**语义（semantic） 元素**

HTML5 定了 8 个新的 HTML 语义（semantic） 元素。所有这些元素都是块级 元素。

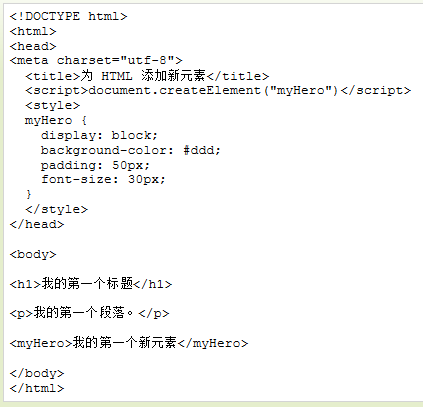
为了能让旧版本的浏览器正确显示这些元素，你可以设置 CSS 的 display 属性值为 block:

header, section, footer, aside, nav, main, article, figure {

display: block;

}

自定义元素myHero



Internet Explorer 8 及更早 IE 版本的浏览器不支持以上的方式。



eg.

新元素

<http://www.runoob.com/html/html5-new-element.html>

**Canvas**

定义画布：<canvas id="myCanvas" width="200" height="100"></canvas>

canvas 元素本身是没有绘图能力的。所有的绘制工作必须在 JavaScript 内部完成

使用 JavaScript 来绘制图像，找到 <canvas> 元素：var c=document.getElementById("myCanvas");

var ctx=c.getContext("2d");

getContext("2d") 对象是内建的 HTML5 对象，拥有多种绘制路径、矩形、圆形、字符以及添加图像的方法。

画布坐标，左上角为（0,0）

下面的两行代码绘制一个红色的矩形：

ctx.fillStyle="#FF0000";

ctx.fillRect(0,0,150,75);

设置fillStyle属性可以是CSS颜色，渐变，或图案。fillStyle默认设置是#000000（黑色）。

fillRect(x,y,width,height) 方法定义了矩形当前的填充方式。x,y为起始点，width和height为绘制大小

画线：ctx.moveTo(0,0);定义线条开始坐标

ctx.lineTo(200,100); 定义线条结束坐标

ctx.stroke();绘制线条方法 "ink" 的方法

绘制圆形：ctx.beginPath();

ctx.arc(95,50,40,0,2\*Math.PI);绘圆方法

ctx.stroke(); "ink" 的方法

绘制文本：ctx.font="30px Arial";定义字体，30px这里指高30px

ctx.fillText("Hello World",10,50); 字为实心

ctx.strokeText("Hello World",10,50);字为空心

渐变：渐变可以填充在矩形, 圆形, 线条, 文本等等, 各种形状可以自己定义不同的颜色。

当我们使用渐变对象，必须使用两种或两种以上的停止颜色。

// Create gradient

var grd=ctx.createRadialGradient(75,50,5,90,60,100);创建一个径向/圆渐变，开始圆心坐标、半径，

结束圆坐标、半径

var grd=ctx.createLinearGradient(0,0,200,0);创建线条渐变，(x,y,x1,y1) x,y为渐变起始点坐标，

x1,y1位渐变结束点坐标

grd.addColorStop(0,"red"); addColorStop()方法指定颜色停止，参数使用坐标来描述，可以是0至1.

grd.addColorStop(1,"white");

// Fill with gradient

ctx.fillStyle=grd; 设置fillStyle或strokeStyle的值为渐变，然后绘制形状，如矩形，文本，或一条线。

ctx.fillRect(10,10,150,80);

图像：var img=document.getElementById("scream");

ctx.drawImage(img,10,10); 把一幅图像放置到画布上的方法，

**SVG**

SVG 指可伸缩矢量图形 (Scalable Vector Graphics)用于定义用于网络的基于矢量的图形使用 XML 格式定义图形图像在放大或改变尺寸的情况下其图形质量不会有损失 是万维网联盟的标准

