要整理的html新特性：

HTML 5 视频

HTML 5 音频

HTML 5 拖放

HTML 5 Canvas

HTML 5 Canvas vs. SVG

HTML 5 Web 存储

HTML 5 应用程序缓存

HTML 5 Web Workers

<http://www.bootcss.com/p/lesscss/#-variables>

见素材html5新特性

HTML 5 视频

支持三种视频格式：Ogg，MPEG 4，WebM

<video></video> 之间的内容是供不支持video元素的浏览器显示的

video 元素允许多个 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>

<source src="video.ogg" type="video/ogg">

<source src="video.mp4" type="video/mp4">

</video>

属性 值 描述

autoplay autoplay 视频在就绪后马上播放。

controls controls 向用户显示控件

loop loop 循环播放

preload preload 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。

src url 要播放视频的 URL。

height pixels 设置视频播放器的高度。

width pixels 设置视频播放器的宽度。

使用 DOM 进行控制，下面列出了大多数浏览器支持的视频方法、属性和事件：

方法 属性 事件

play() currentSrc play

pause() currentTime pause

load() videoWidth progress

canPlayType videoHeight error

duration timeupdate

ended ended

error abort

paused empty

muted emptied

seeking waiting

volume loadedmetadata

height

width

注释：在所有属性中，只有 videoWidth 和 videoHeight 属性是立即可用的。其他属性在视频的元数据已加载后才可用。

HTML 5 音频

支持三种音频格式：Ogg Vorbis，MP3，Wav

<audio></audio> 之间的内容是供不支持video元素的浏览器显示的

audio 元素允许多个 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可识别的格式。

<audio>

<audio src="audio.ogg" type="audio/ogg">

<audio src="audio.mp3" type="audio/mpeg">

</audio>

属性 值 描述

autoplay autoplay 音频在就绪后马上播放。

controls controls 向用户显示控件

loop loop 循环播放

preload preload 音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。

src url 要播放的音频的 URL。

HTML 5 拖放

拖放即抓取对象以后拖到另一个位置。任何元素都能够拖放。

<!DOCTYPE HTML>

<html>

<head>

<script type="text/javascript">

function ondragover(ev){

ev.preventDefault();

}

function ondragstart(ev){

ev.dataTransfer.setData("Text",ev.target.id);

}

function ondrop(ev){

ev.preventDefault();

var data=ev.dataTransfer.getData("Text");

ev.target.appendChild(document.getElementById(data));

}

</script>

</head>

<body>

<div id="div"

ondragover="ondragover(event)"

ondrop="ondrop(event)">

</div>

<img id="img"

src="img.png"

draggable="true"

ondragstart="ondragstart(event)"

/>

</body>

</html>

HTML 5 Canvas

canvas使用javascript在网页上绘制图像。

canvas是一个矩形区域，可以控制其每一像素。

html中有

<canvas id="canvas" width="200" height="100"></canvas>

js中有

var c=document.getElementById("canvas");

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

绘制一条线

cxt.moveTo(10,10);

cxt.lineTo(150,50);

cxt.lineTo(10,50);

cxt.stroke();

绘制一个红色的圆

cxt.fillStyle="#FF0000";

cxt.beginPath();

cxt.arc(70,18,15,0,Math.PI\*2,true);

cxt.closePath();

cxt.fill();

绘制一个红色的矩形

cxt.fillStyle="#FF0000";

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

绘制渐变背景

var grd=cxt.createLinearGradient(0,0,175,50);

grd.addColorStop(0,"#FF0000");

grd.addColorStop(1,"#00FF00");

cxt.fillStyle=grd;

cxt.fillRect(0,0,175,50);

把一幅图像放置到画布上

var img=new Image()

img.src="flower.png"

cxt.drawImage(img,0,0);

HTML 5 Canvas vs. SVG

Canvas

使用JavaScript来绘制2d图形。

依赖分辨率

不支持事件处理器

适合游戏

能够以 .png 或 .jpg 格式保存结果图像

SVG

使用xml描述2d图形

不依赖分辨率

支持事件处理器

不适合游戏应用

HTML 5 Web 存储

localStorage

localStorage.变量名=变量值;

判断是否已经有变量名 if (localStorage.变量名){}

sessionStorage

sessionStorage.变量名=变量值;

判断是否已经有变量名 if (sessionStorage.变量名){}

HTML 5 应用程序缓存

<html manifest="文件名.appcache">

在web服务器上为manifest文件配置正确的MIME-type即text/cache-manifest。

manifest 文件可分为三个部分：

CACHE MANIFEST 在此标题下列出的文件将在首次下载后进行缓存

NETWORK 在此标题下列出的文件需要与服务器的连接，且不会被缓存

FALLBACK 在此标题下列出的文件规定当页面无法访问时的回退页面（比如 404 页面）

\*指所有资源

#开头的是注释行，更新注释行中的日期和版本号是一种使浏览器重新缓存文件的办法。

一旦应用被缓存，它就会保持缓存直到发生下列情况：

用户清空浏览器缓存

manifest 文件被修改

由程序来更新应用缓存

一旦文件被缓存，则浏览器会继续展示已缓存的版本，即使修改了服务器上的文件

浏览器对缓存数据的容量限制可能不太一样（某些浏览器设置的限制是每个站点 5MB）。

HTML 5 Web Workers

web worker 是运行在后台的javascript，不会影响页面的性能。

创建web worker文件webWorker.js

使用postMessage()方法向html页面传回一段消息。

检测 Web Worker 支持并创建 Web Worker 对象

if(typeof(Worker)!=="undefined"){

//支持

w=new Worker("webWorker.js");

w.onmessage=function(event){

//event.data中存有来自webWorker的数据。

};

}else{

//不支持

}

终止 Web Worker

w.terminate();

由于 web worker 位于外部文件中，因此无法访问下例javascript对象：

window 对象

document 对象

parent 对象