第9章 HTML5 API

课程介绍

- 媒体操作
- 手势事件
- 拖放事件
- 定位事件

9.1 媒体操作

9.1.1 <audio>音频标签

<audio> 标签定义声音，比如音乐或其他音频流。

可以在开始标签和结束标签之间放置文本内容，这样老的浏览器就可以显示出不支持该标签的信息。

实例：

```
<audio src="sources/Itsok.mp3" controls="controls">
    您的浏览器不支持 audio 标签。
</audio>
```

9.1.1.1 Audio对象的标签属性

属性	值	描述
autoplay	autoplay	如果出现该属性，则视频在就绪后马上播放。
controls	controls	如果出现该属性，则向用户显示控件，比如播放按钮。
loop	loop	如果出现该属性，则当媒介文件完成播放后再次开始播放。
preload	preload	如果出现该属性，则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	要播放的视频的 URL。

9.1.1.2 音频标签方法

方法	描述
canPlayType(type)	检测浏览器是否支持音频类型，返回'probable'：浏览器最可能支持该类型；'maybe'：可能支持；''：不支持
load()	重新加载音频，用于更改src之后使用，无参数，无返回值
play()	播放音频，无参数，返回一个promise
pause()	暂停音频，无参数，无返回值

canPlayType(type)：type规定要检测的音频/视频类型。

常用值：

- · ogg
- · mp3
- · webm
- · mpeg
- · ogg
- · mp4

9.1.1.3 JavaScript只读属性

buffered

- audio.buffered.end(0)：获取已缓冲的秒数
- audio.buffered.length：获取缓冲范围
- audio.buffered.start(index)：获取某个已缓冲返回的开始位置
- audio.buffered.end(index)：获取某个已缓冲范围的结束位置
- currentSrc：返回当前音频的URL
- currentTime：返回当前音频的现在时间
- ended：音频是否结束
- duration：返回音频时长，以秒计
- networkState：返回音频的网络状态[0:尚未初始化；1:已经选取资源，但未使用网络；2:正在下载数据；3:未找到资源]
- paused：是否处于暂停状态

实例：

```
var myAudio = document.getElementById("media");
//播放音频
myAudio.play();
console.log(myAudio.canPlayType('audio/mp3'));// probably
//1.5秒后暂停播放
setTimeout(function() {
    myAudio.pause();
    console.log(myAudio.currentTime);       // 1.5536
    console.log(myAudio.ended);          // false
    console.log(myAudio.duration);       // 195.265306
    console.log(myAudio.networkState);  // 1
    console.log(myAudio.paused);            // true
},1500);
```

9.1.2 视频标签

<video>标签定义视频，比如电影片段或其他视频流。

可以在开始标签和结束标签之间放置文本内容，这样老的浏览器就可以显示出不支持该标签的信息。

实例：

```
<video src="sources/test.mp4" controls="controls">
    您的浏览器不支持 video 标签。
</video>
```

9.1.2.1 视频标签属性

属性	值	描述
autoplay	autoplay	如果出现该属性，则视频在就绪后马上播放。
controls	controls	如果出现该属性，则向用户显示控件，比如播放按钮。
height	pixels	设置视频播放器的高度。
loop	loop	如果出现该属性，则当媒介文件完成播放后再次开始播放。
preload	preload	如果出现该属性，则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	要播放的视频的 URL。
width	pixels	设置视频播放器的宽度。

9.1.2.2 Media对象方法和属性

Media方法和属性——HTMLVideoElement继承自 HTMLMediaElement。

视频的播放进度状态等

方法	描述
Media.currentTime = value;	当前播放的位置，赋值可改变位置（以秒计）
Media.startTime;	一般为0，如果为流媒体或者不从0开始的资源，则不为0
Media.duration;	当前资源长度 流返回无限
Media.paused;	是否暂停
Media.defaultPlaybackRate = value;	默认的回放速度，可以设置，默认1.0
Media.playbackRate = value;	当前播放速度，设置后马上改变，默认1.0
Media.played;	返回已经播放的区域，TimeRanges，关于此对象见下文
Media.seekable;	返回可以seek的区域 TimeRanges
Media.ended;	是否结束
Media.autoPlay;	是否自动播放
Media.loop;	是否循环播放
Media.play();	播放
Media.pause();	暂停
Media.controls	是否有默认控制条
Media.volume	音量0.1~1.0
Media.muted	静音，true|false

