**html5**

**移动端开发**

**移动端实现效果：两边固定，中间自适应**

**thin 移动端的一像素**

**优点 | 缺点**

**---|---**

**圣杯 | 结构简单，无多余 dom 层 | 中间部分宽度小于左侧时布局混乱**

**绝对定位 | 结构简单，且无需清理浮动 | 两侧高度无法支撑总高度**

**双飞翼 | 支持各种宽高变化，通用性强 | dom 结构多余层，增加渲染树生成的计算量**

**flex 弹性盒布局**

开一个弹性盒 父级设置 display:flex；

display:flex; -webKit-flex兼容写法  **设置一个弹性盒**  （后边的元素才会起效）

项目默认设置：

项目在一行显示

项目自动继承父盒子

float vertical-align:middle clear不起作用

**6个属性设置在父元素上**

**flex-direction:**

row column row-reverse column-reverse

**flex-wrap**

换行

**flex-flow**

包含（ flex-direction flex-wrap）

**justify-content**

flex-start 效果相当于float:left

flex-end 效果相当于float:right

space-between 两边无空格

space-around 两边空格为元素宽度的一半

space-evenly 水平平摊 左右有距离

注：flex-direction 决定主轴的方向（row横 column横） justify-content方向与主轴方向平行，align-items方向垂直于主轴，两个属性指向随主轴方向变化而变化；

**align-items**

//设置垂直(侧轴)方向的对齐方式

flex-start 交叉轴起点对齐。

flex-end 交叉轴终点对齐。

center 交叉轴中点对齐。

baseline 项目的第一行文字的基线对齐。

stretch (默认值)：如果项目未设置高度或设为auto，将占满整个容器的高度

**align-content**

//设置多轴对齐的方式 (如果只有一根轴线，一行，这个属性不起作用)

flex-start

flex-end

center

space-between

stretch

space-around

**以下6个属性设置在项目(子元素)上**

**order 排序**

**flex**

**flex-grow:1**

**flex-shrink:0**

**flex-basis 默认的**

**align-self 单独**

start end center baseline stretch

子元素/项目：

flex-grow:1 撑开剩余的空间

order: 弹性子盒子顺序

**设备检测**

判断是苹果还是安卓

window.navigator.appVersion

**手机像素**

**逻辑像素**

在ps中量取的设计稿的尺寸 最小是1px

**物理像素**

手机出厂已设置

苹果5分辨率640 物理像素320

**设备像素比** （dpr） devicePixelRatio

出厂已设置 window.devicePixelRatio 获取

（逻辑像素/物理像素 的值）

现市场大多用5寸屏 设计稿所谓750 设备髋375 viewport：375px

**手机视口**

**布局视口** （layout viewport） 大多为980

document.documentElement.clientWidth

**视觉视口**

<meta name=’viewport’>

**理想视口**

完美适配设备的 viewport

**响应式布局** responsive

@media screen(all) and (max-width:414px){ //and前后有空格

div{

background:#fcc;

}

}

rem em 相对单位

计算相对单位rem px

媒体查询 vh，vw js sass，Flexible.js

Flexible.js 解决 初始缩放比例/动态计算根元素字体大小

**meta标签及属性**

<metaname="viewport"content="width=device-width,initial-scale=1,minimum-scale=1,maximum-scale=1,user-scalable=no">

①viewport获取视口宽度

②device-width设备宽度

③initial-scale初始化缩放比例

④minimum-scale最小缩放比例

⑤maximum-scale最大缩放比例

⑥user-scalable=no 禁止用户缩放 默认值是yes开启状态

**移动端1px解决方案**

**ios设备**

1. 缩放

设置了 border:solid 1px #000;

加设 transform:scale(0.5)

1. 在meta标签上设置 meta:inital-scale=0.5

**缩放设置格式：**

div{

width：200px;

height:200px;

background:red;

transform:scale(0.5);

transform-origin:top left 设置缩放后的位置

}

**android设备**

·border-image

·渐变

·阴影

·伪元素

**字体图标 icon-font**

@font-face 定义服务器端字体

字体的四种格式：ttf woff eot svg

rem

pc端字体12px相当于移动端的40px；

rem是相当于html/根元素的字体大小

计算rem

1. @media

2. js

**动画**

**transform**：{

**旋转**

rotate(135deg) 旋转135度

rotateX(135deg) 以x轴为轴心旋转135度

rotateY(135deg) 以Y轴为轴心旋转135度

rotateZ

rotate3d(X,Y,Z,45deg)

