动画是指元素从一种样式逐渐变化为另一种样式的效果。通过CSS3的@keyframes（关键帧）规则，可以创建动画，从而取代动画图片、Flash 动画以及JavaScript编写的动画

在@keyframes 中规定某项CSS 样式，就能创建由当前样式逐渐改为新样式的动画效果。

基本语法

```
animation:animation-name|animation-duration|animation-timing-function|anim
```

属性	描述
@keyframes	规定动画。
animation	所有动画属性的复合属性，除了animation-play-state属性。
animation-name	规定@keyframes动画的名称。
animation-duration	规定动画完成一个周期所花费的秒或毫秒，默认是0。
animation-timing-function	规定动画的速度曲线，默认是ease，其它与transition-timing-function属性值相同。
animation-delay	规定动画何时开始，默认是0。
animation-iteration-count	规定动画被播放的次数n(值为1(默认)、infinite)。
animation-direction	规定动画是否在下一周期逆向地播放(值为normal(默认)、alternate)。
animation-play-state	规定动画是否正在运行或暂停，其值为running(默认)、paused。
animation-fill-mode	规定对象动画时间之外的状态（其值为None、forwards、Backwards、both）。

@keyframes 规则定义

采用@keyframes 规则创建动画，需要将它绑定到一个CSS 的选择器，否则动画不会有任何效果。定义至少以下两项CSS3 动画属性，即可将动画绑定到选择器：规定动画的名称、规定动画的时长。

```
     @keyframes myAnimation {
             from {Properties:Properties value; }
             Percentage {Properties:Properties value; }
             to {Properties:Properties value; }
}
     @keyframes myAnimation {
             0% {Properties:Properties value; }
             Percentage {Properties:Properties value; }
            100% {Properties:Properties value; }
}
```

@keyframes 规则的绑定

绑定动画名称（例如myAnimation）到某个元素(div)的样式上，并指定时长。格式如下

```
div{
/* 设置图层基本样式 */
width:100px;height:100px;background:red;position:relative;
/* 设置标准动画子属性 */
      animation: myAnimation 8s;
-moz-animation: myAnimation 8s;       /* Firefox */
-webkit-animation: myAnimation 8s;    /* Safari 和 Chrome */
```

```
-o-animation: myAnimation 8s;           /* Opera */
}
```

CSS3 多列属性

使用CSS3 多列属性可以创建多个列来对文本进行布局，如同编辑报纸和杂志一样。常用的CSS3 多列属性主要有column-count、column-gap、column-rule 等

属性	描述
columns	规定设置column-width和column-count的复合属性。
column-count	规定元素应该被分隔的列数。
column-width	规定列的宽度。
column-fill	规定如何填充列。
column-gap	规定列之间的间隔。
column-rule	设置所有column-rule-*属性的复合属性。
column-rule-width	规定列之间规则的宽度。
column-rule-style	规定列之间规则的样式。
column-rule-color	规定列之间规则的颜色。
column-span	规定元素应该横跨的列数。

```
   columns: column-width column-count; /* 复合属性*/
   column-count: number|auto
   column-width: auto|length;
   column-rule: column-rule-width column-rule-style column-rule-color;
      /* 复合属性*/
   column-rule-width: thin|medium|thick|length;
   column-rule-style: none|hidden|dotted|dashed|solid|double|groove| ridge
inset|outset;
   column-rule-color: color;
   column-gap: length|normal;
   column-fill: balance|auto;   /* balance列长短平衡; auto列顺序填充*/
```

CSS3 文本效果

文本阴影text-shadow 属性

```
    text-shadow: h-shadow v-shadow blur color;  //语法
    text-shadow:2px 2px 8px #FF0000;//示例
```

h-shadow 定义水平阴影，允许负值，必需；
v-shadow 定义垂直阴影，允许负值，必需；

blur 可选。模糊的距离。color 可选。阴影的颜色。省略的长度是0。

文本换行text-wrap 属性

```
text-wrap: normal|none|unrestricted|suppress;
```

文本换行规则。所有浏览器目前均不支持此属性(?)

控制换行word-wrap 属性

```
word-wrap: normal|break-word;
```

ord-wrap 自动换行属性允许强制文本进行换行，会对单词进行拆分。
该属性有两个值：normal、break-word。
其中normal 表示只在允许的断字点换行（浏览器保持默认处理)
break-word 表示在长单词或 URL 地址内部进行换行。

文本溢出text-overflow 属性

```
 text-overflow: clip|ellipsis|string;
```

text-overflow: 属性规定当文本溢出包含元素时发生的事情。
该属性有三个属性值，分别为clip、ellipsis、string。
其中clip 表示修剪文本
ellipsis 表示显示省略符号来代表被修剪的文本
string 表示使用给定的字符串来代表被修剪的文本