实例：

```
<body>
    <video id="media" src="sources/test.mp4" controls width="400px" heigt="400px">
</video>
    <button>播放</button>
</body>

<script type="text/javascript">
    var video = document.getElementById("media");
    var btn = document.getElementsByTagName("button")[0];
    // 实现切换播放
    btn.onclick = function(){
        if(this.innerHTML=="播放"){
            video.play()
            this.innerHTML = "暂停"
        }else{
            video.pause()
            this.innerHTML = "播放"
        }
    }
```

```
</script>
```

9.2 手势事件

9.2.1 click事件

单击事件，类似于PC端的click，但在移动端中，连续click的触发有200ms ~ 300ms的延迟。

9.2.2 touch类事件

触摸事件，有touchstart touchmove touchend touchcancel 四种之分。

touch类事件：

事件名	解释
touchstart	手指触摸到屏幕会触发
touchmove	当手指在屏幕上移动时，会触发
touchend	当手指离开屏幕时，会触发
touchcancel	可由系统进行的触发，比如手指触摸屏幕的时候，突然alert了一下，或者系统中其他打断了touch的行为，则可以触发该事件

9.2.3 tap类事件

触碰事件，一般用于移动端代替click事件，有tap longTap singleTap doubleTap四种之分tap 事件在用户轻击一个元素时触发。

tap类事件：

事件名	解释
tap	手指碰一下屏幕会触发
longTap	手指长按屏幕会触发
singleTap	手指碰一下屏幕会触发
doubleTap	手指双击屏幕会触发

9.2.4 swipe类事件

滑动事件，有swipe swipeLeft swipeRight swipeUp swipeDown 五种之分。

swipe类事件：

事件名	解释
swipe	手指在屏幕上滑动时会触发
swipeLeft	手指在屏幕上向左滑动时会触发
swipeRight	手指在屏幕上向右滑动时会触发
swipeUp	手指在屏幕上向上滑动时会触发
swipeDown	手指在屏幕上向下滑动时会触发

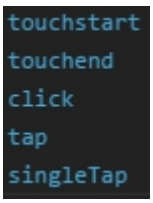
9.2.5 事件监听

实例：

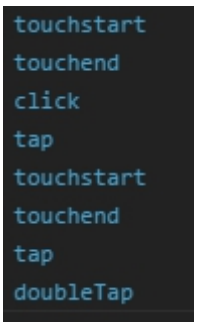
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <meta content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-
scalable=0" name="viewport">
        <title></title>
        <style type="text/css">
            #box {
                overflow: hidden;
                margin: 50px auto;
                width: 200px;
                height: 200px;
                border: 1px solid #ccc;
            }
            .aa {
                float: left;
                margin: 10px;
                width: 100px;
                height: 100px;
                text-align: center;
                line-height: 100px;
                font-size: 32px;
                background-color: #ccc;
            }
        </style>
    </head>
    <body>
        <div id="box">
            <div class="aa">aa</div>
        </div>
        <div id="write"></div>
    </body>
    <script type="text/javascript">
        var box = document.getElementById('box');
        var write = document.getElementById('write');
        var aa = document.querySelector('.aa');
```

```
        var str = '';
        function addEvent(elem, type, showAll) {
            type = type.split(' ');
            type.forEach(function(item) {
                elem.addEventListener(item, function(ev) {
                    str += (showAll ? ev : ev.type) + '<br />';
                    write.innerHTML = str;
                });
            });
        }
        addEvent(aa, 'tap click touchstart touchmove touchend touchcancel swipe swipeLeft swipeRight swipeUp swipeDown longTap singleTap doubleTap', false);
    </script>
</html>
```

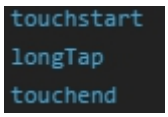
运行，点一下“aa”，可以看到click事件在touchend之后。



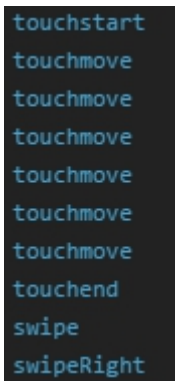
快速点两下，如图为相关事件触发的顺序，可以看到click事件因为延迟的原因只触发了一次。



长按，如图为相关事件触发的顺序。



向右滑动一下，如图为相关事件触发的顺序。



9.3 拖放事件

拖放是一种常见的特性，即抓取对象以后拖到另一个位置。在 HTML5 中，拖放是标准的一部分，任何元素都能够拖放。

注意：Internet Explorer 9、Firefox、Opera 12、Chrome 以及 Safari 5 支持拖放。 在 Safari 5.1.2 中不支持拖放。

设置元素可以被拖动，必须现在元素上添加属性draggable，否则拖动无效。

实例：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>
    <body>
        <div title="拖拽我" draggable="true">拖动我吧</div>
    </body>
</html>
```