**平移**

translateX(100px) 沿着x轴向右平移100px 有3d

**缩放**

scale(1) 有3d

**斜切**

skew 没有3d

matrix() 方法把所有 2D 转换方法组合在一起。

}

C3  **<http://www.w3school.com.cn/css3/css3_2dtransform.asp>**

## **2D Transform 方法**

|  |  |
| --- | --- |
| **函数** | **描述** |
| matrix(*n*,*n*,*n*,*n*,*n*,*n*) | 定义 2D 转换，使用六个值的矩阵。 |
| translate(*x*,*y*) | 定义 2D 转换，沿着 X 和 Y 轴移动元素。 |
| translateX(*n*) | 定义 2D 转换，沿着 X 轴移动元素。 |
| translateY(*n*) | 定义 2D 转换，沿着 Y 轴移动元素。 |
| scale(*x*,*y*) | 定义 2D 缩放转换，改变元素的宽度和高度。 |
| scaleX(*n*) | 定义 2D 缩放转换，改变元素的宽度。 |
| scaleY(*n*) | 定义 2D 缩放转换，改变元素的高度。 |
| rotate(*angle*) | 定义 2D 旋转，在参数中规定角度。 旋转 |
| skew(*x-angle*,*y-angle*) | 定义 2D 倾斜转换，沿着 X 和 Y 轴。 没有3D |
| skewX(*angle*) | 定义 2D 倾斜转换，沿着 X 轴。 |
| skewY(*angle*) | 定义 2D 倾斜转换，沿着 Y 轴。 |

**transition 过渡动画**

**需要事件触发**

**起作用的情况**

**同一属性 值发生变化**

**不支持的属性 display z-index**

**属性：**

**transition：all 5s 3s； 过渡 默认all所有属性 过渡动画效果**

执行时间5s 延时时间3s

**color 5s; 只是字体颜色变化**

**transition-duration 执行时间**

**transition-timing-function 动画执行速度 ease-out 动画越来越快**

**transition-delay 延时执行**

**backface-visibility:hidden 设置使转到背面时看不到正面的字**

**transform-style:preserve-3d; 以3d形式展示 默认2d：flat**

**transform-origin：top 位置 可以写% px**

|  |  |  |
| --- | --- | --- |
| **属性** | **描述** | **CSS** |
| [transition](http://www.w3school.com.cn/cssref/pr_transition.asp" \o "CSS3 transition 属性) | 简写属性，用于在一个属性中设置四个过渡属性。 | 3 |
| [transition-property](http://www.w3school.com.cn/cssref/pr_transition-property.asp" \o "CSS3 transition-property 属性) | 规定应用**过渡**的 CSS 属性的名称。 | 3 |
| [transition-duration](http://www.w3school.com.cn/cssref/pr_transition-duration.asp" \o "CSS3 transition-duration 属性) | 定义过渡效果花费的时间。默认是 0。 | 3 |
| [transition-timing-function](http://www.w3school.com.cn/cssref/pr_transition-timing-function.asp" \o "CSS3 transition-timing-function 属性) | 规定过渡效果的时间曲线。默认是 "ease"。 | 3 |
| [transition-delay](http://www.w3school.com.cn/cssref/pr_transition-delay.asp" \o "CSS3 transition-delay 属性) | 规定过渡效果何时开始。默认是 0。 | 3 |

## **CSS3 动画属性**

**animation**

下面的表格列出了 **@keyframes** 规则和所有动画属性：

|  |  |  |
| --- | --- | --- |
| **属性** | **描述** | **CSS** |
| [@keyframes](http://www.w3school.com.cn/cssref/pr_keyframes.asp" \o "CSS3 @keyframes 规则) | 规定动画。 | 3 |
| [animation](http://www.w3school.com.cn/cssref/pr_animation.asp" \o "CSS3 animation 属性) | 所有动画属性的简写属性，除了 animation-play-state 属性。 | 3 |
| [animation-name](http://www.w3school.com.cn/cssref/pr_animation-name.asp" \o "CSS3 animation-name 属性) | 规定 @keyframes 动画的名称。 | 3 |
| **[animation-duration](http://www.w3school.com.cn/cssref/pr_animation-duration.asp" \o "CSS3 animation-duration 属性)** | 规定动画完成一个周期所花费的秒或毫秒。默认是 0。 动画持续多久 | 3 |
| [animation-timing-function](http://www.w3school.com.cn/cssref/pr_animation-timing-function.asp" \o "CSS3 animation-timing-function 属性) | 规定动画的速度曲线。默认是 "ease" 慢速开始 慢速结束 中间快速。 | 3 |
| [animation-delay](http://www.w3school.com.cn/cssref/pr_animation-delay.asp" \o "CSS3 animation-delay 属性) | 规定动画何时开始。默认是 0。 **延时** | 3 |
| [animation-iteration-count](http://www.w3school.com.cn/cssref/pr_animation-iteration-count.asp" \o "CSS3 animation-iteration-count 属性) | 规定**动画被播放的次数**。默认是 1。  **infinite 无限播放** | 3 |
| **[animation-direction](http://www.w3school.com.cn/cssref/pr_animation-direction.asp" \o "CSS3 animation-direction 属性)** | 规定动画是否在下一周期逆向地播放。默认是 "normal"。 | 3 |
| [animation-play-state](http://www.w3school.com.cn/cssref/pr_animation-play-state.asp" \o "CSS3 animation-play-state 属性) | 规定动画是否正在运行或暂停。默认是 "running"。 **paused(暂停)** | 3 |
| [animation-fill-mode](http://www.w3school.com.cn/cssref/pr_animation-fill-mode.asp" \o "CSS3 animation-fill-mode 属性) | 规定对象动画时间之外的状态。 **forwards留住最后一帧的动画**  **backwards(返回第一帧)** | 3 |

