**HTML5知识点**

**DAY01**

**一：表单**

**1.1表单新类型**

**email类型，url类型,tel类型，date类型,number类型，range类型**

**1.2表单新元素**

**datalist元素-类似于select元素**

**datalist元素-预定义数据内容（option）**

**使用<input>元素的list属性**

**progress元素-进度条**

**max-设置最大值**

**value-设置当前进度**

**mater元素-刻度**

**min - 设置最小值**

**max - 设置最大值**

**value - 设置当前值**

**low - 设置低预警值**

**high - 设置高预警值**

**output元素-输出**

**1.3表单新属性**

**placeholder - 给出默认提示**

**autofocus - 自动获取焦点**

**multiple - 允许输入多个值，多个值之间使用”,”**

**form允许表单元素定义在表单外**

**1.4表单心验证**

**required验证是否为空**

**min和max验证最小和最大**

**pattern验证正则表达式是否匹配**

**minlength和maxlength**

**minlength并不是HTML5的新特性**

**maxlength不允许输入超过maxlength长度内容**

**1.5有效状态**

**valid-验证是否成功**

**true表示验证成功**

**valueMissing-验证是否为空（true）**

**patternMismatch验证类型是否匹配**

**typeMismatch验证类型是否匹配**

**rangeMismatch验证是否小于min**

**stepMismatch验证是否与step匹配**

**customError配合setCustomValidity()使用**

**1.6表单验证状态：**

**1.6.1 tooLong**

**作用：判断当前元素的长度是否正确**

**用法：配合maxlength属性使用**

**注意：**

**使用maxlength属性后，实际不可能出现长度大于maxlength的值**

**tooLong很难出现这种情况**

**扩展内容**

**代码的编写 - 逻辑的完整性**

**判断value值是否为空**

**value==””||value==null**

**tooLong状态**

**1.6.2 rangeUnderflow状态**

**如果输入内容小于min属性声明的值，那么这个状态就true**

**if(age.validity.rangeUnderflow){**

**alert(“年龄不符合要求”);**

**}else{**

**alert(“通过”);**

**}**

**作用：判断当前元素值是否小于min值**

**用法：配合min属性使用**

**注意：并不能于max进行比较。**

**1.6.3 stepMismatch**

**作用：判断当前元素之是否与step设置相符**

**用法：配合step使用**

**注意：并不能与min或max属性比较**

**1.6.4 customError状态**

**用法：配合setCustomValidity()方法**

**如果使用该方法，该状态返回 true**

**1.6.5setCustomValidity()方法**

**作用：修改验证错误后的默认提示信息（设置自定义的错误提示内容）**

**问题：使用该方法设置错误信息。