HTML5

## 前言

### 认识HTML5

HTML5并不仅仅只是作为HTML标记语言的一个最新版本，更重要的是它制定了Web应用开发的一系列标准，成为第一个将Web做为应用开发平台的HTML语言。

HTML5定义了一系列新元素，如新语义标签、智能表单、多媒体标签等，可以帮助开发者创建富互联网应用，还提供了一系列Javascript API，如地理定位、重力感应、硬件访问等，可以在浏览器内实现类原生应用，甚至结合Canvas我们可开发网页版游戏，同时结合CSS3的过渡、转换、动画等特性，可以极大的增强用户体验，提升开发功能的可应用性。

我们日常讨论的H5其实是一个泛称，它指的是由HTML5 + CSS3 + Javascript等技术组合而成的一个应用开发平台。

### 语法规范

随着Web技术的更新，HTML也先后经历了HTML 4.01、XHTML 1.0、HTML5几个重要的版本，在版本的演变过程中新增或废弃了一些属性，同时对语法规范也做了一些调整，为了能够保证浏览器可以兼容不同版本语法规范的，我们可以使用<!DOCTYPE>指示浏览器应该如何处理我们的HTML。

HTML5在语法规范上也做了比较大的调整，去除了许多冗余的内容，书写规则更加简洁、清晰。

通过以上的比较我们可以总结得出HTML5在语法规范上的特点：

1、**更简洁**(单标签后可不加反斜杠......)

2、**更宽松**(javascript中可不加text=JavaScript.......)

但是我们在实际开发中应规范书写，不建议太随意 ！并且现在基本所有网站都采用了HTML5的语法规范。

**W3C验证地址**

https://validator.w3.org/

## 语义化

所谓语义化是要使HTML标签具备很好的可读性，可以清晰传达每个标签所要表达的意义，以方便其被友好的处理和解析。

### 语义化标签

对于语义化标签我们并不陌生，如<p>表示一个段落、<ul>表示一个无序列表<h1> ~ <h6>表示一系列标题等，在此基础上HTML5增加了大量更有意义的语义标签，更有利于搜索引擎或辅助设备来理解HTML页面内容。

传统的做法我们通过添加类名如class="header"、class="footer"，使HTML页面具有语义性的，但是不具有通用性（如class="header"也可能被写成class="head"）。

HTML5则是通过增加语义化标签的形式来解决这个问题，例如<header></header>、<footer></footer>等，这样就可以保证其具有通用性。

此章节学习目的为了解增加语义标签的目的，以及各语义标签所表达的意义，在网页布局中能够合理使用标签。

1. 常用语义标签

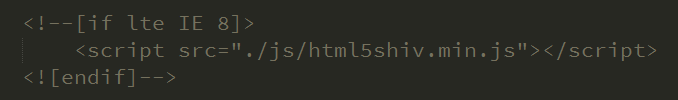
|  |  |
| --- | --- |
| **标签** | **语义** |
| **<nav>** | 导航 |
| **<header>** | 页眉 |
| **<footer>** | 页脚 |
| **<section>(div)** | 区块 |
| **<article>** | 文章 |
| **<aside>** | 侧边栏 |
| **<progress>(min/max/value)** | 进度条 |

本质上新语义标签与<div>、<span>没有区别，只是其具有语义性，使用时除了在HTML结构上需要注意外，其它和普通标签的使用无任何差别，可以理解成<div class="nav"> 相当于 <nav>。**不要好奇，它只是一个标签！**

尽量避免全局使用header、footer、aside等语义标签。

1. 兼容处理

在不支持HTML5新标签的浏览器里，会将这些新的标签解析成行内元素(inline)对待，所以我们只需要将其转换成块元素(block)即可使用，但是在IE9版本以下，并不能正常解析这些新标签，但是却可以识别通过document.createElement('tagName')创建的自定义标签，于是我们的解决方案就是将HTML5的新标签全部通过document.createElement('tagName')来创建一遍，这样IE低版本也能正常解析HTML5新标签了，但在实际开发中我们更多采用的是通过检测IE浏览器的版本来加载第三方的一个JS库来解决兼容问题，这个库文件会帮自动通过document.createElement('tagName')创建所有HTML5的新标签。



**(该注释引用只有IE识别，在chrome中不识别)**

### 微数据

微数据是在如 span、div 的标签内添加属性，让机器（如搜索引擎）识别其含义，某些特定类型的信息，例如评论、人物信息或事件都有相应的属性，用来描述其含义，可以理解成新语义标签的一种补充。

### WAI-ARIA

WAI-ARIA**[1]**, 是Web Accessibility Initiative - Accessible Rich Internet Applications 的缩写，她主要解决的一个问题：让残障人士能无障碍地访问网页上的动态内容。

## 表单

伴随着互联网富应用以及移动开发的兴起，传统的Web表单已经越来越不能满足开发的需求，所以HTML5在Web表单方向也做了很大的改进，如拾色器、日期/时间组件等，使表单处理变的更加高效。**<label>标签的使用form**

### 输入类型（text/password/radio/checkbox/file/button/submit/reset）

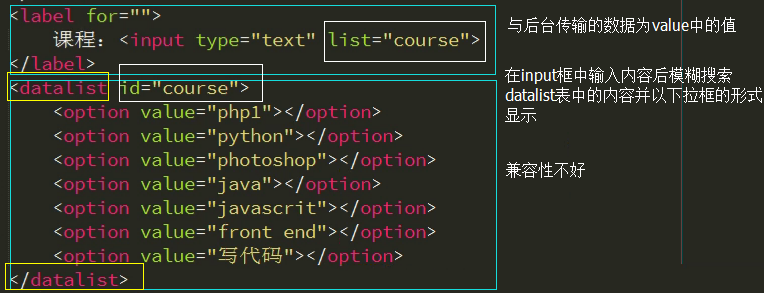
|  |  |  |
| --- | --- | --- |
| **类型** | **使用示例** | **含义** |
| **email** | <input type="email"> | 输入邮箱格式 |
| **tel** | <input type="tel"> | 输入手机号码格式（弹出数字框，不能是小数） |
| **url** | <input type="url"> | 输入url格式 |
| **number** | <input type="number" step=”2” min=”” max=””> | 输入数字格式(可以是小数) |
| **search** | <input type="search"> | 搜索框（体现语义化） |
| **range** | <input type="range" value=”50”> | 自由拖动滑块 |
| **color** | <input type="color"> | 拾色器 |
| **time** | <input type="time"> | 时间 |
| **date** | <input type="date"> | 日期 |
| **datetime** | <input type="datetime"> | 浏览器暂时未支持 |
| **month** | <input type="month"> | 月 |
| **week** | <input type="week"> | 星期 |