**animation：infinite 无限循环**

**sass中for循环**

**@for $i from 1 to 6{ 不包括下标为6的元素**

**&:nth-of-type(#{$i+1}){**

**}**

**}**

...

box-shadow:2px 0,6px 0,10px 0;## animation 动画

```

**@keyframes 关键帧**

@keyframes 名字{

0%（from）{

}

100%(to){

}

}

```

- **animation** :**名字 动画执行时长 动画的运动曲线 延时时间 动画的播放次数 播放方向**

**-animation-name 名字**

**-animation-duration 动画持续多久**

**-animation-delay 指定动画何时开始的属性**

- **animation-iteration-count:infinite(无限循环) 播放次数**

**- animation-direction 播放方向 默认 normal/ altrenate(反向播放)**

**- animation-play-state 设置当前动画正在运行还是暂停 running(正在运行) paused(暂停)**

**- animation-fill-mode 动画执行完成之后 定哪一帧 none (默认)**

**forwards保留最后一帧**

**backwards(第一帧) both**

**-[animation-timing-function](http://www.w3school.com.cn/cssref/pr_animation-timing-function.asp" \o "CSS3 animation-timing-function 属性) 动画的速度曲线 linear匀速动画**

### 动画里面的一种 - 帧动画

- steps （n(自然数),start | end）

- steps-strat 等于 steds(1,strat)

- steps-end 等于 steds(1,end)

- step-start等同于steps(1,start)，动画分成1步，动画执行时为开始左侧端点的部分为开始；

- step-end等同于steps(1,end)：动画分成一步，动画执行时以结尾端点为开始，默认值为end。

**移动端事件**

**touch**

**Dom2级绑定**

**事件源 事件对象TouchEvent**

**手指信息 都存储在三个对象中 touches targetTouches changedTouches**

手指按下 touchstart （mousedown）

手指抬起 touchend （mousemove）

手指移动 touchmove （mouseup）

取消/中断 touchcancel

TouchList {0: Touch, length: 1}

length:1

0:Touch

clientX:65 // 触摸点在浏览器窗口中的横坐标

clientY:18 // 触摸点在浏览器窗口中的纵坐标

force:1 // 触摸点压力大小

identifier:0 // 触摸点唯一标识（ID）

pageX:65 // 触摸点在页面中的横坐标

pageY:18 // 触摸点在页面中的纵坐标

radiusX:11.5 // 触摸点椭圆的水平半径

radiusY:11.5 // 触摸点椭圆的垂直半径

rotationAngle:0 // 旋转角度

screenX:560 // 触摸点在屏幕中的横坐标

screenY:175 // 触摸点在屏幕中的纵坐标

target:div#touchLog 触摸目标

\_\_proto\_\_:Touch

\_\_proto\_\_:TouchList

1.**Touch.identifier：此 Touch 对象的唯一标识符。** 一次触摸动作(我们指的是手指的触摸)在平面上移动的整个过程中，该标识符不变。 可以根据它来判断跟踪的是否是同一次触摸过程，此值为只读属性。