**

**当输入正确时，调用该方法将信息设置为空（“”）**

**不能使用上述有效状态的任何一种判断输入是否正确（所有状态返回false）**

**一旦使用该方法修改默认错误提示信息后，及时输入正确，错误提示依旧存在，导致逻辑错误。**

**如何解决：**

**判断如果用户输入正确的话，将该方法设置提示内容修改为空。**

**问题**

**使用valid状态判断输入是否正确**

**一旦使用该方法，validityState对象所有状态都返回false**

**1.6.6 valid**

**作用：判断当前元素之是否正确**

**\*注意**

**该状态返回true,表示验证成功**

**该状态返回false,表示验证失败**

**DAY02**

**一：音视频处理**

**1.1视频处理**

**目前用于替代flash技术最好的选择-html5**

**几乎所有的浏览器原生支持<video>元素**

**性能更高**

**智能**

**如何实现视频处理---<video>**

**<video> 元素处理视频**

如果当前浏览器不支持<video>元素

在<video>元素内编写提示内容

属性

src引入视频文件的路径

autoplay自动播放视频

**<video>支持的视频格式**

MP4格式-目前比较主流

OGG格式-多用于移动端

webM格式-目前唯一支持超高清格式

**<video>元素的属性**

src引入视频的路径

autoplay自动播放视频

controls提供控制面板

loop表示循环播放

**poster设置播放之前显示的图片**

width和height设置显示视频的宽度和高度

preload预加载

auto（默认值）自动加载

none不加载

metadata只加载视频的基本信息(不含视频）

**高级内容**

**方法**

**play() 播放视频**

**pause(）暂停视频**

**load（）加载视频**

**canPlayType()-判断当前浏览器是否支持指定的视频格式**

**事件**

**play当事件播放时被触发**

**pause当事件暂停时被触发**

**属性**

**pause判断视频是否暂停**

**Boolean值,true表示暂停，否则播放**

**ended判断视频是否结束**

**duration返回当前视频的时长**

**currentTime返回视频播放的当前位置。**

**1.2音频处理**

**<audio>元素**

**第一种，只支持一种音频格式**

**<dudio src=”音频文件的路径”></audio>**

**第二种 - 同时导入多个音频格式**

**<audio>**

**<source src=”一种音频格式” />**

**<source src=”一种音频格式” />**

**<source src=”一种音频格式” />**

**</audio>**

**<audio>元素支持音频格式**

**MP3目前最流行**

**OGG格式**

**WAV比较老的格式**

**<audio>元素的属性**

**outoplay**

**loop**

**controls**

**Canvas(画布）**

**基本内容**

**简单的来说，HTML提供的新元素<canvas>**

**canvas在HTML页面中提供画布功能**

**在画布中绘制各种图形**

**canvas绘制的图形与html页面无关**

**无法通过DOM获取绘制的图形**

**无法为绘制的图像绑定DOM事件**

**只能使用Canvas提供的API**

**canvas用途**

**在HTML页面中绘制图标（例如例如柱状图，饼状图等）**

**网页游戏 - flash技术**

**使用html5中的canvas**

**使用canvas的步骤：**

**1. 