部分类型是针对移动设备生效的，且具有一定的兼容性，在实际应用当中应选择性的使用，扫描二维码可以对比测试。



### 表单元素（input/textarea/select....）

|  |  |
| --- | --- |
| **元素** | **含义** |
| **<datalist>=>option(兼容性不好)** | 数据列表（搜索或关联数据） |
| **<keygen>（单标签）** | 生成加密字符串 |
| **<output>（类似于span标签）** | 输出结果（结果：<output>qqq</output>） |
| **<meter min/max/value/low/high></meter>** | 度量器(通过控制Low/high/value值控制度量器颜色) |



### 表单属性

|  |  |  |
| --- | --- | --- |
| **属性** | **用法** | **含义** |
| **placeholder** | <input type="text" placeholder="请输入用户名"> | 占位符 |
| **autofocus** | <input type="text" autofocus> | 自动获得焦点 |
| **multiple** | <input type="file" multiple> | 多文件上传 |
| **autocomplete** | <input type="text" autocomplete="off /on">(为on时再次输入时下拉框自动展示之前的输入内容) | 自动完成（表单中默认全部打开，设置在form中时全部关闭） |
| **form** | <input type="text" form="某表单ID"> | 关联到某表单，数据跟随该表单一起提交 |
| **novalidate** | <form novalidate></form> | 关闭验证条件  （email/num.） |
| **required** | <input type="text" required> | 必填项 |
| **pattern** | <input type="text" pattern="\d">(=正则表达式) | 自定义验证 |

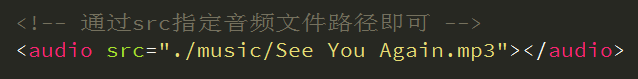
## 多媒体

在HTML5之前，在网页上播放音频/视频的通用方法是利用Flash来播放，但是大多情况下，并非所有用户的浏览器都安装了Flash插件，由此使得处理音频/视频播放变的非常复杂，并且移动设备的浏览器并不支持Flash插件。

### 音频

HTML5通过<audio>标签来解决音频播放的问题。

使用相当简单，如下图所示



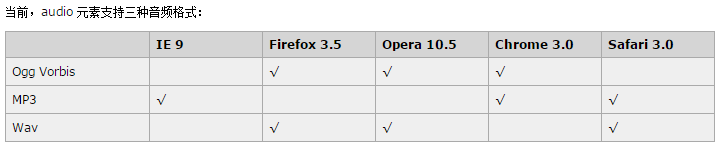
并且可以通过附加属性可以更友好控制音频的播放，如：

autoplay 自动播放

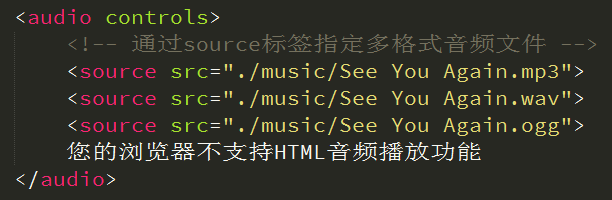
controls 是否显示默认播放控件

loop 循环播放

由于版权等原因，不同的浏览器可支持播放的格式是不一样的，如下图供参考



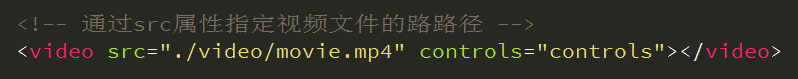
**多浏览器支持的方案，如下图**



### 视频

HTML5通过<video>标签来解决音频播放的问题。

同音频播放一样，<video>使用也相当简单，如下图



同样，通过附加属性可以更友好的控制视频的播放

autoplay 自动播放

controls 是否显示默认播放控件

loop 循环播放

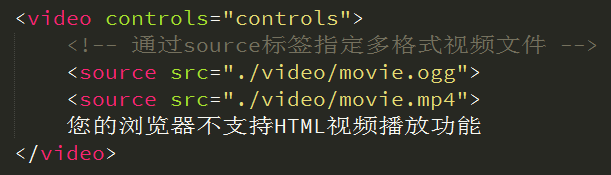
width 设置播放窗口宽度（宽高可只设置一边，另一边自动调整）

height 设置播放窗口的高度

由于版权等原因，不同的浏览器可支持播放的格式是不一样的，如下图供参考



**多浏览器支持的方案，如下图**



## DOM扩展

### 获取元素（属于H5新提供，有兼容性问题）

1、document.getElementsByClassName ('class') 通过类名获取元素，以伪数组形式存在。

2、document.querySelector('selector') 通过CSS选择器获取元素，符合匹配条件的第1个元素。

3、document.querySelectorAll('selector') 通过CSS选择器获取元素，以伪数组形式存在。

### 类名操作（与jquery功能一样，没引用jquery的高级浏览器的情况下使用）

1、Node.classList.add('class') 添加class

2、Node.classList.remove('class') 移除class

3、Node.classList.toggle('class') 切换class，有则移除，无则添加

4、Node.classList.contains('class') 检测是否存在class

Node指一个有效的DOM节点，是一个通称。

### 自定义属性

在HTML5中我们可以自定义属性，其格式如下data-\*=""，例如

data-info="我是自定义属性"，通过Node.dataset['info'] 我们便可以获取到自定义的属性值。

Node.dataset是以对象形式存在的，当我们为同一个DOM节点指定了多个自定义属性时，Node.dataset则存储了所有的自定义属性的值。

假设某元素 <div id="demo" data-name="itcast" data-age="10">

var demo = document.querySelector('#demo');

1、读取 demo.dataset['name'] 或者 demo.dataset['age']

2、设置demo.dataset['name'] = 'web developer'

注：当我们如下格式设置时，则需要以驼峰格式才能正确获取

<div data-my-name="itcast"> 这样获取Node.dataset['myName']