2.Touch.screenX：触点相对于屏幕左边沿的X坐标。只读属性。

3.Touch.screenY：触点相对于屏幕上边沿的Y坐标。只读属性。

4.Touch.clientX：触点相对于可见视区(visual viewport)左边沿的X坐标。不包括任何滚动偏移。只读属性。

5.Touch.clientY：触点相对于可见视区(visual viewport)上边沿的Y坐标。不包括任何滚动偏移。只读属性。

6.Touch.pageX：触点相对于HTML文档左边沿的X坐标。当存在水平滚动的偏移时，这个值包含了水平滚动的偏移。只读属性。

7.Touch.pageY：触点相对于HTML文档上边沿的Y坐标。当存在水平滚动的偏移时，这个值包含了垂直滚动的偏移。只读属性。

8.Touch.radiusX：能够包围用户和触摸平面的接触面的最小椭圆的水平轴(X轴)半径。这个值的单位和 screenX 相同。只读属性。

9.Touch.radiusY：能够包围用户和触摸平面的接触面的最小椭圆的垂直轴(Y轴)半径。这个值的单位和 screenY 相同。只读属性。

10.Touch.rotationAngle：它是这样一个角度值：由radiusX 和 radiusY描述的正方向的椭圆，需要通过顺时针旋转这个角度值，才能最精确地覆盖住用户和触摸平面的接触面。只读属性。

11.Touch.force：手指挤压触摸平面的压力大小，从0.0(没有压力)到1.0(最大压力)的浮点数。只读属性。

1. Touch.target：当这个触点最开始被跟踪时(在 touchstart 事件中)，触点位于的HTML元素。哪怕在触点移动过程中，触点的位置已经离开了这个元素的有效交互区域，或者这个元素已经被从文档中移除。需要注意的是，如果这个元素在触摸过程中被移除，这个事件仍然会指向它，但是不会再冒泡这个事件到 window 或 document 对象。因此，如果有元素在触摸过程中可能被移除，最佳实践是将触摸事件的监听器绑定到这个元素本身，防止元素被移除后，无法再从它的上一级元素上侦测到从该元素冒泡的事件。只读属性。

高级单例模式 ：对象 函数 闭包

**[localStorage 笔记](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/2E3AB96CF0F24E9194870807007ED7F0?method=download&read=true&shareKey=c88f3e58c27d2f0722bdb3295226d6c8" \l "h5-localstorage-%E7%AC%94%E8%AE%B0)**

* + [在客户端存储数据](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/2E3AB96CF0F24E9194870807007ED7F0?method=download&read=true&shareKey=c88f3e58c27d2f0722bdb3295226d6c8" \l "%E5%9C%A8%E5%AE%A2%E6%88%B7%E7%AB%AF%E5%AD%98%E5%82%A8%E6%95%B0%E6%8D%AE)
    - * **[cookie](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/2E3AB96CF0F24E9194870807007ED7F0?method=download&read=true&shareKey=c88f3e58c27d2f0722bdb3295226d6c8" \l "cookie)**
      * **[localStorage](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/2E3AB96CF0F24E9194870807007ED7F0?method=download&read=true&shareKey=c88f3e58c27d2f0722bdb3295226d6c8" \l "localstorage)**
      * **[sessionStorage](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/2E3AB96CF0F24E9194870807007ED7F0?method=download&read=true&shareKey=c88f3e58c27d2f0722bdb3295226d6c8" \l "sessionstorage)**
  + [localStorage基本语法](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/2E3AB96CF0F24E9194870807007ED7F0?method=download&read=true&shareKey=c88f3e58c27d2f0722bdb3295226d6c8" \l "localstorage%E5%9F%BA%E6%9C%AC%E8%AF%AD%E6%B3%95)
    - * **[localStorage本地方法](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/2E3AB96CF0F24E9194870807007ED7F0?method=download&read=true&shareKey=c88f3e58c27d2f0722bdb3295226d6c8" \l "localstorage%E6%9C%AC%E5%9C%B0%E6%96%B9%E6%B3%95)**
      * **[localStorage 的巧用](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/2E3AB96CF0F24E9194870807007ED7F0?method=download&read=true&shareKey=c88f3e58c27d2f0722bdb3295226d6c8" \l "localstorage-%E7%9A%84%E5%B7%A7%E7%94%A8)**
    - [自己扩展：真实项目中使用的第三方插件](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/2E3AB96CF0F24E9194870807007ED7F0?method=download&read=true&shareKey=c88f3e58c27d2f0722bdb3295226d6c8" \l "%E8%87%AA%E5%B7%B1%E6%89%A9%E5%B1%95%E7%9C%9F%E5%AE%9E%E9%A1%B9%E7%9B%AE%E4%B8%AD%E4%BD%BF%E7%94%A8%E7%9A%84%E7%AC%AC%E4%B8%89%E6%96%B9%E6%8F%92%E4%BB%B6)