事件说明：

事件	说明	作用位置
ondragstart	当拖拽元素开始被拖拽的时候触发的事件	被拖曳元素上
ondragenter	当拖曳元素进入目标元素的时候触发的事件	目标元素上
ondragover	拖拽元素在目标元素上移动的时候触发的事件	目标元素上
ondrop	被拖拽的元素在目标元素上同时鼠标放开触发的事件	目标元素上
ondragend	当拖拽完成后触发的事件	被拖曳元素上

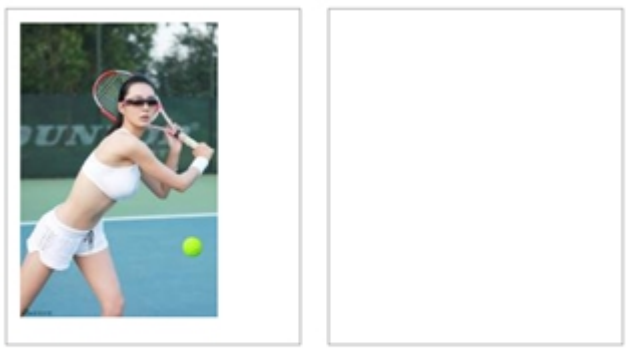
注意：

1. 事件对象中包含DataTransfer对象，它是用来拖拽对象用来传递的媒介，使用一般为Event.dataTransfer 。
2. Event.effectAllowed 属性：就是拖拽的效果。
3. Event.preventDefault() 方法：阻止默认的些事件方法等执行。在ondragover中千万一定要执行preventDefault()，否则ondrop事件不会被触发。另外，如果是从其他应用软件或是文件中拖东西进来，尤其是图片的时候，默认的动作是显示这个图片或是相关信息，并不是真的执行drop。此时需要用用document的ondragover事件把它直接干掉 。

实例：

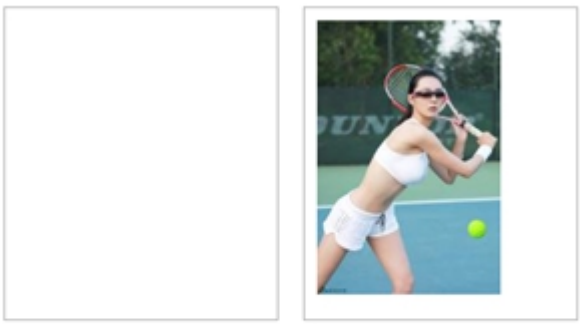
```
<!DOCTYPE HTML>
<html>
    <head>
        <style type="text/css">
            #box1,
            #box2 {
                float: left;
```

```
                width: 198px;
                height: 230px;
                margin: 10px;
                padding: 10px;
                border: 1px solid #aaaaaa;
            }
        </style>
        <script type="text/javascript">
            function allowDrop(ev) { // 拖动的时候组织默认事件
                ev.preventDefault();
            }
            function drag(ev) { // 被拖动的元素拖动的时候，写入数据到dataTransfer
                // 将拖动的元素的id保存在数据data中
                ev.dataTransfer.setData("data", ev.target.id);
            }
            function drop(ev) { // 放的时候
                ev.preventDefault();
                // 从dataTransfer对象中获取数据data去拿到id
                var id = ev.dataTransfer.getData("data");
                // 当前这个节点添加新的内容
                ev.target.appendChild(document.getElementById(id));
            }
        </script>
    </head>
    <body>
        <div id="box1" ondrop="drop(event)" ondragover="allowDrop(event)">
            <img src="img/test.jpg" draggable="true" ondragstart="drag(event)"
id="drag1" />
        </div>
        <div id="box2" ondrop="drop(event)" ondragover="allowDrop(event)"></div>
    </body>
</html>
```



拖动前

拖动中



拖动完成

9.4 定位事件

HTML5 Geolocation API 用于获得用户的地理位置。 Internet Explorer 9、Firefox、Chrome、Safari 以及 Opera 支持地理定位。对于拥有 GPS 的设备，比如 iPhone，地理定位更加精确。该对象主要有三个方法：getCurrentPosition()，watchPosition()，clearWatch()。但是我们要判断浏览器的支持情况。

