杰普软件科技有限公司

www.briup.com

Tel: (021)55660810
Fax: (021)55660802
Email:training@briup.com
Msn: training.sh@hotmail.com
Home:http://www.briup.com

地址：上海市闸北区万荣路1188弄F栋6号三层-上海服务外包科技园龙软园区
邮编：200436

电话：021-56657112
传真：021-55661523-8003
电邮：training@briup.com
主页：http://www.briup.com

Briup High-End IT Training

HTML5 API

Brighten Your Way And Raise You Up.

学习目标

- 掌握媒体元素API
- 熟练掌握Canvas绘图API
- 掌握拖放的API
- Web Workers

杰普软件科技有限公司
www.briup.com

Tel: (021)55660810
Fax: (021)55660802
Email:training@briup.com
Msn: training.sh@hotmail.com
Home:http://www.briup.com

地址：上海市闸北区万荣路1188弄F栋6号三层-上海服务外包科技园龙软园区
邮编：200436

电话：021-56657112
传真：021-55661523-8003
电邮：training@briup.com
主页：http://www.briup.com

Briup High-End IT Training

HTML5 API

第 1 章 媒体元素API

Brighten Your Way And Raise You Up.

学习目标

- 掌握媒体元素标签和属性
- 掌握自定义媒体播放器
- 掌握媒体元素标签共有方法
- 熟悉媒体元素标签共有事件

媒体元素标签audio和video

◆ 媒体元素

在HTML5问世之前，要在网络上展示视频，音频，动画，除了使用第三方自主开发的播放器之外，使用得最多的工具就是Flash,但是需要在浏览器上安装各种插件，并且有时速度很慢。HTML5新增了两个与媒体相关的标签，让开发人员不必依赖任何插件就能在网页中嵌入跨浏览器的音频和视频内容，这两个标签是 <audio> <video>。

◆ 在页面中嵌入

```
<!--通过src指定一个视频地址-->
<video src="resource/demo.mp4" type="video/mp4" controls autoplay>
</video>

<!--通过source元素为同一个媒体数据指定多个播放格式与编码方式，确保浏览
    器可以从中选择一种自己支持的播放格式-->
 <video autoplay controls>
    <source src="resource/demo_1.mp4" type="video/mp4">
    <source src="resource/demo_2.mp4" type="video/mp4">
</video>
```

视频格式

各大浏览器对不同视频格式的支持程度

浏览器	MP4	WebM	Ogg
IE	yes	yes	no
Chrome	yes	yes	yes
Firefox	yes	yes	yes
Safari	yes	no	no
Opera	yes	yes	yes

基本属性

- **width** 视频的宽度，以像素为单位
- **height** 视频的高度，以像素为单位
- **src** 视频的地址
- **controls** 控制条
- **autoplay** 设置自动播放
- **poster** 当视频不可用时，向用户展示一副替代用的图片
- **loop** 是否循环
- **volume** 音量，取值为0~1
- **muted** 是否处于静音状态
- **type** 媒体类型 MIME mp4- video/mp4
- **defaultPlaybackRate** 默认播放速率
- **playbackRate** 当前播放速率 >1快放 <1慢放
- **currentTime/duration** 当前播放位置、总的播放时间

只读属性

- **currentSrc** 读取播放中的媒体数据的URL地址
- **readyState** 准备状态

  0 没有获取到媒体的任何信息

  1 获取媒体数据无效，无法播放

  2 当前帧已获得，但还未获取到下一帧数据

  3 当前帧已获得，也获取到下一帧数据

  4 当前帧已获得，也获取到了让播放器前进的数据

- **played、paused、ended**

  已经播放的时间段、是否暂停、是否播放完毕

共有方法

- **play()** 播放媒体，自动将元素的paused属性的值变为false
- **pause()** 暂停播放，自动将元素的paused属性变为true
- **load()** 重新载入媒体进行播放，自动将元素的playbackRate属性的值变为defaultPlaybackRate属性的值，自动将元素的error值变为null
- **canPlayType()** 用来测试浏览器是否支持指定的媒体类型