使用 SVG 的优势在于：

图像可通过文本编辑器来创建和修改, 图像可被搜索、索引、脚本化或压缩,是可伸缩的图像,可在任何的分辨率下被高质量地打印,可在图像质量不下降的情况下被放大

在 HTML5 中，您能够将 SVG 元素直接嵌入 HTML 页面中

<body>

<svg xmlns="http://www.w3.org/2000/svg" version="1.1" height="190">

<polygon points="100,10 40,180 190,60 10,60 160,180"

style="fill:lime;stroke:purple;stroke-width:5;fill-rule:evenodd;">

</svg>

</body>

SVG 与 Canvas两者间的区别

SVG 是一种使用 XML 描述 2D 图形的语言。

Canvas 通过 JavaScript 来绘制 2D 图形。

SVG 基于 XML，这意味着 SVG DOM 中的每个元素都是可用的。您可以为某个元素附加 JavaScript 事件处理器。

在 SVG 中，每个被绘制的图形均被视为对象。如果 SVG 对象的属性发生变化，那么浏览器能够自动重现图形。

Canvas 是逐像素进行渲染的。在 canvas 中，一旦图形被绘制完成，它就不会继续得到浏览器的关注。如果其位置发生变化，那么整个场景也需要重新绘制，包括任何或许已被图形覆盖的对象。

下表列出了 canvas 与 SVG 之间的一些不同之处。

|  |  |
| --- | --- |
| **Canvas** | **SVG** |
| * 依赖分辨率 * 不支持事件处理器 * 弱的文本渲染能力 * 能够以 .png 或 .jpg 格式保存结果图像 * 最适合图像密集型的游戏，其中的许多对象会被频繁重绘 | * 不依赖分辨率 * 支持事件处理器 * 最适合带有大型渲染区域的应用程序（比如谷歌地图） * 复杂度高会减慢渲染速度（任何过度使用 DOM 的应用都不快） * 不适合游戏应用 |

**getCurrentPosition() 方法来获得用户的位置。**

<script>  
var x=document.getElementById("demo");  
function getLocation()  
  {  
  if (navigator.geolocation)  
    {  
    navigator.geolocation.getCurrentPosition(showPosition);  
    }  
  else{x.innerHTML="该浏览器不支持获取地理位置。";}  
  }  
function showPosition(position)  
  {  
  x.innerHTML="Latitude: " + position.coords.latitude +   
  "<br>Longitude: " + position.coords.longitude;   
  }  
</script>

实例解析:

检测是否支持地理定位

如果支持，则运行 getCurrentPosition() 方法。如果不支持，则向用户显示一段消息。

如果getCurrentPosition()运行成功，则向参数showPosition中规定的函数返回一个coordinates对象

showPosition() 函数获得并显示经度和纬度

getCurrentPosition() 方法的第二个参数用于处理错误。它规定当获取用户位置失败时运行的函数：

function showError(error)  
  {  
  switch(error.code)   
    {  
    case error.PERMISSION\_DENIED:  
      x.innerHTML="用户拒绝对获取地理位置的请求。"  
      break;  
    case error.POSITION\_UNAVAILABLE:  
      x.innerHTML="位置信息是不可用的。"  
      break;  
    case error.TIMEOUT:  
      x.innerHTML="请求用户地理位置超时。"  
      break;  
    case error.UNKNOWN\_ERROR:  
      x.innerHTML="未知错误。"  
      break;  
    }  
  }

错误代码：

Permission denied - 用户不允许地理定位

Position unavailable - 无法获取当前位置

Timeout - 操作超时

如需在地图中显示结果，需要访问可使用经纬度的地图服务，比如谷歌地图或百度地图：

function showPosition(position)

{

var latlon=position.coords.latitude+","+position.coords.longitude;

var img\_url="http://maps.googleapis.com/maps/api/staticmap?center="

+latlon+"&zoom=14&size=400x300&sensor=false";

document.getElementById("mapholder").innerHTML="<img src='"+img\_url+"'>";

}

在上例中，使用返回的经纬度数据在谷歌地图中显示位置（使用静态图像）。

watchPosition() - 返回用户的当前位置，并继续返回用户移动时的更新位置（就像汽车上的 GPS）。

clearWatch() - 停止 watchPosition() 方法

下面的例子展示 watchPosition() 方法。

<script>

var x=document.getElementById("demo");

function getLocation()

{

if (navigator.geolocation)

{

navigator.geolocation.watchPosition(showPosition);

}

else{x.innerHTML="该浏览器不支持获取地理位置。";}

}

function showPosition(position)

{

x.innerHTML="纬度: " + position.coords.latitude +

"<br>经度: " + position.coords.longitude;

}

</script>

**<video>** 元素提供了 播放、暂停和音量控件来控制视频。同时<video> 元素元素也提供了 width 和 height 属性控制视频的尺寸.如果设置的高度和宽度，所需的视频空间会在页面加载时保留。。如果没有设置这些属性，浏览器不知道大小的视频，浏览器就不能再加载时保留特定的空间，页面就会根据原始视频的大小而改变。<video> 与</video> 标签之间插入的内容是提供给不支持 video 元素的浏览器显示的。<video> 元素支持多个 <source> 元素. <source> 元素可以链接不同的视频文件。浏览器将使用第一个可识别的格式.