在html页面中定义<canvas>元素**

**2. 在javascript代码中**

**（1）获取<canvas>元素**

**使用width和height属性-没有任何问题**

**使用CSS样式**

**通过style属性设置 - 绘制图形被拉伸**

**通过style元素设置 - 按比例**

**（2）创建画布对象**

**getContext(“2d”)方法**

**该方法返回2D和3D效果两种画布对象**

**注意：参数必须是”2d”或”3d” 字母必须是小写的**

**使用canvas提供的API**

**绘制图像**

**绘制矩形**

**fillRect(x,y,width,height) - 实心矩形**

**x和y - 矩形的左上角坐标值**

**width - 设置矩形的宽度**

**height- 设置矩形的高度**

**strkeRect(x,y,width,height) - 空心矩形**

**clearRect(x,y,width,height) - 清除指定区域的矩形**

**设置颜色**

**fillStyle - 设置填充颜色**

**strokeStyle - 设置描边颜色**

**GlobalAlpha - 设置透明度（0-1）**

**设置渐变**

**线性渐变 - createLinearGradient(x1,y1,x2,y2)**

**具有基准线 - 起点（x1,y1）和终点(x2,y2)**

**扇形（射线）渐变 - createRedialGradient(x1,y1,r1,x2,y2,r2)**

**具有柱形（锥形）- 两个圆的面积**

**参数**

**X1和y1 - 第一个圆的圆心坐标值**

**r1-第一个圆的半径**

**x2,y2-第二个圆的圆心坐标值**

**r2-第二个圆的半径**

**绘制文字**

**方法**

**fillText(text,x,y) - 实心文字**

**text绘制的文字内容**

**x和y绘制的坐标**

**strokeText(text,x,y) 空心文字**

**属性**

**font类似于css中的font属性**

**textAlign设置文字的水平方向对齐**

**Left**

**Center**

**right**

**textBaseline设置文字垂直方向对齐**

**top顶部对齐**

**middle（垂直）居中对齐**

**bottom底部对齐**

**hanging悬挂基线**

**alphabetic-字母基线**

**注意：**

**无论是水平方向还是垂直方向对齐，是基准线对齐，并不是文字对齐**

**无论是水平方向还是在垂直方向对齐，他都不是一个必要的属性（不适用也可以）**

**阴影效果**

**shadowColor 设置阴影的颜色**

**shadowOffsetx设置水平方向的阴影**

**shadowOffsety设置垂直方向的阴影**

**shadowBlur设置阴影的模糊程度**

**创建路径**

**(表示）方法**

**beginPath()表示开始创建路径**

**closePath() 表示借宿创建路径**

**设置方法**

**rect(x,y,width,height)-设置矩形形状**

**x和y设置矩形的左上角坐标值**

**width,和height设置矩形的宽度和高度**

**arc(x,y,radius,stratAngle,endAngle,direction)**

**设置圆形形状**

**x和y设置圆形的圆心坐标值**

**radius设置圆形的半径**