事件

- loadstart　浏览器开始在网上寻找媒体数据
- progress　浏览器正在获取媒体数据
- suspend　浏览器暂停获取媒体数据
- abort　浏览器在下载完全部媒体数据之前中止获取媒体数据，非错误引起
- error　获取媒体数据过程中出错
- emptied　所在网络变未初始化状态
- stalled　浏览器舱室获取媒体数据失败
- play　即将开始播放
- paused　播放暂停
- loadedmetadata　浏览器已经获取完毕媒体时间长和字节数
- loadeddata　浏览器已加载完毕当前播放位置的媒体数据，准备播放

事件

- **waiting** 播放过程中由于得不到下一帧而停止播放
- **playing** 正在播放
- **canplay** 正在播放，并且以当前播放速率能够将媒体播放完毕，需缓存
- **canplaythrough** 浏览器可以播放媒体，并且以当前播放速率能够将媒体播放完毕，无需缓存
- **seeking** 请求数据，**seeking**属性为**true**
- **seeked** 停止请求数据，**seeking**属性为**false**
- **timeupdate** 当前播放位置改变
- **ended** 播放结束
- **ratechange** 播放速率改变
- **durationchange** 播放时长改变
- **volumechange** 音量改变

杰普软件科技有限公司
www.briup.com

Tel: (021)55660810
Fax: (021)55660802
Email:training@briup.com
Msn: training.sh@hotmail.com
Home:http://www.briup.com

地址：上海市闸北区万荣路1188弄F栋6号三层-上海服务外包科技园龙软园区
邮编：200436

电话：021-56657112
传真：021-55661523-8003
电邮：training@briup.com
主页：http://www.briup.com

Briup High-End IT Training

HTML5 API

第 2 章 画布API

Brighten Your Way And Raise You Up.

学习目标

- 掌握canvas元素的基本概念，并在页面上放置一个canvas元素，绘制出一个简单矩形
- 掌握使用路径绘制圆形与多边形
- 掌握渐变图形的绘制
- 掌握在画布中使用图像
- 掌握在画布中绘制文字，给文字加边框
- 掌握保持及恢复绘图状态的方法，保存绘制出来的图形
- 掌握在画布中制作简单动画

Canvas简介

canvas是H5中最受欢迎的功能，这个元素负责在页面中设定一个区域，然后就可以通过JS动态地在这个区域中绘制图形，canvas元素最早由苹果公司推出，当时主要用在Dashboard微件中，很快H5加入了这个元素，主要浏览器也迅速开始支持它。

2D上下文简介

画布本身是一个标签，没有绘图功能，有绘图功能的是图形上下文，图形上下文是一个封装了很多绘图功能的对象。

当getContext()方法参数为“2d”的时候，获取到2D上下文。使用2D上下文提供的方法，可以绘制简单的2D图形，比如矩形，弧线和路径。2D上下文的坐标开始于canvas元素的左上角，原点坐标是(0,0)。

绘制图形步骤

1. 取得canvas对象
2. 取得2d上下文(context)
3. 设定绘图样式

   fillStyle　　填充的样式

   strokeStyle 图形边框的样式
4. 指定线宽

   使用图形上下文对象lineWidth
5. 绘制矩形(填充与绘制边框)

   fillRect　　填充矩形

   strokeRect　　绘制矩形边框

```
// 1.获取canvas元素
var one = document.getElementById("one");
// 2.获取2D上下文
var context = one.getContext("2d");
// 3.设置绘图样式
context.fillStyle = "red";
context.strokeStyle = "blue";
// 4.指定线宽
context.lineWidth = 5;
// 5.绘制矩形
context.fillRect(0,0,100,100);
context.strokeRect(110,0,200,100);
```

绘制矩形

绘制矩形

矩形是唯一一种可以直接在2d上下文中绘制的形状

fillRect(x,y,width,height); 绘制矩形

strokeRect(x,y,width,height); 绘制矩形框

clearRect(x,y,width,height); 清除画布中的矩形框

这三个方法都能接受4个参数，矩形的x坐标，y坐标，宽度，高度，这些参数的单位都是像素

绘制路径-圆形

1. 开始创建路径

```
context.beginPath();
```

2. 设置路径（圆形）

```
context.arc(x,y,radius,startAngle,endAngle,counterclockwise);
```

参数分别为：圆心x坐标，圆心y坐标，半径，开始角度，结束角度，绘制方向(true为逆时针，false为顺时针)