<video> 元素支持三种视频格式： MP4, WebM, 和 Ogg.

属性和方法

<!DOCTYPE html>

<html>

<body>

<div style="text-align:center">

<button onclick="playPause()">播放/暂停</button>

<button onclick="makeBig()">放大</button>

<button onclick="makeSmall()">缩小</button>

<button onclick="makeNormal()">普通</button>

<br>

<video id="video1" width="420">

<source src="mov\_bbb.mp4" type="video/mp4">

<source src="mov\_bbb.ogg" type="video/ogg">

您的浏览器不支持 HTML5 video 标签。

</video>

</div>

<script>

var myVideo=document.getElementById("video1");

function playPause()

{

if (myVideo.paused)

myVideo.play();

else

myVideo.pause();

}

function makeBig()

{

myVideo.width=560;

}

function makeSmall()

{

myVideo.width=320;

}

function makeNormal()

{

myVideo.width=420;

}

</script>

</body>

</html>

在<audio> 与 </audio> 之间你需要插入浏览器不支持的<audio>元素的提示文本 。

<audio> 元素允许使用多个 <source> 元素. <source> 元素可以链接不同的音频文件，浏览器将使用第一个支持的音频文件,control 属性供添加播放、暂停和音量控件。

<audio controls>  
  <source src="horse.ogg" type="audio/ogg">  
  <source src="horse.mp3" type="audio/mpeg">  
您的浏览器不支持 audio 元素。  
</audio>

目前, <audio>元素支持三种音频格式文件: MP3, Wav, 和 Ogg

**Input新类型**

color 类型用在input字段主要用于选取颜色

date 类型允许你从一个日期选择器选择一个日期

datetime 类型允许你选择一个日期（UTC 时间）（谷歌不支持）

datetime-local 类型允许你选择一个日期和时间 (无时区).

email 类型用于应该包含 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

month 类型允许你选择一个月份。

number 类型用于应该包含数值的输入域，使用下面的属性来规定对数字类型的限定：

max- 规定允许的最大值

min - 规定允许的最小值

step - 规定合法的数字间隔（如果 step="3"，则合法的数是 -3,0,3,6 等）

value - 规定默认值

数量 ( 1 到 5 之间 ): <input type="number" name="quantity" min="1" max="5">

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

属性来规定对数字类型的限定同number

search 类型用于搜索域，比如站点搜索或 Google 搜索。

tel 定义输入电话号码字段

time 类型允许你选择一个时间,定义可输入时间控制器（无时区）

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

week 类型允许你选择周和年。

**<datalist>** 元素规定输入域的选项列表。

<datalist> 属性规定 form 或 input 域应该拥有自动完成功能。当用户在自动完成域中开始输入时，浏 览器应该在该域中显示填写的选项：使用 <input> 元素的列表属性与 <datalist> 元素绑定.(下拉提示）

<input> 元素使用<datalist>预定义值:

<input list="browsers">

<datalist id="browsers">

<option value="Internet Explorer">

<option value="Firefox">

<option value="Chrome">

<option value="Opera">

<option value="Safari">

</datalist

**<keygen>** 元素的作用是提供一种验证用户的可靠方法。<keygen>标签规定用于表单的密钥对生成器字段。当提交表单时，会生成两个键，一个是私钥，一个公钥。私钥（private key）存储于客户端，公钥（public key）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（client certificate）。

带有keygen字段的表单:

<form action="demo\_keygen.asp" method="get">  
用户名: <input type="text" name="usr\_name">  
加密: <keygen name="security">  
<input type="submit">  
</form>

**<output>** 元素用于不同类型的输出，比如计算或脚本输出

将计算结果显示在 <output> 元素:

<form oninput="x.value=parseInt(a.value)+parseInt(b.value)">0  
<input type="range" id="a" value="50">100 +  
<input type="number" id="b" value="50">=  
<output name="x" for="a b"></output>  
</form>