**startAngle,endAngle设置圆形的起始位置**

**direction按照顺时针或者逆时针绘制**

**Boolean默认值为false,表示的是顺时针**

**绘制方法**

**stroke()绘制轮廓**

**fill()绘制填充**

**注意：context.beginPath()方法必须放在绘制方法的墙面**

**绘制线条（直线和折线）- 创建路径**

**moveTo（x,y）设置这条线的(直线/折线）的起点坐标**

**moveLine(x,y) 设置这条线的(直线/折线）的终点坐标**

**线型**

**到目前为止，所有的画布操作都是用相同的线条，即默认线型。实际上线条的宽度、断电都可以根据实际绘图需要进行调整。**

**以下是修改线型的属性：**

**lineWidth:指定线条粗细，默认值是1.0**

**canvas.lineWidth=”10”;**

**设置线条**

**lineWidth() 设置线条的宽度**

**默认值为1,单位是px**

**lineCap() :设置线条断电的形状**

**butt - 默认值,向线条的每个末端添加平直的边缘**

**round-向线条的每个末端添加圆形线帽（圆角）**

**square-向线条的每个末端添加正方向线帽（正方向）**

**注意：round和square会使线条略变微长**

**lineJoin-设置两条线焦点的形状**

**miter默认值，尖角**

**round圆角**

**bevel斜角**

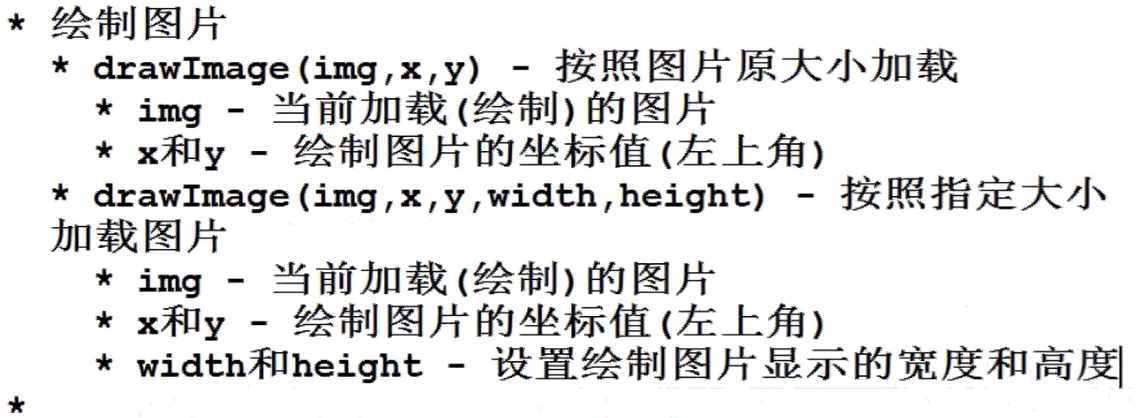
**miterLimit配合lineJoin使用**

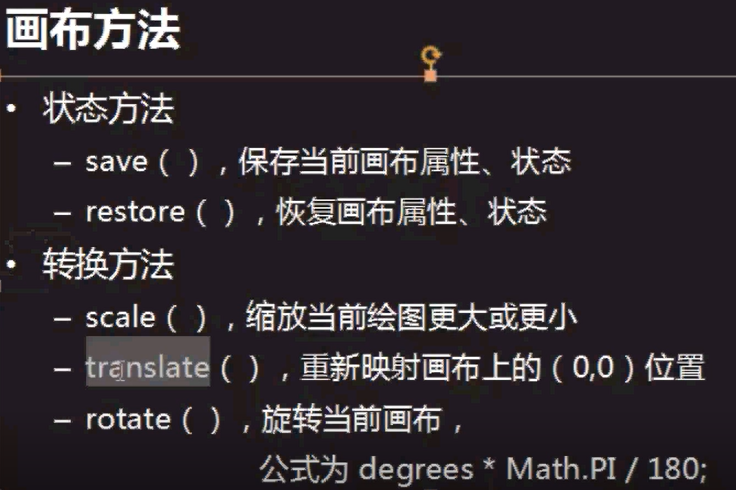
**lineJoin设置为miter,该属性设置尖角的延伸范围**

**canvas处理图片**

**绘制图片**

**drowImage(img,x,y) 按照图片原大小加载**





**scale(x,y) 缩放（缩小或者放大）**

**x表示水平方向的缩放**

**y表示垂直方向的缩放**

**参数的取值**

**如果为1的话，表示不缩放（原大小）**

**如果小于1的话，表示缩小**

**如果大于1的话，表示放大**

**translate（x,y）重新定位(x,y)**

**Rotate(旋转角度)旋转画布**

**公式为degrees\*Math.PI/180;**

**扩展内容**

**前端web开发时，建议使用的浏览器**

**Chrome**

**Firefox**

**Safari (苹果系统常用）**

**内向问题**

**面试 - 你认为你的性格，是外向还是内向**

**内向-沟通能力比较差，团队协作出问题。**

**外向-担心工作时，不够专心。**

**良好的沟通能力**

**程序员-形成程序员思维**

**都是学习和培养**

**Chart.js是一个简单，面向对象，为设计者和开发人员准备的图标绘制工具库。**

**官网地址：<http://www.chartjs.org/>**

**作用-提供各种图表**

**提供6中图表**

**四：SVG**

**4.1基本内容**

**svg（Scalable Vector Graphics）是一种使用xml技术标书二位图形的语言**

**SVG可以使用三种方式描述微微入刑：**

**-矢量图，例如直线或者曲线构成的路径。**

**-图片（image）**

**-文本（text）**

**svg并不属于HTML5专有内容**

**html5提供有关svg原生的内容**

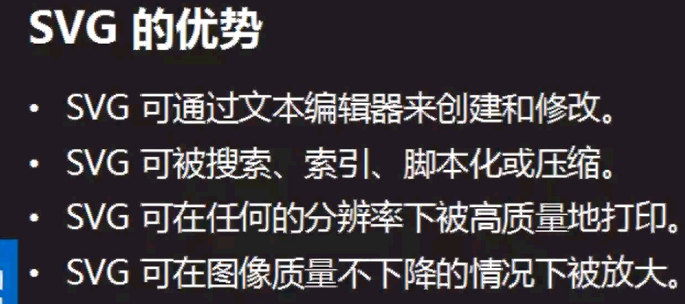
