事件5：触屏事件

原生触屏事件

回顾我们之前所讲的一些事件：如onclick点击事件，这些都是在PC端操作的，虽然他们也可以在移动端使用，但是因为有300毫秒的延时，在移动端的用户体验非常不好，所以js有了在移动端专用的触摸事件。

touch事件

一个触摸事件可以细分为：`touchstart`触摸开始,`touchmove`在屏幕移动`touchup`在屏幕上松开，这些事件分别对应着PC端的：`onmousedown`、`onmousemove`、`onmouseup`。我们可以通过这些事件来判断用户咋屏幕上的一些手势。下面这个比较少用：`touchcancel //触摸过程被系统取消时触发`

首先我们来看如何给一个元素绑定手势：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scal
e=1.0, maximum-scale=1.0, user-scalable=0">
        <title></title>
        <style type="text/css">
            #div1{
                width: 200px;
                height: 200px;
                background-color: #ccc;
            }
        </style>
    </head>
    <body>
        <div id="div1"></div>

    </body>
    <script type="text/javascript">
        var div1 = document.querySelector("#div1");
        // 给DIV添加触摸手势
        div1.addEventListener("touchstart",function (){
            alert('a');
        },false);
```

```
    </script>
</html>
```

以上代码运行的结果是只要我们一碰div，则弹出一个警告框出来（这里请使用谷歌浏览器的手机调试界面进行调试，否则可能不会出效果）。

touch事件中常见的属性

每个事件都有以下列表，比如`touchend`的`targetTouches`当然是 0 咯：

```
touches          //位于屏幕上的所有手指的列表
targetTouches    //位于该元素上的所有手指的列表
changedTouches   //涉及当前事件的所有手指的列表
```

每个事件有列表，每个列表还有以下属性：

```
pageX    //相对于页面的 X 坐标
pageY    //相对于页面的 Y 坐标
clientX  //相对于视区的 X 坐标
clientY  //相对于视区的 Y 坐标
screenX  //相对于屏幕的 X 坐标
screenY  //相对于屏幕的 Y 坐标

identifier // 当前触摸点的惟一编号
target   //手指所触摸的 DOM 元素
```

其他相关事件：

```
event.preventDefault ()    //阻止触摸时浏览器的缩放、滚动条滚动
var supportTouch = "createTouch" in document  //判断是否支持触摸事件
```

练习：使用原生的触摸事件实现滑动手势

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8" />
    <title>Document</title>
    <style type="text/css">
    body {
        height: 2000px;
```

```
    }
        #touch {
            width: 500px;
            height: 500px;
            position: absolute;
            background-color: red;
            margin: 10px;
        }
    </style>
</head>
<body>
    <div id="touch"></div>
</body>

<script type="text/javascript">
    var div = document.getElementById('touch');

    // 全局变量，用来保存初始触摸坐标
    var starX, starY;

    // 开始触摸事件
    function touchStarFun (ev) {
        var event = window.event || ev;
        console.log(event);
        // 获取到第一个坐标
        var star = event.touches[0];
        // x坐标
        starX = Number(star.pageX);
        // y坐标
        starY = Number(star.pageY);
        console.log(starY);

    }

    function touchMoveFun (ev) {
        var event = window.event || ev;
        var move = event.touches[0];
        var moveX = Number(move.pageX);
        var moveY = Number(move.pageY);

        if (moveX - starX > 100) {
            alert("向右移动");
        } else if (moveX - starX < -100) {
            alert("向左移动");
        }

        if (moveY - starY > 100) {
            alert("向下移动");
        } else if (moveY - starY < -100) {
            alert("向上移动");
        }
    }
```

```
    div.addEventListener('touchstart', touchStarFun, false);
    div.addEventListener('touchmove', touchMoveFun, false);

</script>
</html>
```

设备事件

我们在生活中最常见的就是微信的摇一摇，可以摇出一个新好友、识别一首歌曲。那么如何检测到用户摇动了手机呢？这里就需要调取陀螺仪这个设备，获取陀螺仪里的参数，进而来判断手机是否被摇动。