3. 关闭路径

```
context.closePath();
```

4. 设定绘制样式，进行图形绘制

```
context.fillStyle = “rgba(225,0,0,0.25)”;
context.fill();  //填充图形
```

绘制路径-直线

- **moveTo(x,y)**

  将光标移动到指定坐标点，绘制直线的时候以这个坐标点为起点

  参数：起始点x坐标，起始点y坐标

- **lineTo(x,y)**

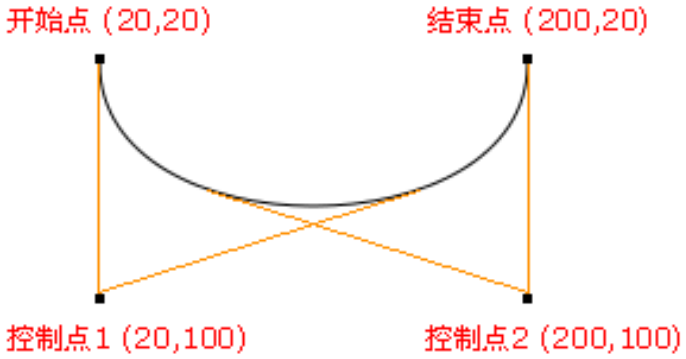
  表示直线的终点。

  参数：结束点x坐标，结束点y坐标

绘制路径-曲线

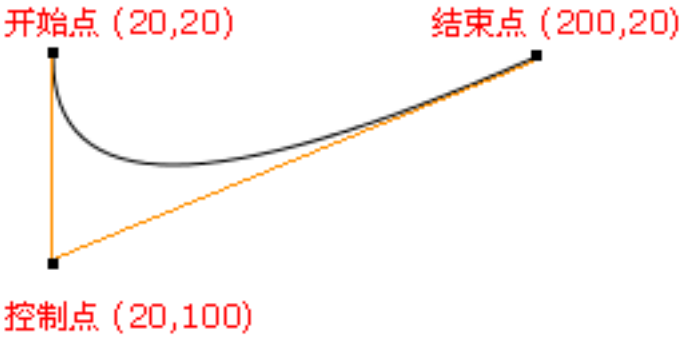
- **bezierCurveTo(c1x,c1y,c2x,c2y,x,y)**

从上一点开始绘制一条曲线，到(x,y)为止，并且以(c1x, c1y)和(c2x, c2y)为控制点，曲线会圆滑的靠近这两个控制点。



- **quadraticCurveTo(cx,cy,x,y)**

从上一点开始绘制一条**二次贝塞尔曲线**，到(x,y)为止，并且以(cx,cy作为控制点)



绘制渐变

◆ 线性渐变

绘制线性渐变时需要使用LinearGradient对象，通过2D上下文来创建和修改

1. 创建渐变对象

   var gradient = context.createLinearGradient(xstart,ystart,xend,yend)

   参数：起始点x坐标，起始点y坐标，结束点x坐标，结束点y坐标

2. 添加渐变颜色

   gradient.addColorStop(offset,color);

   参数：0-1之间的偏移量，颜色值

   由于是渐变，所以至少需要使用两次addColorStop方法以追加两个颜色(开始颜色和结束颜色)

3. 设置填充渐变

   context.fillStyle = gradient;

4. 使用渐变绘制矩形

context.fillRect(x,y,width,height);

径性渐变

- 径向渐变是指沿着圆形的半径方向向外进行扩展的渐变方式

  context.createRadialGradient(xstart,ystart,radiusStart,xend,yend,radiusend) 参数：起始圆的圆心x坐标，y坐标，半径，结束圆的圆心x坐标，y坐标，半径

  如果想从某个形状的中心点开始创建一个向外扩散的径向渐变的效果，就要将两个圆定义为同心圆

绘制变形

绘制图形的时候，我们可能经常会想要旋转图形，或者对图形进行变形处理，使用Canvas API的坐标轴变换处理功能，可以实现这种效果。

◆平移

context.translate(x,y)

参数：坐标原点在x轴方向移动的单位，坐标原点在y轴方向移动的单位

◆扩大

context.scale(x,y)

参数：水平方向的放大倍数，垂直方向放大的倍数

如果是缩小，将这两个参数设为0到1之间的数就可以

◆旋转

rotate(angle)