**在HTML5出现之前，就有svg内容**

**svg，简单的来说就是矢量图**

**svg的扩展名为“.svg”**

**svg使用的是SML语法**

**4.2**



**绘制图形**

**矩形元素**

**<rect x=”” y=”” width=”” height=”” />**

**圆形元素**

**<circle cx=”” cy=”” r=”” />**

**椭圆元素**

**<ellipse cx=”” cy=”” rx=”” ry=”” />**

**直线元素**

**<line x1=”” y1=”” x2=”” y2=””/>**

**折线元素**

**<polyline points=””>**

**points - 设置起点、折点以及终点  
 x和y之间使用","  
 多个点之间用空格分开  
折线的特点  
 默认将折线中的区域(起点到终点），默认提供黑色**

**多边形元素**

**<polygon points=””/>**

**特效元素**

**渐变**

**线型渐变 用<linearGradient>标签定义**

**放射性（扇形渐变）渐变 用<radiaGradient>标签定义**

**滤镜 - 高斯模糊**

**滤镜使用<filter>元素**

**注意 - 定义在<defs>元素中**

**扩展内容**

**前端开发工具**

**Aptana Studio 3 - 代码提示**

**Webstrom - 国内前端开发人员神器**

**实际开发中**

**多用SVG**

**不失真**

**可被搜索**

**页面优化- 减烧外部链接**

**<a href=””>**

**<img src=””>**

**Canvas - html绘制图片**

**实际运行中，是以图片的形式出现（.png）**

**不被搜索引擎抓取**

**放大后失真**

**游戏版本**

**480\*680**

**1024\*768**

**SVG内容**

**内容量非常大**

**静态绘制图形**

**动态动画效果**

**专门提供事件**

**在实际开发中的使用**

**svg在将来的工作开发，使用率并不高**

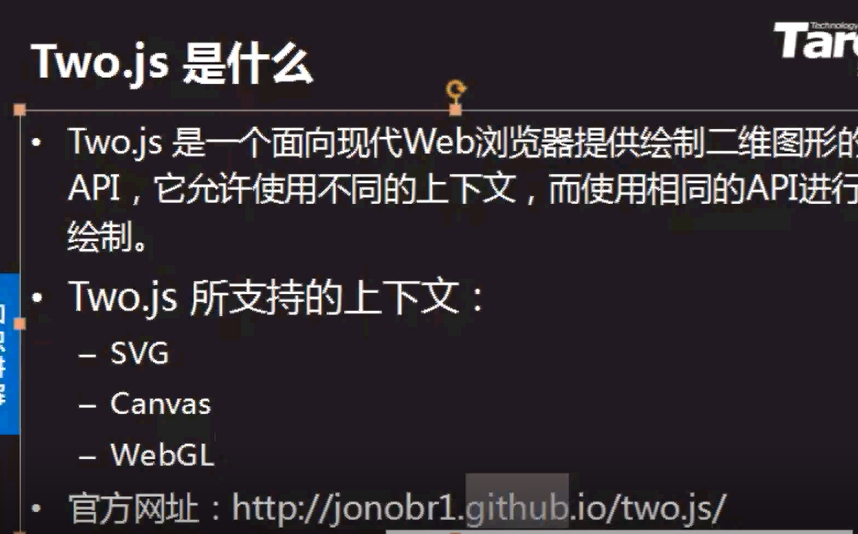
**svg图片一般都是有UI设计师来完成**

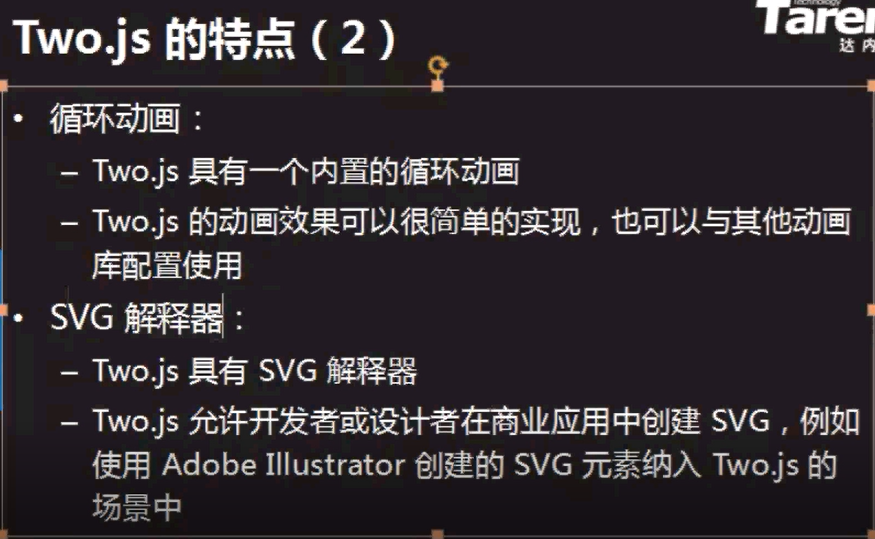
**目前网络上很多专业提供现成的svg图片的网站**

**五：**

**three.js - 专门用于绘制三维图形**

**two.js - 专门用于绘制二位图形**





**Web前端 - 移动端**

**移动智能终端**

**iphone - ios系统 - Object-c|Swfit**

**Andoroid系统**

**Windows Mobile系统**

**BlackBerry系统 - 企业级应用 QNX**

**WebOS系统 - 全键盘+触摸屏**

**塞班 - 诺基亚**

**MeeGo - Inter和诺基亚**

**移动扩平台 - HTML / CSS / jAVASCRIPT**

**一次编写，到处运行（phoneGap）**

**游戏分类（画面）**

**2D效果**

**2.5D效果-实际还是2D效果，模拟3D效果**

**3D效果**

**作为前端web开发人员的原则**

**眼见为实 - 不适用我们前端人员**

**HTML5的案例真正难在逻辑(JS代码）**

**总结 - html5离不开javascript**

**回答疑问**

**第二阶段的javascript,抓紧学习**

**个人并不建议 - 留级**

**编写简历时 - 项目经验**

**经常多上网，多做网上的案例（先从小案例入手）**

**如何成为优秀的程序员**

1. **自学能力很强**