## 新增API

### 网络状态

我们可以通过window.onLine来检测，用户当前的网络状况，返回一个布尔值。

window.online用户网络连接时被调用

window.offline用户网络断开时被调用



### 全屏

HTML5规范允许用户自定义网页上任一元素全屏显示。

1、Node.requestFullScreen() 开启全屏显示

2、Node.cancelFullScreen() 关闭全屏显示

由于其兼容性原因，不同浏览器需要添加前缀如：

webkit内核浏览器：webkitRequstFullScreen、webkitCancelFullScreen，如chrome浏览器。

Gecko内核浏览器：mozRequestFullScreen、mozCancelFullScreen，如火狐浏览器。

3、document.fullScreen检测当前是否处于全屏

不同浏览器需要添加前缀

document.webkitIsFullScreen、document.mozFullScreen

全屏伪类选择器

:full-screen .box {}、:-webkit-full-screen {}、:moz-full-screen {}



### 文件读取

通过FileReader对象我们可以读取本地存储的文件，使用 [File](https://developer.mozilla.org/zh-CN/docs/DOM/File" \t "_blank" \o "此页面仍未被本地化, 期待您的翻译!)对象来指定所要读取的文件或数据。其中File对象可以是来自用户在一个 [<input>](https://developer.mozilla.org/zh-CN/docs/HTML/Element/input" \t "_blank" \o "This article hasn't been written yet. Please consider contributing!)元素上选择文件后返回的[FileList](https://developer.mozilla.org/zh-CN/docs/DOM/FileList" \t "_blank" \o "一个FileList对象通常来自于一个HTML input元素的files属性,你可以通过这个对象访问到用户所选择的文件.该类型的对象还有可能来自用户的拖放操作,查看DataTransfer对象了解详情.)对象，也可以来自由拖放操作生成的  [DataTransfer](https://developer.mozilla.org/zh-CN/DragDrop/DataTransfer" \t "_blank" \o "En/DragDrop/DataTransfer)。

#### FileList对象

由于HTML5中我们可以通过为表单元素添加multiple属性，因此我们通过<input>上传文件后得到的是一个FileList对象（伪数组形式）。

#### FileReader对象

HTML5新增内建对象，可以读取本地文件内容。

var reader = new FileReader; 可以实例化一个对象

**实例方法**

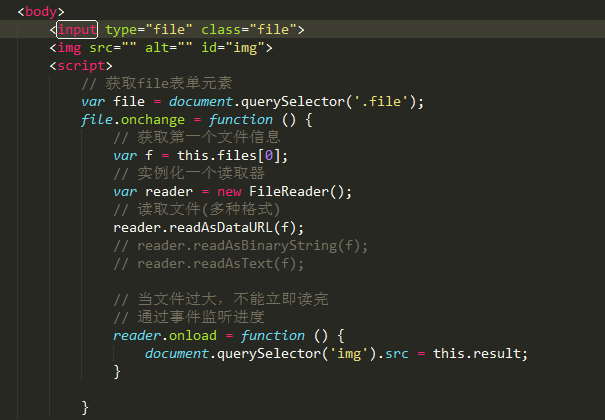
1、[readAsDataURL](https://developer.mozilla.org/zh-CN/docs/Web/API/FileReader" \l "readAsDataURL())() 以DataURL形式读取文件

**事件监听**

onload 当文读取完成时调用

**属性**

result 文件读取结果



### 拖拽

在HTML5的规范中，我们可以通过为元素增加draggable="true"来设置此元素是否可以进行拖拽操作，其中图片、链接默认是开启的。

#### 拖拽元素

页面中设置了**draggable="true"**属性的元素，其中<img>、<a>标签默认是可以被拖拽的

#### 目标元素

页面中任何一个元素都可以成为目标元素，不需要特殊指定。

#### 事件监听

根据元素类型不同，需要设置不同的事件监听

**1、拖拽元素**

ondrag 应用于拖拽元素，整个拖拽过程都会调用

ondragstart 应用于拖拽元素，当拖拽开始时调用

ondragleave 应用于拖拽元素，当鼠标离开拖拽元素时调用

ondragend 应用于拖拽元素，当拖拽结束时调用

**2、目标元素**

ondragenter 应用于目标元素，当拖拽元素进入时调用

ondragover 应用于目标元素，当停留在目标元素上时调用

ondrop 应用于目标元素，当在目标元素上松开鼠标时调用

ondragleave 应用于目标元素，当鼠标离开目标元素时调用



### 地理定位

在HTML规范中，增加了获取用户地理信息的API，这样使得我们可以基于用户位置开发互联网应用，即基于位置服务 (Location Base Service)

#### 获取地理信息

通过IP地址、三维坐标、GPS、Wi-Fi、手机信息等多种方式获取地址信息，如下图对不同获取方式的优缺点进行了比较，浏览器会自动以最优方式去获取用户地理信息。



#### 隐私

HTML5 **Geolocation** 规范提供了一套保护用户隐私的机制。必须先得到用户明确许可，才能获取用户的位置信息。

#### API详解

1、获取当前地理信息

navigator. geolocation.getCurrentPosition(successCallback, errorCallback, options)

2、重复获取当前地理信息

navigator. geolocation.watchPosition(successCallback, errorCallback, options)

当成功获取地理信息后，会调用succssCallback，并返回一个包含位置信息的对象position。

position.coords.latitude纬度

position.coords.longitude经度

position.coords.accuracy精度

position.coords.altitude海拔高度

当获取地理信息失败后，会调用errorCallback，并返回错误信息error

可选参数 options 对象可以调整位置信息数据收集方式(一般默认)

a) enableHighAccuracy 高精度模式

b) timeout 超时设置，单位为ms

c) maximumAge表示浏览器重新获取位置信息的时间间隔，单位为ms

#### 应用

在现实开发中，通过调用第三方API（如百度地图API）来实现地理定位信息，这些API都是基于用户当前位置的，并将用位置位置（经/纬度）当做参数传递，就可以实现相应的功能。**http://lbsyun.baidu.com/index.php?title=jspopular(根据文档使用（密钥）)**



### 历史管理（单页面应用）

提供window.history，对象我们可以管理历史记录，可用于**单页面应用**，Single Page Application，可以无刷新改变网页内容。

#### 新增方法

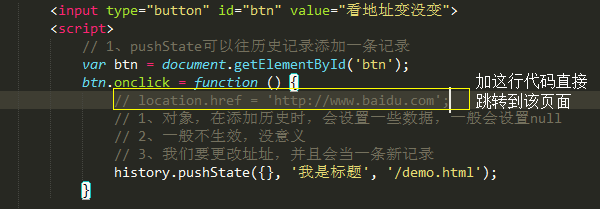
1、pushState(data, title, url) 追加一条历史记录

data用于存储自定义数据，通常设为null

title网页标题，基本上没有被支持，一般设为空

url 以当前域为基础增加一条历史记录，不可跨域设置

2、replaceState(data, title, url) 与pushState()基本相同，不同之处在于replaceState()，只是替换当前url，不会增加/减少历史记录。