## **在客户端存储数据**

#### ****cookie****

* 但是 cookie 不适合大量数据的存储，因为它们由每个对服务器的请求来传递，这使得 cookie 速度很慢而且效率也不高。

HTML5 提供了两种在客户端存储数据的新机制

#### ****localStorage****

* localStorage 存储的数据没有时间限制。只要浏览器不清楚数据，它**储存的数据一直存在**

#### ****sessionStorage****

* sessionStorage 方法针对一个 会话窗口进行数据存储。**当用户关闭浏览器窗口后，数据会被删除**。

## **localStorage基本语法**

HTML5 使用 JavaScript 来存储和访问数据。可以通过调用Window.sessionStorage和Window.localStorage属性创建一个Storage对象的实例。

* Storage 对象以简单的键值得形式来储存数据，键值都是以字符串的形式进行储存，如果一个值是数字，它将被转为字符串。

#### ****localStorage本地方法****

| **header 1** | **header 2** |
| --- | --- |
| 添加键值对 | localStorage.setItem(key,value); |
| 获取键值对 | localStorage.getItem(key); |
| 删除键值对 | localStorage.removeItem(key);// 删除数据数据时使用单个参数； |
| 清除所有键值对 | localStorage.clear();// 清空数据数据时不需要参数； |
| 获取localStorage的键名 | localStorage.key; |
| 获取localStorage中保存的键值对的数量 | localStorage.length; |

#### ****localStorage 的巧用****

* 对用户访问页面的次数进行计数：

<script type="text/javascript">

**if** (localStorage.pagecount){

localStorage.pagecount=Number(localStorage.pagecount) +1;

}**else**{

localStorage.pagecount=1;

}

</script>

用户每次刷新页面都会触发localStorage，触发localStorage后都会给 localStorage.pagecount的值增加增加一，以此来达到统计访问量的目的。

用户在当前 session 中访问页面的次数进行计数时，将上面代码中的localStorage换成sessionStorage即可。

### **自己扩展：真实项目中使用的第三方插件**

对于数据多维的储存，使用Storage插件会更加方便，比较常用有Storage.js、store.js等插件。

store.js 插件：

该插件小巧，简介，兼容性很强，可实现大多数日常的应用

右击

document.oncontextmenu=function(){}

Drag&Drop拖放.md

HTML5 中提供了直接拖放的 API，极大的方便我们实现拖放效果，不需要去写一大堆的 js，只需要通过监听元素的拖放事件就能实现各种拖放功能。 想要拖放某个元素，必须设置该元素的 **draggable 属性为 true，当该属性为 false 时，将不允许拖放**。而 img 元素和 a 元素都默认设置了 draggable 属性为 true，可直接拖放，如果不想拖放这两个元素，把属性设为 false 即可。