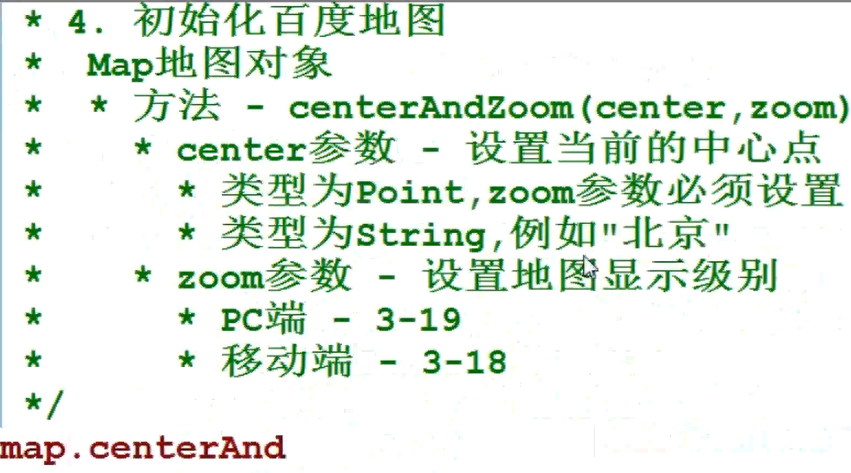
**网上的案例**

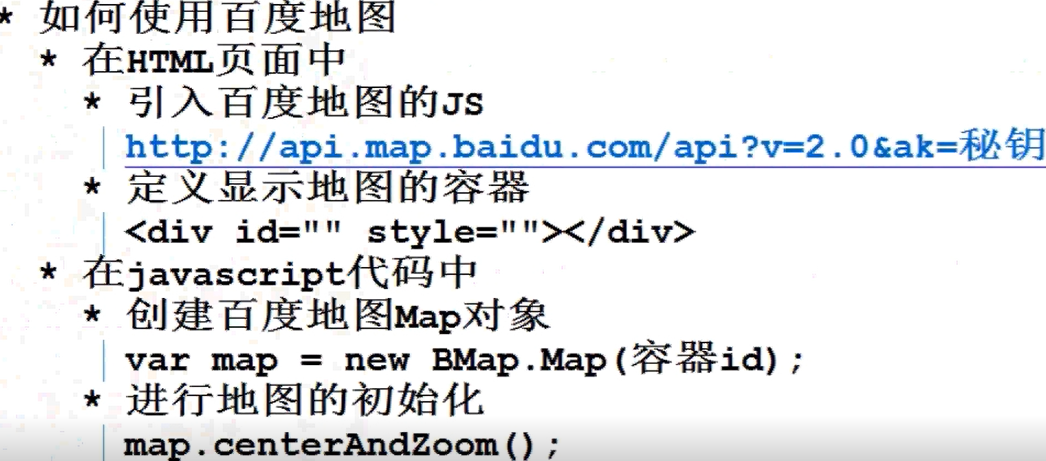
**网上的资料**

1. **善于归纳总结**

**第5天**

**百度地图的嵌入**





**Geolocation类**

**作用：返回用户当前的位置**

**问题:依靠浏览器来实现当前位置定位的功能**

**构造器：Geolocation()**

**方法**

**getCurrentPosition(callback)方法**

**作用：返回用户当前位置**

**参数callback**

**定位成功，该回调函数具有一个**

**GeolocationResult参数**

**方法：**

**getPoint（address,callback,city）方法**

**作用 - 对指定的地址进行解析**

**参数**

**address - 设置解析的地址内容**

**callbac - 回调函数**

**地址定位成功，具有Poiunt参数**

**city-当前中心城市**

**构造器 - InfoWindow（addr,options)**

**addr - 这是的地址**

**options- 设置信息窗口的设置**

**HTML5中拖拽**

**托放（拖拽）API**

**实现拖拽效果**

**要拖拽的文件是什么？- 源元素**

**要拖拽到哪里去？- 目标元素**

**目前实现拖放效果**

**使用原生DOM就能实现 - 做麻烦**

**使用它jQuery的插件 - 拖拽效果**

**HTML5中拖拽**

**源元素事件**

**dragstart - 当鼠标开始拖放时，被触发的**

**drag- 类似于mousemove ,表示拖放过程**

**dragend - 当鼠结束托放时被触发**

**目标元素事件**

**drop当鼠标实现拖放效果时被触发/在目标元素中投放源元素**

**dragover当对象拖放到目标元素被触发/进入目标元素时被触发**

**dragenter当鼠标实现拖放效果时被触发/第一次进入目标元素时被触发**

**dragleave - 当鼠标拖放离开目标元素被触发**

**默认情况下,该事件没有被触发**

**原因-html页面默认情况下，不允许拖放**

**称之为HTML页面的默认行为**

**解决：阻止页面的默认行为**

**事件对象event.preventDefault()方法**

**dataTransfer对象**

**作用 - 类似于window系统的剪切板的功能**

**功能：**

**以将源元素的信息（数据）,存储在这里**

**将存储在该对象的源元素信息，提供给目标元素**

**方法**

**setData()-设置（源元素）数据**

**在源元素事件中使用**

**getData()-获取设置的数据**

**在目标元素中使用**

**clearData-清除（设置的）数据**

**优点-提高性能**

**所有的数据内容，存储在浏览器内存中**

**当使用完毕数据内容时，清除**

**setDragImage()**

**作用-修改拖放过程中，鼠标根数的图片效果**

**用法-drag,dragstart等事件**

**注意-实际操作过程中，该方法几乎不用**

**DAY 06**

**- Web worker**

**运行在后台的javascript**

**基本内容**

**单线程和多线程**

**woker可以模拟多线程的效果**

**-可以充分利用多核CPU的优势**

**-对多线程支持非常好**

**-不会影响页面的技能**

**-不能访问Web页面和使用DOM API**

**注意：在worker中只能使用javascript中的ECMA**

**目前主浏览器都支持Worker,除IE8**

**Worker提供 API**

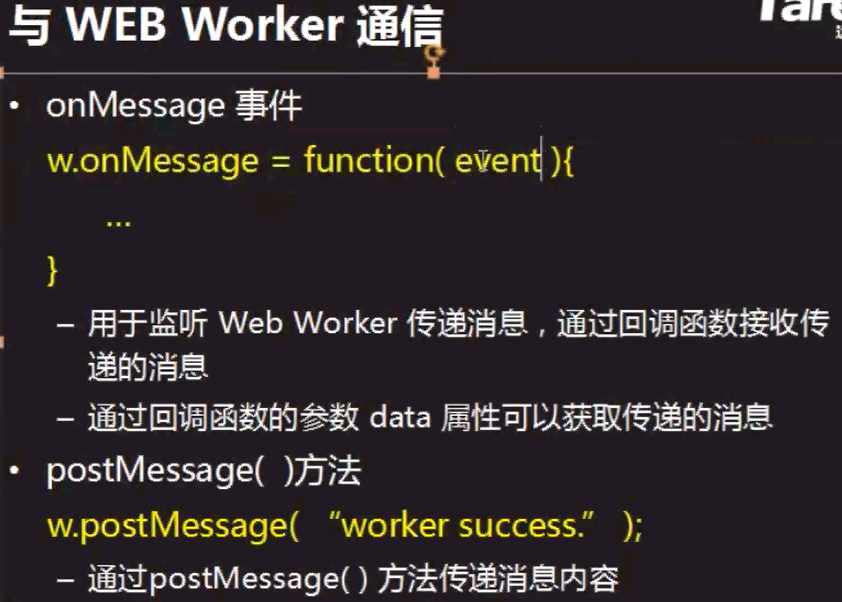