参数：旋转的角度

旋转的中心点是坐标轴的原点，旋转是以顺时针方向进行的，要想逆时针旋转时，将angle设定为负数即可

绘制图像

在HTML5中，不仅可以使用Canvas API绘制图形，还可以读取磁盘或网络中的图像文件，然后使用Canvas API将该图像绘制在画布中。

◆创建图像对象

```
var image = new Image();

image.src = “images/a.jpg”;
```

◆绘制图像

- context.drawImage(image,x,y);

  参数：图像对象，绘制位置的起始x坐标，起始y坐标

- context.drawImage(image,x,y,w,h);

  参数：图像对象，绘制位置的起始x坐标，起始y坐标，绘制图像的宽，高

绘制图像

当图像文件是一个来源于网络的比较大的图像文件时，用户就需要耐心等待图像全部装载完毕才能看到该图像，这种情况下可以使用以下方法来解决

```
image.onload = function(){
        //绘制图像的函数
}
```

绘制图像

图像平铺

- **createPattern(image,type)**

参数：image对象，平铺类型

type的取值如下:

no-repeat	不平铺
repeat-x	横向平铺
repeat-y	纵向平铺
repeat	全方向平铺

返回值：pattern对象，可以为fillStyle的值

- 示例

```
var pattern = createPattern(image,'repeat');
context.fillStyle = pattern;
context.fillRect(x,y,width,height);
```

图像裁剪

Canvas API的图像裁剪功能是指，在画布内使用路径，只绘制该路径所包含区域内的图像，不绘制路径外部的图像。

- **context.clip();**

这个方法会把Canvas的当前路径裁剪下来，一旦调用了clip方法之后，接下来向Canvas绘制图形时，之后clip()裁剪的路径覆盖的部分才会被显示出来。

- 示例

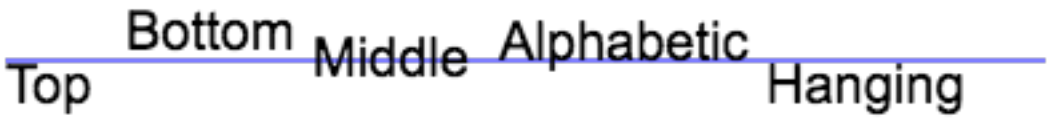
```
context.beginPath();
context.arc(200,150,80,0,Math.PI*2);
context.closePath();
context.clip();
var image = new Image();
image.src = "../images/1.jpg";
image.onload = function () {
        context.drawImage(image,0,0,400,300);
}
```

绘制文本

在HTML5中，可以在Canvas画布中进行文字的绘制，同时也可以指定绘制文字的 字体大小，对齐方式，文字文理填充等。

◆2D上下文文字修饰

- font　　设置文字字体
- textAlign　　文本对齐方式 start，end，left，right，center
- textBaseline　　文本的基线，top，hanging，middle，alphabetic，ideographic，bottom



绘制文本

在HTML5中，可以在Canvas画布中进行文字的绘制，同时也可以指定绘制文字的 字体大小，对齐方式，文字文理填充等。

◆绘制填充或勾勒文本

- fillText(text,x,y,width)　　使用fillStyle属性绘制填充文本
- strokeText(text,x,y,width)　　使用strokeStyle属性向文本描边

参数：

- ✓ text　　表示要绘制的文字
- ✓ x,y　　表示绘制文字的起点横坐标与纵坐标
- ✓ maxWidth　　表示显示文字时的最大宽度，防止文字溢出

```
context3.font = 'italic 30px sans-serif';
context3.textBaseline = 'top';
context3.textAlign = 'start';
context3.fillText('hello world',150,60);
```

杰普软件科技有限公司
www.briup.com

Tel: (021)55660810
Fax: (021)55660802
Email:training@briup.com
Msn: training.sh@hotmail.com
Home:http://www.briup.com

地址：上海市闸北区万荣路1188弄F栋6号三层-上海服务外包科技园龙软园区
邮编：200436

电话：021-56657112
传真：021-55661523-8003
电邮：training@briup.com
主页：http://www.briup.com

Briup High-End IT Training

HTML5 API

第 3 章 拖放API

Brighten Your Way And Raise You Up.