#### 事件监听

onpopstate事件，当前进或后退时则触发，通过事件对象ev.state可以读取到存储的数据。



### Web存储

随着互联网的快速发展，基于网页的应用越来越普遍，同时也变的越来越复杂，为了满足各种各样的需求，会经常性在本地存储大量的数据，传统方式我们以document.cookie来进行存储的，但是由于其存储大小只有4k左右，并且解析也相当的复杂，给开发带来诸多不便，HTML5规范则提出解决方案。

#### 特性

1、设置、读取方便

2、容量较大，**sessionStorage**约5M、**localStorage**约20M

4、只能存储字符串，可以将对象**JSON.stringify()** 编码后存储

#### window.sessionStorage

**1、生命周期为关闭浏览器窗口**

**2、在同一个窗口下数据可以共享**

#### window.localStorage

**1、永久生效，除非手动删除**

**2、可以多窗口共享**

#### 方法详解

setItem(key, value) 设置存储内容

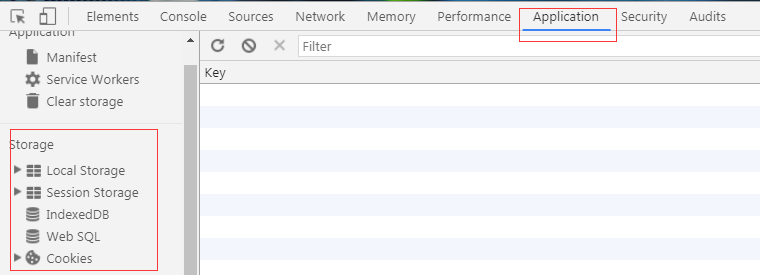
getItem(key) 读取存储内容

removeItem(key) 删除键值为key的存储内容

clear() 清空所有存储内容

key(n) 以索引值来获取存储内容





#### 其它

WebSQL、IndexDB

### 应用缓存

HTML5中我们可以轻松的构建一个离线（无网络状态）应用，只需要创建一个cache manifest文件。

#### 优势

1、可配置需要缓存的资源

2、网络无连接应用仍可用

3、本地读取缓存资源，提升访问速度，增强用户体验

4、减少请求，缓解服务器负担

#### 缓存清单

一个普通文本文件，其中列出了浏览器应缓存以供离线访问的资源，推荐使用.appcache为后缀名，添加MIME类型

AddType text/cache-manifest .appcache

例如我们创建了一个名为demo.appcache的文件，然后在需要应用缓存在页面的根元素(html)添加属性manifest="demo.appcache"，路径要保证正确。

#### manifest文件格式

1、顶行写CACHE MANIFEST

2、CACHE: 换行 指定我们需要缓存的静态资源，如.css、image、js等

3、NETWORK: 换行 指定需要在线访问的资源，可使用通配符

4、FALLBACK: 换行 当被缓存的文件找不到时的备用资源

#### 事件监听

可自行查阅资料

#### 其它

1、CACHE: 可以省略，这种情况下将需要缓存的资源写在CACHE MANIFEST

2、可以指定多个CACHE: NETWORK: FALLBACK:，无顺序限制

3、#表示注释，只有当demo.appcache文件内容发生改变时或者手动清除缓存后，才会重新缓存。

4、chrome 可以通过chrome://appcache-internals/工具和离线（offline）模式来调试管理应用缓存

### 多媒体

方法：load()、play()、pause()

属性：currentSrc、currentTime、duration

事件：ontimeupdate，onended等

### 应用

1. 自定义视频播放器

CSS3

## CSS3简介

如同人类的的进化一样，CSS3是CSS2的“进化”版本，在CSS2基础上，增强或新增了许多特性， 弥补了CSS2的众多不足之处，使得Web开发变得更为高效和便捷。

### CSS3的现状

1、浏览器支持程度差，需要添加私有前缀

2、移动端支持优于PC端

3、不断改进中

4、应用相对广泛

### 如何对待

1、坚持渐进增强原则

2、考虑用户群体

3、遵照产品的方案

4、听Boss的

## 准备工作

### 统一环境

由于CSS3兼容性问题的普遍存在，为了避免因兼容性带来的干扰，我们约定统一的环境，以保证学习的效率，在最后会单独说明兼容性的问题。

1、Chrome浏览器 version 46+

2、Firefox浏览器 firefox 42+

3、PhotoShop CS6（建议）

### 如何使用手册

学会使用工具，可以让我们事半功倍。

[] 表示全部可选项

|| 表示或者

| 表示多选一

？ 表示0个或者1个

\* 表示0个或者多个

{} 表示范围

学会查看手册，培养自主学习能力。

## 基础知识

### 选择器

CSS3新增了许多灵活查找元素的方法，极大的提高了查找元素的效率和精准度。CSS3选择器与jQuery中所提供的绝大部分选择器兼容。

#### 属性选择器

其特点是通过属性来选择元素，具体有以下5种形式：

1、E[attr] 表示存在attr属性即可；

2、E[attr=val] 表示属性值完全等于val；

3、E[attr\*=val] 表示的属性值里包含val字符并且在“任意”位置；

4、E[attr^=val] 表示的属性值里包含val字符并且在“开始”位置；

5、E[attr$=val] 表示的属性值里包含val字符并且在“结束”位置；

#### 伪类选择器(冒号：)

除了以前学过的:link、:active、:visited、:hover，CSS3又新增了其它的伪类选择器。

1、以某元素相对于其父元素或兄弟元素的位置来获取无素的结构伪类。

重点理解**通过E来确定元素的父元素从而选择对应的该子元素**。

E:first-child第一个子元素

E:last-child最后一个子元素

E:nth-child(n) 第n个子元素，计算方法是E元素的全部兄弟元素；

E:nth-last-child(n) 同E:nth-child(n) 相似，只是倒着计算；

n为具体数值或者变量，当为变量时从0开始。0.1.2.3.4.5........

n遵循线性变化，其取值0、1、2、3、4、... 但是当n<=0时，选取无效。

n可是多种形式：nth-child(2n+0)、nth-child(even)、nth-child(2n+1)、nth-child(odd)、nth-child(-1n+3)(选择的为前面3（n）个)、nth-last-child(-1n+3)(选择的为后面3（n）个)；