### **拖放事件**

* **被拖放的元素称为源对象**
* **经过的元素称为过程对象**
* **到达的元素称为目标对象**

##### **源对象**

* dragstart：源对象开始拖放。
* drag： 源对象拖放过程中。
* dragend：源对象拖放结束。

##### **过程对象**

* dragenter：源对象开始进入过程对象范围内
* dragover：源对象在过程对象范围内移动。
* dragleave：源对象离开过程对象的范围。

##### **目标对象**

* **drop**：源对象被释放时触发。 凡是在页面内都认为是过程对象，所以要阻止document的dragover的默认行为

### **dataTransfer 对象**

在所有拖放事件中提供了一个数据传递对象 dataTransfer，用于在源对象和目标对象间传递数据。 dataTransfer 对象不支持IE

##### **setData()**

该方法向 dataTransfer 对象中存入数据。接收两个参数，第一个表示要存入数据种类的字符串，现在支持有以下几种：

text/plain：文本文字。

text/html：HTML文字。

text/xml：XML文字。

text/uri-list：URL列表，每个URL为一行。

第二个参数为要存入的数据。

**event**.dataTransfer.setData('text/plain','Hello World');

##### **getData()**

该方法从 dataTransfer 对象中读取数据。参数为在 setData 中指定的数据种类。

**event**.dataTransfer.getData('text/plain');

##### **clearData()**

该方法清除 dataTransfer 对象中存放的数据。参数可选，为数据种类。若参数为空，则清空所有种类的数据

**event**.dataTransfer.clearData();

##### **setDragImage()**

该方法通过用img元素来设置拖放图标。接收三个参数，第一个为图标元素，第二个为图标元素离鼠标指针的X轴位移量，第三个为图标元素离鼠标指针的Y轴位移量。

##### **effectAllowed 和 dropEffect 属性**

这两个属性结合起来设置拖放的视觉效果。

DAY9-移动开发-day9-多媒体(视频和音频).md

* [audio](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/3E9F8B01F4DB42D39BB7A5EC2A44D43C?method=download&read=true&shareKey=dd7b4afd854ddb62c9ebbf999d1883f0" \l "audio)
  + [语法](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/3E9F8B01F4DB42D39BB7A5EC2A44D43C?method=download&read=true&shareKey=dd7b4afd854ddb62c9ebbf999d1883f0" \l "语法)
  + [音频的兼容性](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/3E9F8B01F4DB42D39BB7A5EC2A44D43C?method=download&read=true&shareKey=dd7b4afd854ddb62c9ebbf999d1883f0" \l "音频的兼容性)
  + [视频的兼容性](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/3E9F8B01F4DB42D39BB7A5EC2A44D43C?method=download&read=true&shareKey=dd7b4afd854ddb62c9ebbf999d1883f0" \l "视频的兼容性)
  + [属性](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/3E9F8B01F4DB42D39BB7A5EC2A44D43C?method=download&read=true&shareKey=dd7b4afd854ddb62c9ebbf999d1883f0" \l "属性)
  + [方法](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/3E9F8B01F4DB42D39BB7A5EC2A44D43C?method=download&read=true&shareKey=dd7b4afd854ddb62c9ebbf999d1883f0" \l "方法)
  + [事件](https://note.youdao.com/md/preview.html?file=/yws/api/personal/file/3E9F8B01F4DB42D39BB7A5EC2A44D43C?method=download&read=true&shareKey=dd7b4afd854ddb62c9ebbf999d1883f0" \l "事件)

# **audio**

该标签往页面插入音频标签

## **语法**

<audio src="song.mp3" controls="controls" loop="loop" autoplay="autoplay">亲 您的浏览器不支持html5的audio标签</audio>

拥有两份源文件的音频播放器

<audio controls>

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

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

Your browser does not support the audio element.

</audio>

## **音频的兼容性**

| **浏览器** | **版本** | **支持格式** |
| --- | --- | --- |
| Internet Explorer | 9.0+ | MP3, AAC |
| Chrome | 6.0+ | Ogg Vorbis, MP3, WAV（9.0+） |
| Firefox | 3.6+ | Ogg Vorbis, WAV |
| Safari | 5.0+ | MP3, AAC, WAV |
| Opera | 10.0+ | Ogg Vorbis, WAV |

大约有80%的浏览器支持HTML5的audio标签，但是并没有一种统一的音频格式。从支持的格式来看，要让所有的浏览器可以播放audio元素上的音频，最佳的方式是提供MP3和Ogg两种格式，兼容代码如下：

<audio controls>

<source src="elvis.mp3" type='audio/mpeg; codecs="mp3"'>

<source src="elvis.oga" type='audio/ogg; codecs="vorbis"'>

*<!-- 向后兼容代码：如，显示提示信息、提供下载链接使用flash播放器等 -->*

浏览器不支持<code>audio</code>标签

</audio>

## **视频的兼容性**

| **浏览器** | **版本** | **支持格式** |
| --- | --- | --- |
| Internet Explorer | 9.0+ | MP4 |
| Chrome | 6.0+ | MP4,WebM,Ogg |
| Firefox | 3.6+ | WebM,Ogg |
| Safari | 5.0+ | MP4 |
| Opera | 10.0+ | WebM,Ogg |

从浏览器支持的视频格式来看，最佳的方式是提供WebM和MP4两种格式的视频。兼容代码如下：

<**video** controls>

<source src=**video**.webm type=video/webm>

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

<!—向后兼容代码: -->

<**iframe** width="480" height="360" src="http://www.youtube.com/embed/xzMUyqmaqcw?rel=0" frameborder="0" allowfullscreen></iframe>

</video>

## **属性**

| **名称** | **说明** |
| --- | --- |
| src | 歌曲的路径 |
| controls | 设置或返回音频/视频是否显示控件（比如播放/暂停等） |
| loop | 设置或返回音频/视频是否应在结束时重新播放 |
| autoplay | 当歌曲加载后自动播放，**但是只有pc端可以实现,移动端不行** |
| currentTime | 指示音频/视频播放的当前位置（以秒计） |
| playbackRate | 设置或返回音频/视频播放的速度 |
| duration | 返回当前音频/视频的长度（以秒计） |
| ended | 返回音频/视频的播放是否已结束 |
| **muted** | **设置或返回音频/视频是否静音** |
| **paused** | **设置或返回音频/视频是否暂停** |
| **volume** | **设置或返回音频/视频的音量** |