**检查当前浏览器是否支持worker**

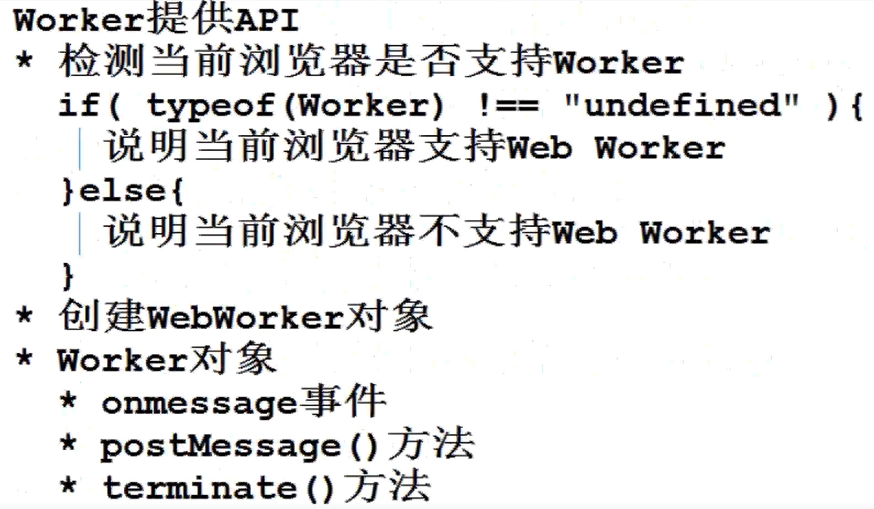
**创建webWorker对象**

**onmessage事件**

**postMessage（）方法**

**terminate（）方法**



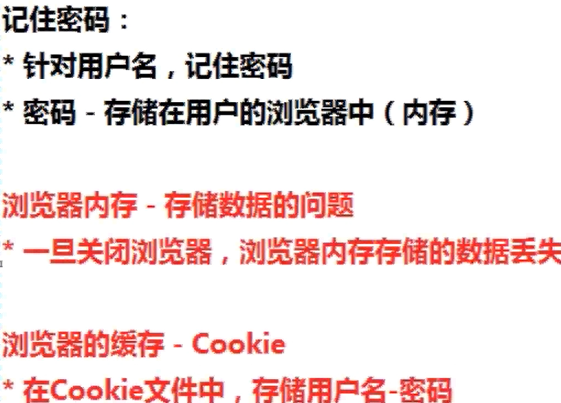


**- Web 存储API**

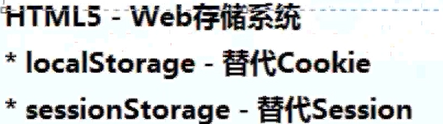
**基本内容**

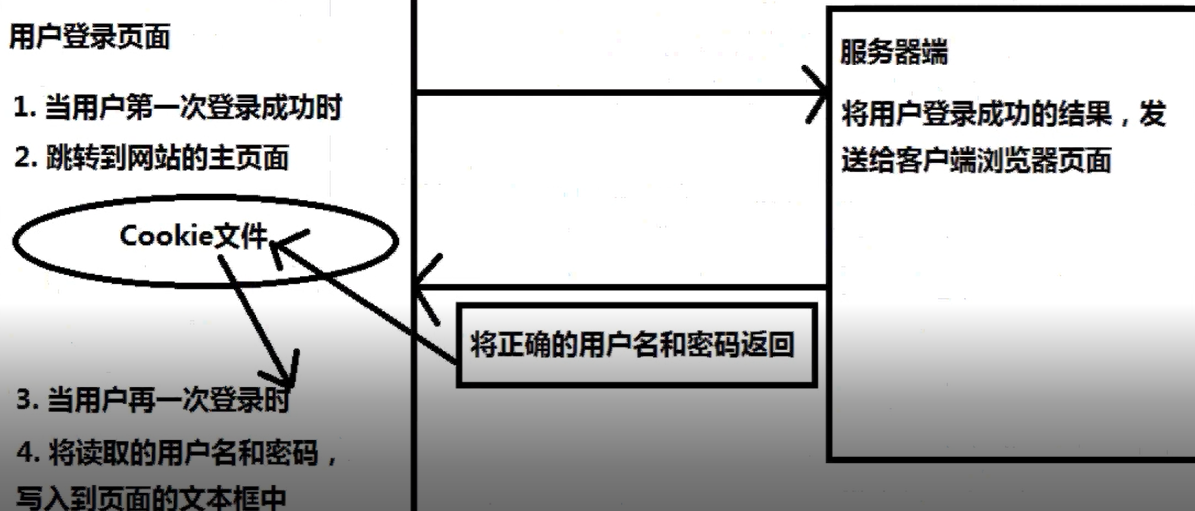
**数据的仓库 - 用于存储数据**

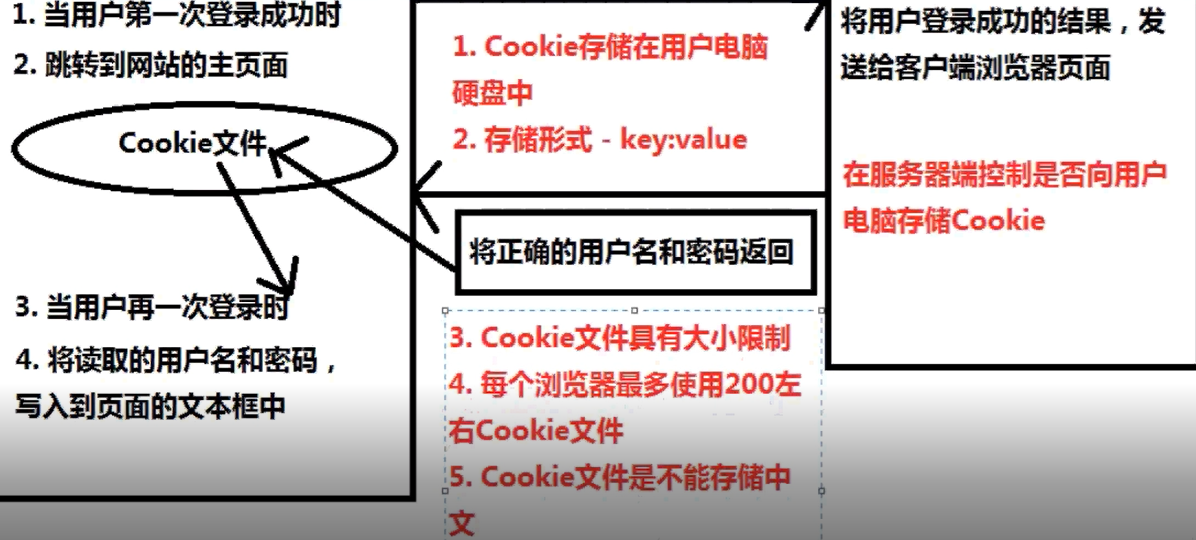
**原始的存储数据：**



**Html5中**







**HTML5存储系统**

**LocalStorage(本地存储) - 代替cookie**

**sessionStorage(绘画存储) - 代替Session（类似于内存）**

**-sessionStorage**

**setItem(key,value)方法**

**作用 - 设置数据（新增数据）**

**参数：**

**key-作为存储数据的标示（唯一，不可重复）**

**value-存储的数据内容（number|string）**

**返回值-key对应的value值**

**getItem(key)方法**

**作用-获取数据（读取数据）**

**参数：**

**Key-根据key获取对应的数据内容**

**返回值-key对应的value值**

**key()方法**

**作用：根据索引值返回对应的key**

**参数：**

**Index-索引值**

**返回值-返回key值**

**removeItem(key)方法**

**作用-删除数据**

**参数：**

**key-根据key删除指定数据内容**

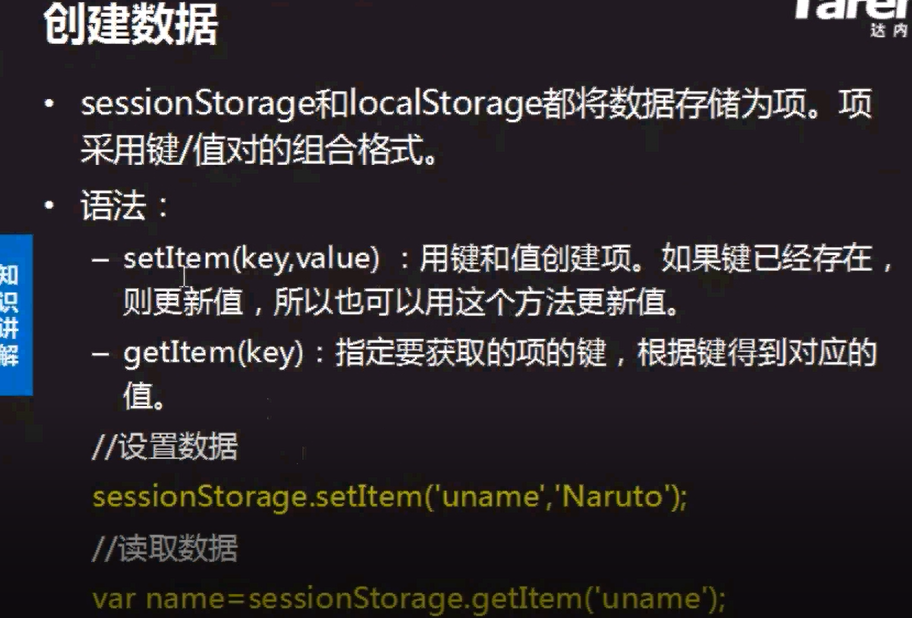
**clear()方法**

**作用：将存储系统的所有数据删除（清空）**

**length属性**

**作用：返回当前存储系统的数据个数**

**-localStorage**



**sessionStorage**

**对于数据的操作**

**新增 - setItem()**

**查询 - getItem()**

**Lenght属性**

**Key（index）**

**删除 -removeItem()**

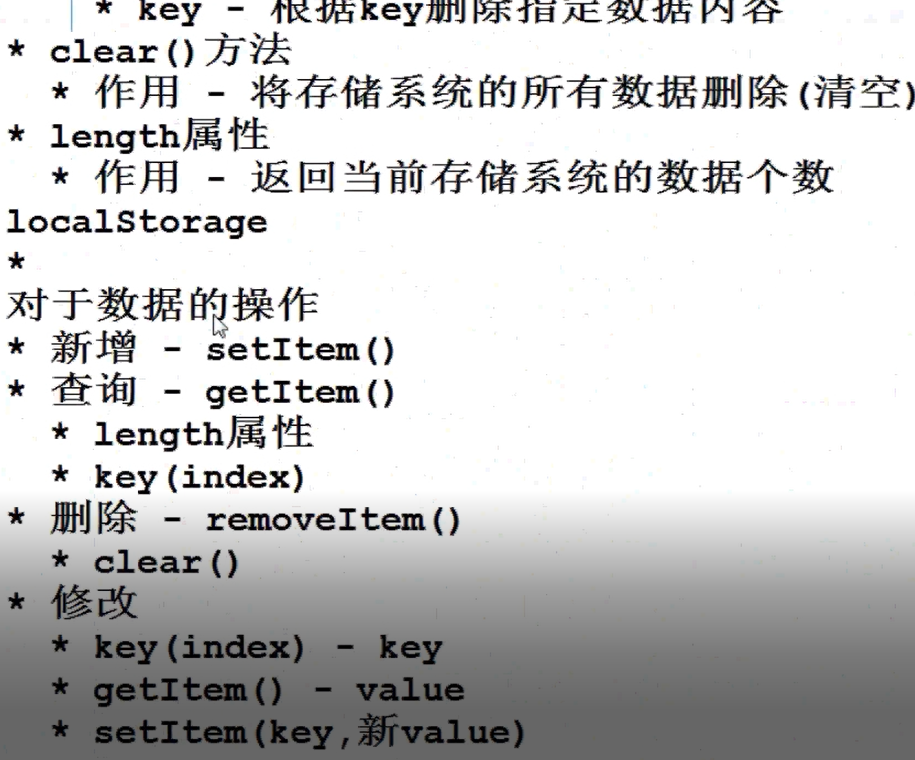
**Clear()**

**修改**

**Key(index） - key**

**getItem() value**

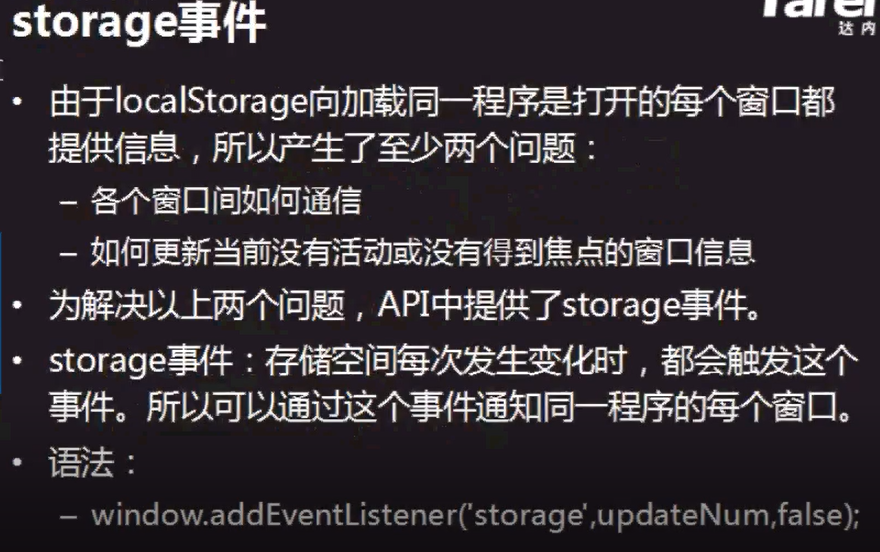
**setItem(key,新value)**

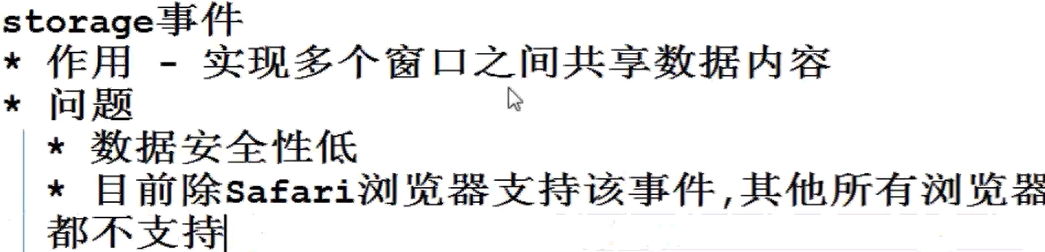


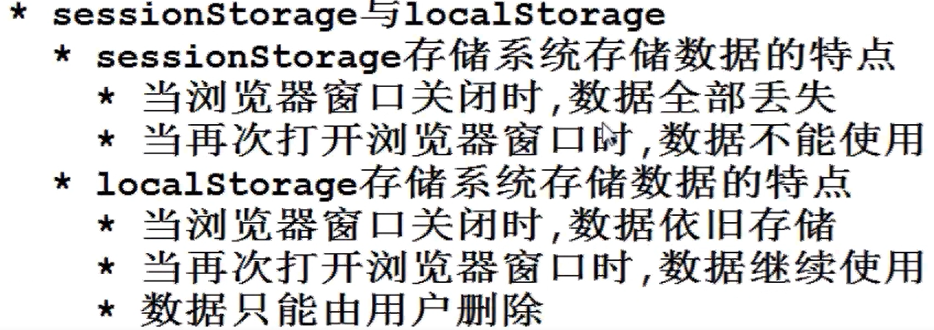