需要满足y=an+b

注：指E元素的父元素，并对应位置的子元素必须是E

E:not(.special) /\*除了.special以外的元素被选中\*/

E:empty 选中没有任何子节点的E元素；（使用不是非常广泛）

/\*当元内容为空会被选中\*/

/\*在某些特定场合灵活使用会带来很好的效果\*/

2、目标伪类

E:target 结合锚点进行使用，处于当前锚点的元素会被选中；

锚点：节点的内容的某个区域可以是链接的目标或源。

#### 伪元素选择器(双冒号：：)

E::first-letter文本的第一个单词或字（如中文、日文、韩文等）；

E::first-line 文本第一行；

E::selection 可改变选中文本的样式；（不能改变字号）

重点：E::before、E::after

li::before {

content: 'itcast';

/\*display: block;\*/

width: 100px;

height: 100px;

background-color: blue;

}

是一个行内元素，需要转换成块元素

/\*::before ::after 会分别在元素内部的最前面和最后面添加一个元素\*/

/\*并且我们可像定义一个正常元素一样去定义它的样式\*/

/\*必须要有content ，content 可以写入文本内容，通常给空\*/

/\*新创建的元素是一个行内元素，在使用时通常会转成块元素，但并不是必须要转\*/

E:after、E:before 在旧版本里是伪元素，CSS3的规范里“:”用来表示伪类，“::”用来表示伪元素，但是在高版本浏览器下E:after、E:before会被自动识别为E::after、E::before，这样做的目的是用来做兼容处理。

E:after、E:before后面的练习中会反复用到，目前只需要有个大致了解

":" 与 "::" 区别在于区分伪类和伪元素

参考文档

:before和::before的区别

https://www.qianduan.net/before-and-before-the-difference-between/

### 颜色

新增了RGBA、HSLA模式，其中的A 表示透明度通道，即可以设置颜色值的透明度，相较opacity，它们不具有继承性，即不会影响子元素的透明度。

Red、Green、Blue、Alpha即RGBA

Hue、Saturation、Lightness、Alpha即HSLA

R、G、B 取值范围0~255

H 色调 取值范围0~360，0/360表示红色、120表示绿色、240表示蓝色

S 饱和度 取值范围0%~100%

L 亮度 取值范围0%~100%

A 透明度 取值范围0~1

关于透明度：

1、opacity只能针对整个盒子设置透明度，子盒子及内容会继承父盒子的透明度；

2 、transparent 不可调节透明度，始终完全透明(可应用于三角形制作)

border:20px solid transparent;

三角形

width:0 height:0 border-width:20px border-style:solid border-color:red transparent

transparent transparent transparent

RGBA、HSLA可应用于所有使用颜色的地方。

### 文本

text-shadow，可分别设置偏移量、模糊度、颜色（可设透明度）。

text-shadow: 5px 5px 5px red, -5px -5px 5px blue, -15px -15px 5px pink;

1、水平偏移量 正值向右 负值向左；

2、垂直偏移量 正值向下 负值向上；

3、模糊度是不能为负值；

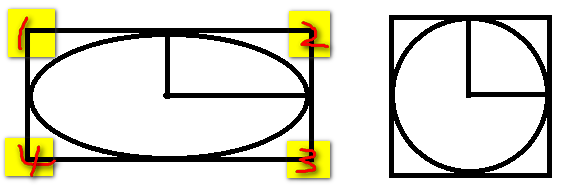
### 边框

其中边框圆角、边框阴影属性，应用十分广泛，兼容性也相对较好，具有符合渐进增强原则的特征，我们需要重点掌握。

#### 边框圆角

border-radius

圆角处理时，脑中要形成圆、圆心、横轴、纵轴的概念，正圆是椭圆的一种特殊情况。如下图



可分别设置长、短半径，以“/”进行分隔，遵循“1，2，3，4”规则，“/”前面的1~4个用来设置横轴半径（分别对应横轴1、2、3、4位置 ），“/”后面1~4个参数用来设置纵轴半径（分别对应纵轴1、2、3、4位置 ）

border-radius:10px 20px 30px 40px / 10px 20px 30px 40px(横轴半径/竖轴半径)

椭圆圆角，当横轴与竖轴相等时为圆。（完整版）

#### 边框阴影

box-shadow

1、水平偏移量 正值向右 负值向左；

2、垂直偏移量 正值向下 负值向上；

3、模糊度是不能为负值；

4、inset可以设置内阴影；

设置边框阴影不会改变盒子的大小，即不会影响其兄弟元素的布局。

可以设置多重边框阴影，实现更好的效果，增强立体感。

符合渐进增强，实际开发中可以大胆使用

box-shadow: 2px 2px 10px #CCC, 4px 4px 4px blue, inset 8px 8px 10px red;

box-shadow：第一层阴影，第二层，内阴影；

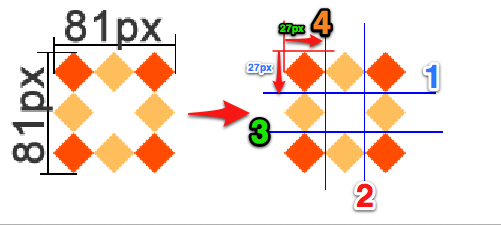
#### 边框图片

border-image: url(./images/border.png) 27 stretch repeat;（总写）

引入-----border-image-source: url(./images/border.png);

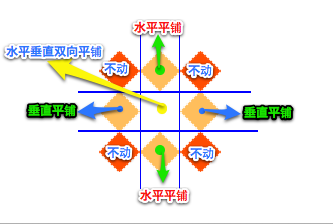
裁剪-----border-image-slice: 34 36 27 27;  
 平铺-----border-image-repeat: round; round/repeat/stretch（分x方向与y方向）