## **方法**

| **名称** | **说明** |
| --- | --- |
| load() | 重新加载音频/视频元素 |
| play() | 开始播放音频/视频 |
| pause() | 暂停当前播放的音频/视频 |

## **事件**

| **名称** | **说明** |
| --- | --- |
| canplaythrough | 当浏览器可在不因缓冲而停顿的情况下进行播放时 |
| ended | 当目前的播放列表已结束时 |
| timeupdate | 当目前的播放位置已更改时 |

**canvas**

**canvas是HTML5中新出的一个元素，开发者可以通过JS脚本语言动态绘制图片**

**canvas标签用来绘制图像，本身就是一个画布，没有绘画能力，所有的绘制都是通过js脚本实现的**

**默认大小**

**Canvas 的默认大小为300像素×150像素（宽×高，像素的单位是px）。不能通过行内样式来修改宽高，它有自己的宽高属性。**

**渲染上下文**

**getContext()只有一个参数，上下文的格式**

**var canvas = document.getElementById('tutorial');**

**var ctx = canvas.getContext('2d');**

### **绘制线条**

#### **步骤**

* **开启路径**

beginPath()

* **设置起始点**

moveTo(x,y);

* **设置'结束点'**

lineTo(x,y);

* **描边/填充**

stroke()/fill()

* **关闭路径**

closePath()

### **绘制矩形的方法**

**rect(x,y,width,height)**

需要配合 stroke() 或 fill() 方法在画布上实际地绘制矩形

fillRect(x, y, width, height)

绘制一个填充的矩形

strokeRect(x, y, width, height)

绘制一个矩形的边框

clearRect(x, y, width, height)

清除指定矩形区域，让清除部分完全透明

### **圆弧**

**arc(x,y,r,sAngle,eAngle,counterclockwise);**

**PI=π**

**deg=Math.PI/180;**

x,y 是圆心点的坐标

r是圆的半径

sAngle、eAngle是圆的起始角度和结束角度。以弧度计

counterclockwise可选。规定应该逆时针还是顺时针绘图。False = 顺时针，true = 逆时针

### **样式**

strokeStyle

描边的颜色

fillStyle

填充的颜

### **阴影**

shadowColor

阴影的颜色

shadowBlur

阴影的模糊级别

shadowOffsetX

阴影的水平距离

shadowOffsetY

阴影的垂直距离

### **线条的样式**

**lineCap**

设置或返回线条的结束端点样式

* butt 默认。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。
* square 向线条的每个末端添加正方形线帽

lineWidth

当前线条的宽度，以像素计。

### **旋转**

rotate(angle);

旋转角度，以弧度计。

**角度转弧度公式**

**deg\*Math.PI/180**

如需旋转 5 度，可规定下面的公式：5\*Math.PI/180。

### **重新映射画布（0,0）**

**translate(x,y);**

### 绘制文本方法

fillText(text,x,y,maxWidth)

在画布上绘制“被填充的”文本

text 规定在画布上输出的文本。x：开始绘制文本的x坐标位置（相对于画布）

y:开始绘制文本的Y坐标位置（相对于画布。

maxWidth 可选。允许的最大文本宽度，以像素计。

strokeText()

在画布上绘制文本（无填充）

同上

### 绘制文本的属性

textAlign

start 默认。文本在指定的位置开始。end 文本在指定的位置结束。

center 文本的中心被放置在指定的位置。left 文本左对齐。right 文本右对齐。

textBaseline

alphabetic默认。文本基线是普通的字母基线top 文本基线是 em 方框的顶端。。

hanging 文本基线是悬挂基线。

middle 文本基线是 em 方框的正中。

ideographic 文本基线是表意基线。bottom 文本基线是 em 方框的底端。

font 属性设置或返回画布上文本内容的当前字体属性

| **值** | **说明** |
| --- | --- |
| **font-style** | **规定字体样式。可能的值：normal、italic、oblique** |
| **font-variant** | **规定字体变体。可能的值：normal、small-caps** |
| **font-weight** | **规定字体的粗细。可能的值normal、bold、600等** |
| **font-size / line-heigh** | **规定字号和行高，以像素计** |
| **font-family** | **规定字体系列** |