实例：

```
if (navigator.geolocation){  // 支持
}else{  // 不支持
    console.log("您的浏览器不支持定位")
}
```

9.4.1 getCurrentPosition方法

getCurrentPosition()方法 可以传递三个参数。调用这个方法就会触发请求用户共享地理定位信息的对话框 。这个方法接收三个具体的参数分别是成功的回调函数，失败回调函数，以及配置选项。

getCurrentPosition方法参数说明：

参数	说明	是否必须
successFn(position)	成功回调函数 ,成功的回调函数会返回一个位置数据对象	是
errorFn(msg)	失败回调函数 ， 返回一个msg对象包含错误信息和代码	否
option	配置选项对象	否

- position对象有两个属性：coords 和 timestamp 。 coords 对象中将包含下列与位置相关的信息。
- latitude：以十进制度数表示的维度
- longtitude：以十进制度数表示的经度
- accuracy：经纬度坐标的精度，以米为单位

有些浏览器可能会在 coords 对象中提供如下属性。

- altitude：以米为单位的海拔高度，如果没有相关数据则值为 null
- altitudeAccuracy：海拔高度的精度，以米为单位，数值越大越不精确
- heading：指南针的方向，0°表示正北，值为 NaN 表示没有检测到数据
- speed：速度，即每秒移动多少米，如果没有相关数据则值为 null
- msg包含两个属性：message 和 code。其中，message 属性是提示文本消息，解释为什么会出错，而 code 属性中保存着一个数值，表示错误的类型：用户拒绝共享（1），位置无效（2）或者超时（3）

实例：

```
if (navigator.geolocation){  // 支持
    navigator.geolocation.getCurrentPosition(success, error);
    // 成功的回调函数
    function success(position) {
        // 获取经纬度信息
        console.log(position.coords.latitude, position.coords.longitude);
    }
    // 失败的回调函数
    function error(msg) {
        // 输出失败的信息
        console.log(msg.code, msg.message);
    }
}else{  // 不支持
    console.log("您的浏览器不支持定位")
}
```

注意：因为我们的这个API默认访问的使用谷歌地图，所以很有可能会出现错误，因为国内访问谷歌地图服务器有限制。

9.4.2 watchPosition()

如果要跟踪用户的位置，那么可以使用 watchPosition() 方法。这个方法的使用和 getCurrentPosition() 完全相同。实际上 watchPosition() 与定时调用 getCurrentPosition() 能得到相同效果。在第一次调用 watchPosition() 方法后，会取得当前位置，执行成功回调或者错误回调。然后，watchPosition() 就地等待系统发出位置已改变的信号。

调用 watchPosition() 会返回一个数值标识符，用于跟踪监控的操作。基于这个返回值可以取消监控操作，只要将其传递给 clearWatch() 方法即可（与使用 setTimeout() 和 clearTimeout() 类似），例如：

```
var watchId = navigator.geolocation.watchPosition(success, error);
clearWatch(watchId);
```

9.4.3 clearWatch()

清除一个跟踪控制监听。需要传入使用watchPosition函数返回的值。

9.4.4 百度地图获取当前位置

我们的百度地图和高度地图是目前国内主要的地图工具提供商。官方都提供了非常相信的信息内容。具体可以参考官方文档进行学习。

实例：

```
<!--// 注意要引入百度key-->
<script type="text/javascript" src="http://api.map.baidu.com/api?v=2.0&ak=???">
</script>

<script>
    var geolocation = new BMap.Geolocation();
    geolocation.getCurrentPosition(function(r){
        if(this.getStatus() == BMAP_STATUS_SUCCESS){
            alert('您的位置: '+r.point.lng+','+r.point.lat);
        }else {
            alert('failed'+this.getStatus());
        }
    });
</script>
```

9.5 课程总结

1.媒体操作

- <audio>音频标签
- Audio对象的标签属性
- 音频标签方法
- JavaScript只读属性
- 视频标签
- 视频标签属性
- Media对象方法和属性

2.手势事件

- click事件
- touch类事件
- tap类事件
- swipe类事件
- 事件监听

3.拖放事件

4.定位事件

- getCurrentPosition方法
- watchPosition()
- clearWatch()
- 百度地图获取当前位置

9.6 实训

实现一个音乐播放器，如下图：