设置的图片将会被“切割”成九宫格形式，然后进行设置。如下图



“切割”完成后生成虚拟的9块图形，然后按对应位置设置背景，

其中四个角位置、形状保持不变，中心位置水平垂直两个方向平铺。如下图



**1、round和repeat之间的区别**

round 会自动调整尺寸，完整显示边框图片。

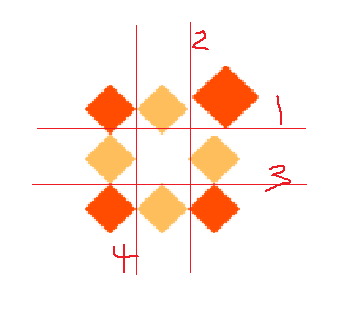


repeat 单纯平铺多余部分，会被“裁切”而不能完整显示。



**2、更改裁切尺寸**

: 34 36 27 27 分别设置裁切如下图





关于边框图片重点理解9宫格的裁切及平铺方式，实际开发中应用不广泛，但是如能灵活动用会给我们带来不少便利。

### 盒模型

CSS3中可以通过box-sizing 来指定盒模型，即可指定为content-box、border-box，这样我们计算盒子大小的方式就发生了改变。

可以分成两种情况：

1、box-sizing: border-box 计算方式为content = width – border - padding

2、box-sizing: content-box 计算方式为content = width（默认）

注：上面的标注的**width指的是CSS属性里设置的width: length，content的值是会自动调整的。**

兼容性比较好

### 背景

背景在CSS3中也得到很大程度的增强，比如背景图片尺寸、背景裁切区域、背景定位参照点、多重背景等。

1. **background-size设置背景图片的尺寸**(等比例缩放)

a) 可以设置长度单位或百分比

b) **cover**会自动调整缩放比例，保证图片始终填充满背景区域，如有溢出部分则会被隐藏。

c ) contain会自动调整缩放比例，保证图片始终完整显示在背景区域。

也可以使用长度单位或百分比

background-size: 200px 100px;（直接设置背景图片大小 ）

**2、background-origin设置背景定位的原点**

border-box以边框做为参考原点；

padding-box以内边距做为参考原点（默认）**（背景图片定位以paddin左上角为参考原点）；**

content-box以内容区做为参考点；

**3、background-clip设置背景区域裁切**

border-box裁切边框以内为背景区域（默认）**（背景颜色定位以border左上角为参考原点）；**

padding-box裁切内边距以内为背景区域；

content-box裁切内容区做为背景区域；

**4、以逗号分隔可以设置多背景，可用于自适应局**

背景图片尺寸在实际开发中应用十分广泛。（上面一个背景图片，下面一个背景图片，中间的内容可按照内容自动调整高度）

### 渐变

渐变是CSS3当中比较丰富多彩的一个特性，通过渐变我们可以实现许多炫丽的效果，有效的减少图片的使用数量，并且具有很强的适应性和可扩展性。

可分为线性渐变、径向渐变

#### 线性渐变

linear-gradient线性渐变指沿着某条直线朝一个方向产生渐变效果。



上图是从黄色渐变到绿色

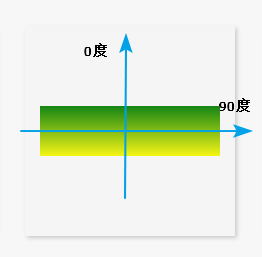
**1、必要的元素：**

a、方向

b、起始颜色

c、终止色；

**2、关于方向如下图**



background: linear-gradient(to right, yellow, green);

background: linear-gradient(0, yellow, green);

background: linear-gradient(90deg, yellow, green);

百分比相对于整个盒子 指的是在整个盒子的位置如50%为盒子1/2处

background: linear-gradient(90deg, yellow 25%, green 50%);

将渐变当成一个图片来对待

background-image: linear-gradient(

90deg, yellow 25%, green 25%, green 50%, pink 50%, pink 75%, blue 75%, blue 100%);

#### 径向渐变

radial-gradient径向渐变指从一个中心点开始沿着四周产生渐变效果



**1、必要的元素：**

a、辐射范围即圆半径

b、中心点 即圆的中心

c、渐变起始色

d、渐变终止色

**2、关于中心点：**中心位置参照的是盒子的左上角

**3、关于辐射范围：**其半径可以不等，即可以是椭圆

background-image: radial-gradient(150px at center, yellow, green);

中心点的位置是以盒子自身为参照

background-image: radial-gradient(300px at left center, yellow, green);

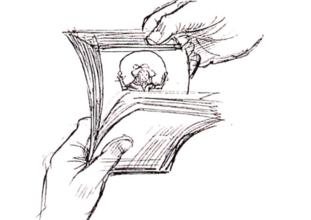
盒子左上角为圆心点，我们可通过坐标来设置中心位置

background-image: radial-gradient(150px at 80px 80px, yellow, green);

### 过渡 ( transition: all 1s;)

过渡是CSS3中具有颠覆性的特征之一，可以实现元素不同状态间的平滑过渡（补间动画），经常用来制作动画效果。

帧动画：通过一帧一帧的画面按照固定顺序和速度播放。如电影胶片



补间动画：自动完成从起始状态到终止状态的的过渡。

关于补间动画更多学习可查看http://mux.alimama.com/posts/1009

特点：当前元素只要有“属性”发生变化时，可以平滑的进行过渡，并不仅仅局限于hover状态。

transition-property设置过渡属性

transition-duration设置过渡时间

transition-timing-function设置过渡速度

transition-delay设置过渡延时

transition: all 1s;

设置需要过渡的属性

transition-property: background-color, left;(一般设置ALL)

设置过渡时间

transition-duration: 2s;

设置过渡的延时

transition-delay: 2s;

过渡执行的速度

transition-timing-function: ease-in-out;

第一个时间是执行时间，第二个是延时间

参数间除时间外没有顺序要求

transition: linear all 2s 1s;

transition: 属性 时间 延时 速度

以上四属性重在理解

### 2D转换(**transform: translate() rotate() scale() ...等**)

转换是CSS3中具有颠覆性的特征之一，可以实现元素的位移、旋转、变形、缩放，甚至支持矩阵方式，配合即将学习的过渡和动画知识，可以取代大量之前只能靠Flash才可以实现的效果。

1. 移动 translate(x, y) 可以改变元素的位置，x、y可为负值；

3D移动：translateX（x）；translateY（y）；translateZ（z）；

a) 移动位置相当于自身原来位置

b) y轴正方向朝下

c) 除了可以像素值，也可以是百分比，相对于自身的宽度或高度(可应用于**设置元素的水平垂直居中（top:50% left:50% transform:translate(-50%,-50%)**）