学习目标

- 掌握利用拖放API将页面中的元素拖放
- 掌握DataTransfer对象的属性和方法
- 掌握拖放时的视觉效果的设定
- 掌握自定义拖放

拖放介绍

拖放是一种常见的特性，即抓取对象以后拖到另一个位置。虽然在HTML5之前已经可以使用mousedown,mousemove,mouseup实现拖放，但是只支持在浏览器内部的拖放，在HTML5中，拖放是标准的一部分，任何元素都能够拖放，并且支持浏览器与其他应用程序之间的数据的互相拖放，同时大大简化了拖放方面的相关代码。

拖放实现步骤

拖放步骤

- 将想要拖放的对象元素的draggable属性设为true,另外img与a元素默认允许拖放
- 编写与拖放相关的事件处理代码

拖放事件

通过拖放事件，可以控制拖放相关的各个方面。其中最关键的地方在于确定哪里发生了拖放事件，有些事件是在被拖动的元素上触发的，有些事件是在放置目标上触发的。

◆拖动事件

- **dragstart** 按下鼠标并开始移动鼠标时，会在拖放的元素上触发dragstart事件。此时光标变成了“不能放”符号(圆环中有一条反斜线)，表示不能把元素放到自己上面。可以通过ondragstart事件处理程序来运行JavaScript代码
- **drag** 触发dragstart事件后，随即会触发drag事件，而且在元素被拖动期间会持续触发该事件(类似mousemove事件)
- **dragend** 当拖放停止时(无论是把元素放到了有效的放置目标，还是放置到了无效的放置目标上)都会触发dragend事件

拖放事件

- 放置事件
  - **dragenter** 只要有元素被拖动到放置目标上，就会触发dragenter事件（类似mourseover事件），元素被拖出了放置目标，会触发dragleave
  - **dragover** dragenter紧随其后的就是dragover事件，而且在被拖动的元素还在放置目标的范围内移动时，就会持续触发该事件
  - **drop** 将拖动元素放置到目标元素上的时候会激发
- 拖放事件生命周期
  - dragstart-→drag→dragenter→dragover→drop→dragend

自定义放置目标

有些元素不允许放置目标元素，所以需要将某个元素变成有效的放置目标。方法是重写dragover事件，阻止事件的默认行为。

```
$('.content').on('dragover',function(event){
        event.preventDefault();
});
```

dataTransfer对象

在拖放操作时实现数据交换，它是事件对象的一个属性，用于从被拖拽元素向放置目标传递字符串格式的数据。因为它是事件对象的属性，所以只能在拖放事件的事件处理程序中访问dataTransfer对象。

◆获取方式

- **event. dataTransfer**

◆方法

dataTransfer.setData(key,val);	在dataTransfer中设置值,在dragstart赋值
dataTransfer.getData(key);	获取由setData保存的值,在drop取值
clearData(format)	从DataTransfer对象或ClipboardData对象中删除指定格式或全部kind值为string的数据。仅在dragstart事件中调用，在其他事件中调用会抛InvalidStateError
setDragImage(element,x,y)	指定一幅图像，当拖放发生的时候，显示在光标下方

dataTransfer对象

属性

- **effectAllowed** 该属性与dropEffect属性结合起来可以设定拖放时的视觉效果。effectAllowed属性表示当一个元素被拖动时允许的视觉效果，仅能在dragstart事件中设定，其他事件中设置均无效
- **dropEffect** 用于设置目标元素将执行的操作，表示实际拖放时的视觉效果，仅能在dragover事件中指定。若属性值属于effectAllowed范围内，则鼠标指针将显示对应的指针样式，否则则显示禁止的指针样式。
- **types** 数据类型为DOMStringList，存储DataTransfer对象中所有数据项的数据类型
- **items** 数据类型为DataTransferItemList，存储DataTransfer对象中所有的数据项

杰普软件科技有限公司
www.briup.com

Tel: (021)55660810
Fax: (021)55660802
Email:training@briup.com
Msn: training.sh@hotmail.com
Home:http://www.briup.com

地址：上海市闸北区万荣路1188弄F栋6号三层-上海服务外包科技园龙软园区
邮编：200436

电话：021-56657112
传真：021-55661523-8003
电邮：training@briup.com
主页：http://www.briup.com

Briup High-End IT Training

HTML5 API

第 4 章 离线应用与客户端存储

Brighten Your Way And Raise You Up.