### 渐变

createLinearGradient(x0,y0,x1,y1); 线性渐变

x0 渐变开始点的x坐标

y0 渐变开始点的y坐标x1 渐变结束点的 x 坐标

y1 渐变结束点的 y 坐标

createRadialGradient(x0,y0,r0,x1,y1,r1);放射性渐变

x0 渐变的开始圆的 x 坐标

y0 渐变的开始圆的 y 坐标

r0 开始圆的半径x1 渐变的结束圆的 x 坐标

y1 渐变的结束圆的 y 坐标

r1 结束圆的半径

addColorStop(stop,color);

stop 介于 0.0 与 1.0 之间的值，表示渐变中开始与结束之间的位置。color 在结束位置显示的 CSS 颜色值

var grd=ctx.createLinearGradient(0,0,170,0);

grd.addColorStop(0,"black");

grd.addColorStop("0.3","magenta");

grd.addColorStop("0.5","blue");

grd.addColorStop("0.6","green");

grd.addColorStop("0.8","yellow");

grd.addColorStop(1,"red");

ctx.fillStyle=grd;

ctx.fillRect(20,20,150,100);

### save()

保存当前环境的状态

### restore()

返回之前保存过的路径状态和属性

### 绘制图像

drawImage(img,sx,sy,swidth,sheight,x,y,width,height);

| **参数** | **描述** |
| --- | --- |
| img | 规定要使用的图像、画布或视频 |
| sx | 可选。开始剪切的 x 坐标位置 |
| sy | 可选。开始剪切的 y 坐标位置 |
| swidth | 可选。被剪切图像的宽度 |
| sheight | 可选。被剪切图像的高度 |
| x | 在画布上放置图像的 x 坐标位置 |
| y | 在画布上放置图像的 y 坐标位置 |
| width | 可选。要使用的图像的宽度。（伸展或缩小图像） |
| height | 可选。要使用的图像的高度。（伸展或缩小图像） |

**求和**

var arr=[{val:1},{val:2}];

arr.reduce(function(prev,next){

prev+=next.val;

return prev

},0)

requestAnimationFrame 定时器

globalcompositeOperation source-over destination-out

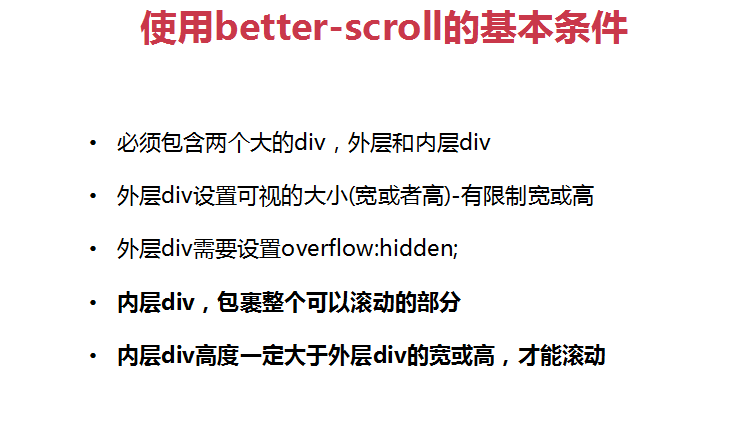
canvas.toDataURL 把canvas中的图片保存为base64编码

**better-scroll** 是什么

<http://ustbhuangyi.github.io/better-scroll/doc/zh-hans/#%E8%B5%B7%E6%AD%A5>

**better-scroll 是一款重点解决移动端（已支持 PC）各种滚动场景需求的插件。**它的**核心是借鉴的 iscroll 的实现**，它的 API 设计基本兼容 iscroll，在 iscroll 的基础上又扩展了一些 feature 以及做了一些性能优化。 **基于iscroll扩展的插件**

**better-scroll 是基于原生 JS 实现的，不依赖任何框架。**它编译后的代码大小是 63kb，压缩后是 35kb，gzip 后仅有 9kb，是一款非常轻量的 JS lib。



## **probeType**

* 类型：Number
* 默认值：0
* 可选值：1、2、3
* 作用：有时候我们需要知道滚动的位置。当 probeType 为 1 的时候，会非实时（屏幕滑动超过一定时间后）派发**[scroll 事件](http://ustbhuangyi.github.io/better-scroll/doc/zh-hans/events.html" \l "scroll)**；当 probeType 为 2 的时候，会在屏幕滑动的过程中实时的派发 scroll 事件；当 probeType 为 3 的时候，不仅在屏幕滑动的过程中，而且在 momentum 滚动动画运行过程中实时派发 scroll 事件。如果没有设置该值，其默认值为 0，即不派发 scroll 事件。

**0 不触发 scroll**

**1 触发 一定时间后**

**2 触发 实时触发**

**3 滑动/图滚动 触发**

**Swiper**

**开源 免费强大的触摸滑动插件**