2、缩放 scale(x, y) 可以对元素进行水平和垂直方向的缩放，x、y的取值可为小数；

3、旋转 rotate(deg) 可以对元素进行旋转，正值为顺时针，负值为逆时针；

a) 当元素旋转以后，坐标轴也跟着发生的转变

b) 调整顺序可以解决，把旋转放到最后

3D旋转：rotateX（deg）；rotateY（deg）；rotateZ（deg）；

4、倾斜 skew(deg, deg) 可以使元素按一定的角度进行倾斜，可为负值，第二个参数不写默认为0。

5、矩阵matrix() 把所有的2D转换组合到一起，需要6个参数。

transform-origin可以调整元素转换的原点，但是对于transform: translate(x，y) 没有影响。

**我们可以同时使用多个转换，其格式为：transform: translate() rotate() scale() ...等，其顺序会影转换的效果。**

### 3D转换

**1、3D坐标轴（CSS）**

用X、Y、Z分别表示空间的3个维度，三条轴互相垂直。如下图



**2、透视（perspective）**

电脑显示屏是一个2D平面，图像之所以具有立体感（3D效果），其实只是一种视觉呈现 ，通过透视可以实现此目的。

透视可以将一个2D平面，在转换的过程当中，呈现3D效果。（没有perspective，便“没有”Z轴）

**并非任何情况下需要透视效果**

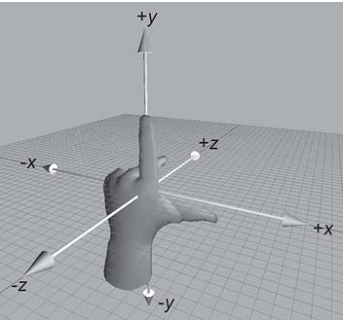
**3、perspective有两种写法**

a) 作为一个属性，设置给父元素，作用于所有3D转换的子元素

b) 作为transform属性的一个值，做用于元素自身 ，并且必须要写在最前面

**4、左手坐标系（通常）**

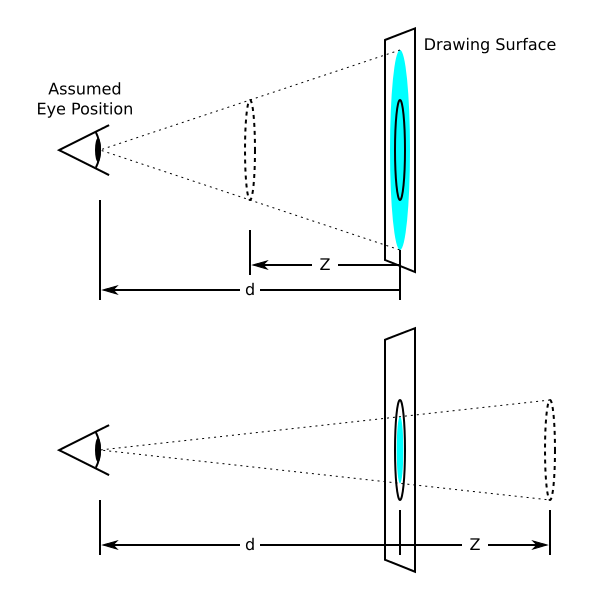
伸出左手，让拇指和食指成“L”形，大拇指向右，食指向上，中指指向前方。这样我们就建立了一个左手坐标系，拇指、食指和中指分别代表X、Y、Z轴的正方向。如下图



**5、左手法则**

左手握住旋转轴，竖起拇指指向旋转轴正方向，正向就是其余手指卷曲的方向。

**6、理解透视距离**



透视会产生“近大远小”的效果

**7、3D呈现（transform-style）（**transform-style:preserve-3d**）**

设置内嵌的元素在 3D 空间如何呈现，这些子元素必须为变形原素。

flat：所有子元素在 2D 平面呈现

preserve-3d：保留3D空间

所有的子元素必须要应用3D转换(设置transform)才可以

**8、backface-visibility(backface-visibility:hidden)**

设置元素背面是否可见

**参考文档**

http://isux.tencent.com/css3/index.html?transform

**CSS3动画库**

animate.css

### 动画

动画是CSS3中具有颠覆性的特征之一，可通过设置多个节点来精确控制一个或一组动画，常用来实现复杂的动画效果。

.box {

width: 200px;

height: 200px;

margin: 0 auto;

background-color: green;

animation: change 10s linear forwards;

}

/\*动画序列\*/

@keyframes change {

/\*百分比是相对于动画的执行时间\*/

0% {

}

20% {

width: 400px;

height: 200px;

}

40% {

width: 400px;

height: 400px;

}

80% {

width: 400px;

height: 400px;

background-color: yellow;

}

100% {

width: 200px;

height: 200px;

background-color: green;

}

}

**1、必要元素：**

a、通过@keyframes指定动画序列；

b、通过百分比将动画序列分割成多个节点；

c、在各节点中分别定义各属性

d、通过animation将动画应用于相应元素；

**2、关键属性（animation: 10s linear forwards）**

a、**animation-name**设置动画序列名称

b、**animation-duration**动画持续时间

c、**animation-delay**动画延时时间

d、**animation-timing-function**动画执行速度，linear（匀速）、ease（渐近）、ease-in（由慢到快）、ease-out（由快到慢） 、ease-in-out（由慢到快再到慢）等

e、**animation-play-state**动画播放状态，running（运动）、paused（暂停）

f、**animation-direction**动画逆播，normal（正常方向 ）、reverse（反方向运行）、alternate（动画先正常运行再反方向运行，并持续交替运行 ）、alternate-reverse（动画先反运行再正方向运行，并持续交替运行 ）

g、**animation-fill-mode**动画执行完毕后状态，none（默认值。不设置对象动画之外的状态） forwards（设置对象状态为动画结束时的状态） backwards（设置对象状态为动画开始时的状态） both（设置对象状态为动画结束或开始的状态）

h、**animation-iteration-count**动画执行次数，infinite（无限循环 ）[<number>](mk:@MSITStore:F:\\前端与移动开发\\CSS3参考手册.chm::/properties/animation/../../values/numeric/number.htm)：