**localStorage**

**提供的属性和方法与sessionStorage一致**

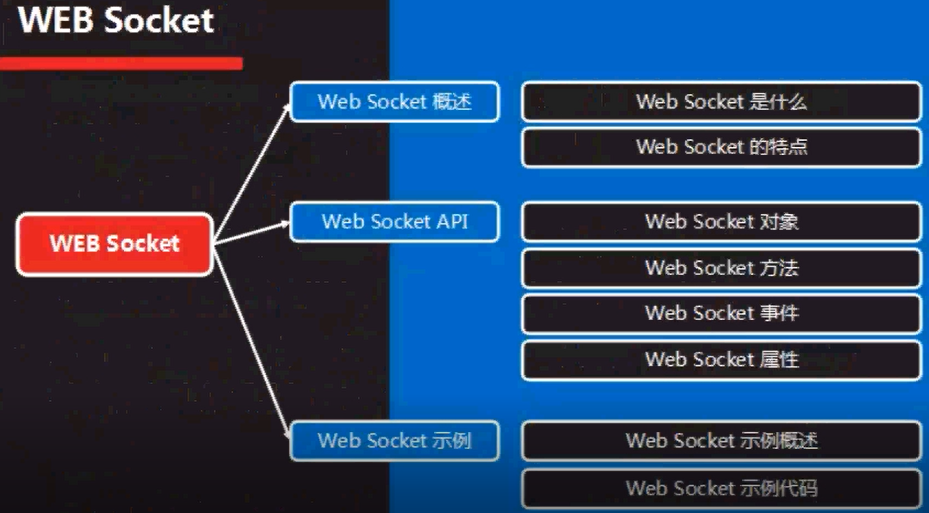
**与sessionStorage相比就多了一个事件**

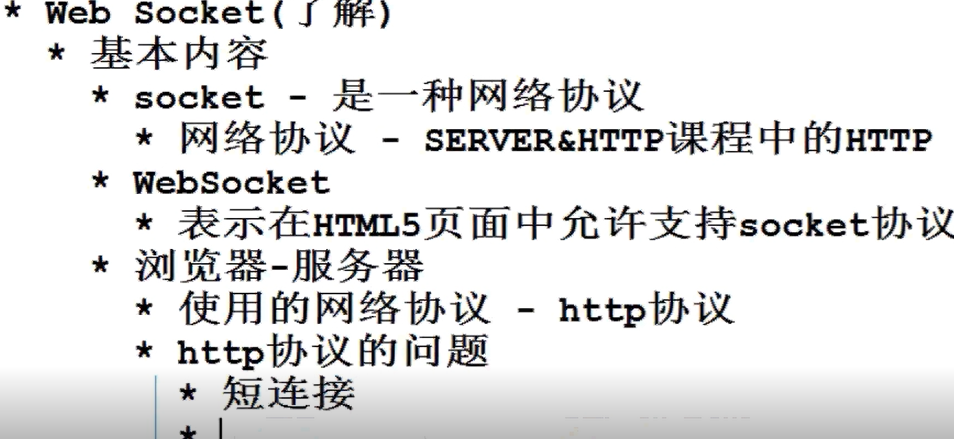


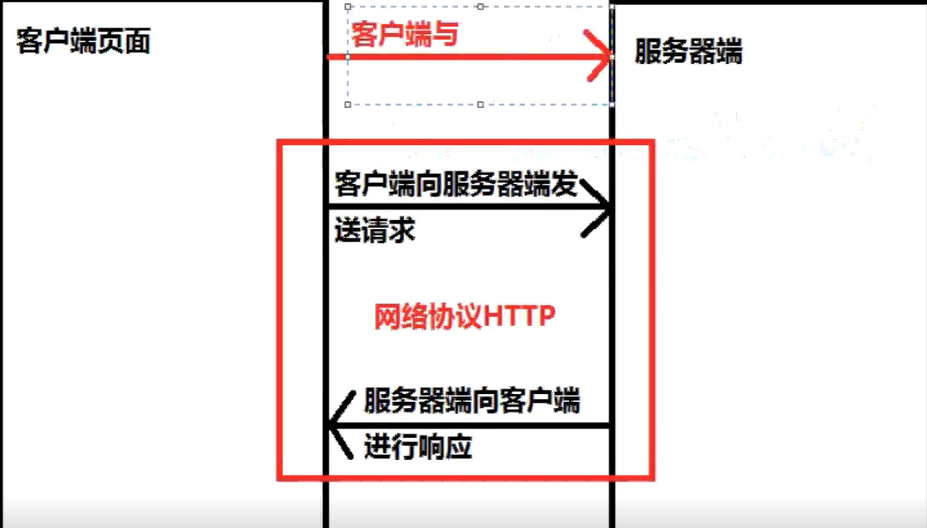




**- Web Socket（了解）**







**扩展**

**<!-- 嵌入视频 -->**

**<video id="player"**

**src="xxx.ogg"**

**poster="mymovie.jpg"**

**width="300" height="200">**

**Video player not available.**

**</video>**

**<!-- 嵌入音频 -->**

**<audio src="xxx.mp3" id="myAudio">Audio player not available.</audio>**

**1**

**2**

**这两个标签，都必须包含 src 属性，即要加载的媒体文件地址。width 和 height 属性是指定视频播放器的大小。poster 属性是在加载视频期间会显示的图像。位于开始和结束标签之间的内容是后备内容，即当浏览器不支持这两个标签时会显示这些内容。 因为不是所有的浏览器都支持所有的媒体格式，所以可以指定不同的媒体来源。这时会用到 `` 标签：**

**<!-- 嵌入视频 -->**

**<video id="player">**

**<source src="xx.webm" type="video/webm; codecs='vp8, vorbis'">**

**<source src="xx.ogv" type="video/ogg; codecs='theora, vorbis'">**

**Video player not available.**

**</video>**

**<!-- 嵌入音频 -->**

**<audio id="myAudio">**

**<source src="xx.ogg" type="audio/ogg">**

**<source src="xx.mp3" type="audio/mpeg">**

**Audio player not available.</audio>**

**记录几个给网站添加背景音乐的HTML5代码，稍微总结一下**

**方式一：<video controls="" autoplay="" name="media"><source src="音乐" type="audio/mpeg"></video>**

**这种方式会显示播放器。**

**方式二：<embed src="music/We Don't Talk Anymore.mp3" autostart="true" loop="true" hidden="true"></embed>**

**这个方式主要是支持的播放器多。一定程度上解决了浏览器兼容的问题。**

**方式三：<audio autoplay="" loop=""><source src="http://mi.0w0.im/Letter\_Song.mp3"></audio>**

**这个不显示播放器。**

**方式四：<bgsound src=背景音乐链接地址 loop=-1>**

**方式五：<audio src="music/We Don't Talk Anymore.mp3" id="aud" autoplay="autoplay" controls="controls" preload="auto"></audio>**