学习目标

- 掌握Web Storage的基本概念，了解sessionStorage和localStorage使用方法以及区别
- 掌握使用sessionStorage和localStorage进行数据存储，对象存储的方法

客户端存储

随着Web应用的出现，也产生了对于能够直接在客户端上存储用户信息能力的要求。由于cookie与IE中提供的持久化用户数据的容量限制以及数据安全问题，不能大量存储数据和安全数据，H5中提供了Web Storage。

Web Storage的两个目标

- 提供一种在cookie之外存储会话数据的途径
- 提供一种存储大量可以跨会话存在的数据的机制

Web Storage种类

- **sessionStorage对象**

  将数据保存在session对象中。session是指用户在浏览某个网站时，从进入网站到浏览器关闭所经过的这段时间，也就是用户浏览这个网站所花费的时间。session对象可以用来保存在这段时间内所要保存的任何数据。(仅限当前选项卡)。

- **localStorage对象**

  将数据保存在客户端本地的硬盘设备中，即使浏览器被关闭了，该数据仍然存在，下次打开浏览器访问网站时仍然可以继续使用。

Web Storage

Storage类型提供了大量的存储空间来存储键值对，Storage实例与其他对象类似，Storage类型只能存储字符串，非字符串的数据在存储之前会被转换成字符串。

◆Storage对象提供的方法

setItem(name,value)	为指定的name设置一个对应的值
getItem(name)	根据指定的名字name获取对应的值
removeItem(name)	删除由name指定的键值对
clear()	删除所有值
key(index)	获得index位置处的值的名字
length	Storage对象中，键值对的数量

- **window对象提供的storage事件**

当在【其他页面】中修改**sessionStorage**或者**localStorage**中的值时要执行的处理，可以使用**window**对象提供的**storage**事件来监听。

event.key	被修改的数据键值
event.oldValue	被修改前的值
event.newValue	被修改后的值
event.url	页面的URL地址
event.storageArea	为变动的sessionStorage对象或localStorage对象

杰普软件科技有限公司
www.briup.com

Tel: (021)55660810
Fax: (021)55660802
Email:training@briup.com
Msn: training.sh@hotmail.com
Home:http://www.briup.com

地址：上海市闸北区万荣路1188弄F栋6号三层-上海服务外包科技园龙软园区
邮编：200436

电话：021-56657112
传真：021-55661523-8003
电邮：training@briup.com
主页：http://www.briup.com

Briup High-End IT Training

HTML5 API

第 5 章 Web Workers

Brighten Your Way And Raise You Up.

Web Workers简介

Web Workers是HTML5新增的，用来在Web应用程序中实现后台处理的一项技术。在使用HTML4的JavaScriptt创建出来的Web程序中，因为所有的处理都是在单线程内执行的，所以如果花费的事件比较长的话，程序界面会处于长时间没有响应的状态，等时间长到一定程度时，浏览器还会跳出一个提示脚本运行时间过长的提示框，使用户不得不中断正在执行的处理。为了解决这个问题，H5新增了Worker-API，使用这个API用户可以创建在后台运行的线程，将程序交给后台运行，对用户在前台页面中执行的操作完全没有影响。

Web Worker的API

- **Worker【构造函数】**
  - onmessage　　监听，当在后台线程代码中调用postMessage时激发
  - postMessage　　调用后台线程。字符串参数将赋值给后台线程中事件对象event.data
- **局限性**
  - 不能跨域加载JS
  - worker内代码不能访问DOM
  - 各个浏览器对Worker的实现不大一致，例如Firefox里允许worker中创建新的worker,而Chrome中就不行

使用方式

- 编写后台执行代码

calculate.js

```
onmessage = function (event) {
        var num = event.data;
        var result = 0;
        for(var i=0;i<num;i++){
                result+=i;
        }
        //向线程创建源送回消息
        postMessage(result);
}
```

- 创建后台线程

```
var worker = new Worker("calculate.js");
```

使用方式

- 启动后台线程

  使用**Worker**对象的**postMessage**方法来对后台线程发送消息

```
worker.postMessage(100000000);
```

- 设定监听，等待线程结束返回计算结果

```
worker.onmessage = function(event){
        //处理接收到的消息
        alert(event.data);
};
```

Question & Answer