指定对象动画的具体循环次数

i、**steps(60)** 表示动画分成60步完成

**参数值的顺序：**

关于几个值，除了**名字，动画时间，延时**有严格顺序要求其它随意

### 伸缩布局

CSS3在布局方面做了非常大的改进，使得我们对**块级元素**的布局排列变得十分灵活，适应性非常强，其强大的伸缩性，在响应式开中可以发挥极大的作用。

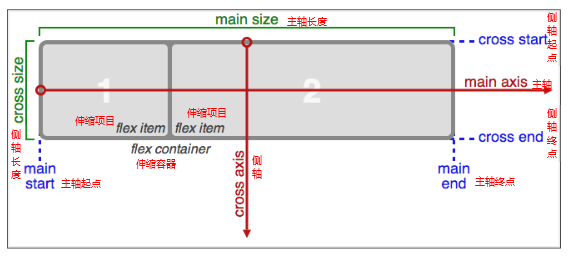
如下图，学习新的概念：

主轴：Flex容器的主轴主要用来配置Flex项目，默认是水平方向

侧轴：与主轴垂直的轴称作侧轴，默认是垂直方向的

方向：默认主轴从左向右，侧轴默认从上到下

主轴和侧轴并不是固定不变的，通过flex-direction可以互换。



**1、必要元素：**

**a、指定一个盒子为伸缩盒子 display: flex**

**b、设置属性来调整此盒的子元素的布局方式 例如 flex-direction**

**c、明确主侧轴及方向**

**d、可互换主侧轴，也可改变方向**

1. **各属性详解**

**设置在父元素上**

1. **flex-direction调整主轴方向（默认为水平方向）**

row:水平排列 column:垂直排列 row-reverse:水平反向排列 column-reverse:垂直反向排列

**b、justify-content调整主轴对齐**

flex-start：起始点对齐、flex-end：终止点对齐、center：居中对齐、 space-around：四周环绕、space-between：两端对齐

**c、align-items调整侧轴对齐**

flex-start：侧轴起始点对齐、flex-end：侧轴终止点对齐、

center：侧轴居中对齐、stretch：拉伸

**d、flex-wrap控制是否换行**

wrap：自动换行、nowrap：不换行

**e、align-content堆栈排列（堆栈对齐），可对应用flex-wrap: wrap后产生 的换行进行控制**

flex-start：起始点对齐、flex-end：终止点对齐、center：居中对齐、space-between：两端对齐、space-around：四周环绕、stretch：拉伸

**f、flex-flow是flex-direction、flex-wrap的简写形式**

**设置在子元素上**

**g、flex子项目在主轴的缩放比例，不指定flex属性，则不参与伸缩分配**

(每一个子元素都设置一个flex：num，分布每一个元素在剩余空间所占的 相应比例)

**h、align-self同algin-item可覆盖父元素设置的algin-item**

flex-start：侧轴起始点对齐、flex-end：侧轴终止点对齐、

center：侧轴居中对齐、stretch：拉伸

**i、order控制子项目的排列顺序，正序方式排序，从小到大**

此知识点重在理解，要明确找出主轴、侧轴、方向，各属性对应的属性值可参考示例源码

### 多列布局

类似报纸或杂志中的排版方式，上要用以控制大篇幅文本。

了解即可，实际意义不大。

/\*设置列数\*/ -webkit-column-count: 4;

/\*设置第列的宽度\*/ -webkit-column-width: 400px;

/\*设置列间距\*/ -webkit-column-gap: 40px;

/\*设置分隔线\*/ -webkit-column-rule: 1px dashed #CCC;

## Web字体

开发人员可以为自已的网页指定特殊的字体，无需考虑用户电脑上是否安装了此特殊字体，从此把特殊字体处理成图片的时代便成为了过去。

支持程度比较好，甚至IE低版本浏览器也能支持。

### 字体格式

不同浏览器所支持的字体格式是不一样的，我们有必要了解一下有关字体格式的知识。

1、TureType(.ttf)格式

.ttf字体是Windows和Mac的最常见的字体，是一种RAW格式，支持这种字体的浏览器有IE9+、Firefox3.5+、Chrome4+、Safari3+、Opera10+、iOS Mobile、Safari4.2+；

2、OpenType(.otf)格式

.otf字体被认为是一种原始的字体格式，其内置在TureType的基础上，支持这种字体的浏览器有Firefox3.5+、Chrome4.0+、Safari3.1+、Opera10.0+、iOS Mobile、Safari4.2+；

3、Web Open Font Format(.woff)格式

woff字体是Web字体中最佳格式，他是一个开放的TrueType/OpenType的压缩版本，同时也支持元数据包的分离，支持这种字体的浏览器有IE9+、Firefox3.5+、Chrome6+、Safari3.6+、Opera11.1+；

4、Embedded Open Type(.eot)格式

.eot字体是IE专用字体，可以从TrueType创建此格式字体，支持这种字体的浏览器有IE4+；

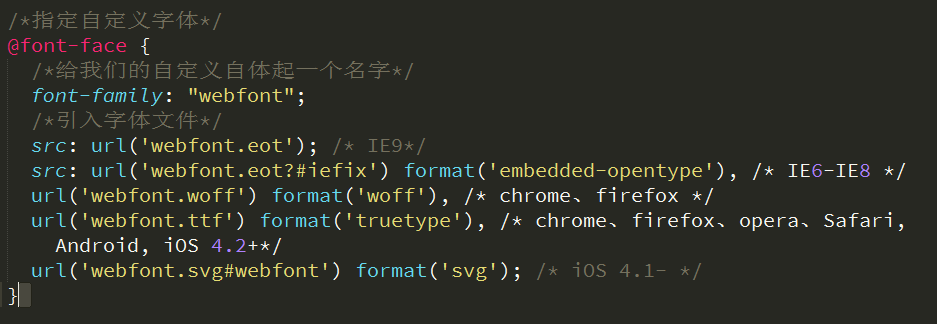
5、SVG(.svg)格式

.svg字体是基于SVG字体渲染的一种格式，支持这种字体的浏览器有Chrome4+、Safari3.1+、Opera10.0+、iOS Mobile Safari3.2+；

了解了上面的知识后，我们就需要为不同的浏览器准备不同格式的字体，通常我们会通过字体生成工具帮我们生成各种格式的字体，因此无需过于在意字体格式间的区别差异。

推荐http://www.zhaozi.cn/、http://www.youziku.com/ 查找更多中文字体

### 字体图标



其实我们可以把文字理解成是一种特殊形状的图片，反之我们是不是也可以把图片制作成字体呢？

答案是肯定的。

常见的是把网页常用的一些小的图标，借助工具帮我们生成一个字体包，然后就可以像使用文字一样使用图标了。

优点：

1、将所有图标打包成字体库，减少请求；

2、具有矢量性，可保证清晰度；

3、使用灵活，便于维护；

**Font Awesome 使用介绍**

http://fontawesome.dashgame.com/

**定制自已的字体图标库**

http://iconfont.cn/

https://icomoon.io/

**SVG素材**

http://www.iconsvg.com/

## 兼容性

通过http://caniuse.com/ 可查询CSS3各特性的支持程度，一般兼容性处理的常见方法是为属性添加私有前缀，如不能解决，应避免使用，无需刻意去处理CSS3的兼容性问题。

## 高级应用