陀螺仪有一个三维坐标系，和CSS3的三维动画一样，X轴代表与手机屏幕横向方向，Y周代表与手机屏幕竖向方向，Z轴表示与手机屏幕垂直的方向，如下图：

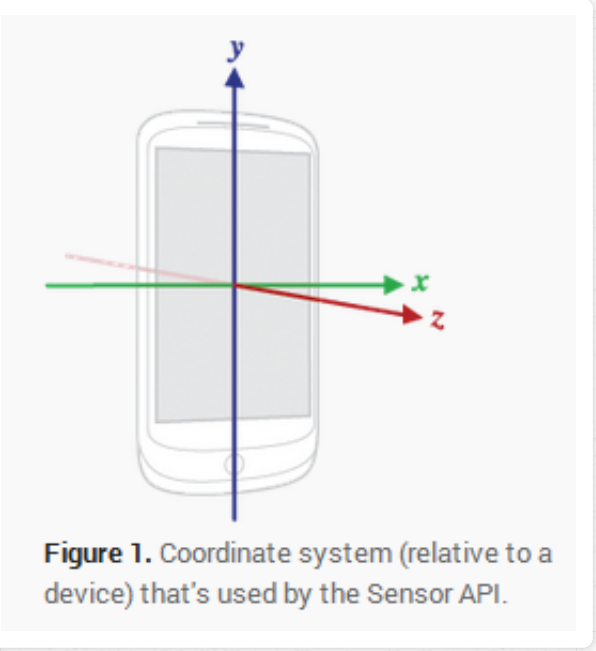


Figure 1. Coordinate system (relative to a device) that's used by the Sensor API.

在HTML5中，有两个关键属性可以让我们获取到手机加速计的相关参数：

1. deviceOrientation：封装了方向传感器数据的事件，可以获取手机静止状态下的方向数据，例如手机所处角度、方位、朝向等。
2. deviceMotion：封装了运动传感器数据的事件，可以获取手机运动状态下的运动加速度等数据。

示例：晃动手机后，弹框提示用户晃动了手机，并记录当前是第几次晃动

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
    </head>

    <!--加载body时调用js函数，载入加速计-->
    <body onload="init()">

    </body>
    <script type="text/javascript">
        // 设置一个初始速度，如果大于这个速度，就认为可以摇晃了手机
        var SHAKE_THRESHOLD = 3000;
        // 上一次晃动手机的时间
        var last_update = 0;
        // 初始化所有值
        var x = y = z = last_x = last_y = last_z = 0;

        function init() {
                // 判断设备是否支持硬件加速计
            if (window.DeviceMotionEvent) {
                window.addEventListener('devicemotion', deviceMotionHan
dler, false);
            } else {
                alert('not support mobile event');
            }
        }

        var times = 0;
        function deviceMotionHandler(eventData) {
                // 取出晃动事件中加速计的值（x\y\z三个坐标轴的值）
            var acceleration = eventData.accelerationIncludingGravity;

            var curTime = new Date().getTime();
            // 晃动时间间隔大于100毫秒，则认为是晃动了两次，否则是同一次晃动事件
            if ((curTime - last_update) > 100) {
                var diffTime = curTime - last_update;
                last_update = curTime;
                // 取出加速计中的三个方向坐标值
                x = acceleration.x;
                y = acceleration.y;
                z = acceleration.z;

                // 计算用户晃动手机的激烈程度
                var speed = Math.abs(x + y + z - last_x - last_y - last
_z) / diffTime * 10000;
                // 用户晃动激烈速度超过我们设定的阈值，则认为是一次用户刻意晃动手
机事件，而不是无意间触碰
                if (speed > SHAKE_THRESHOLD) {
                    alert("摇动了");
```

```
                    times++;
                    document.body.innerHTML = "摇动了"+times+"次";
                }
                last_x = x;
                last_y = y;
                last_z = z;
            }
        }
    </script>
</html>
```

仿微信摇一摇效果

这部分是项目，请参照课上代码*“yaoyiyao”*文件夹的项目

touch.js手势库

touch.js手势库是由百度云团队开发的一套手势库，包含了滑动、缩放、点击、长按等手势，我们可以直接拿来集成在我们的项目中使用。需要注意的时：这个库使用的浏览器内核是webkit，在移动端支持非常好。

touch.js介绍：http://touch.code.baidu.com

引入touch.js

集成touch.js有两种方式，第一种不需要下载他的源码：

```
<script src="http://code.baidu.com/touch-0.2.14.min.js"></script>
```

这种方式直接去服务器中找到这个文件。注意：有时可能找不到。。。。建议使用第二种方式：下载源码，然后手动引入需要的文件中：

```
<script src="js/touch-0.2.14.min.js"></script>
```

使用方法

在引入库后，调用他的`touch.on()`方法即可，参数列表中有三个参数，第一个表示手势添加的对象；第二个表示要添加的是哪种手势，第三个是手势所触发的函数。

示例：使用touch.js给div添加拖拽手势

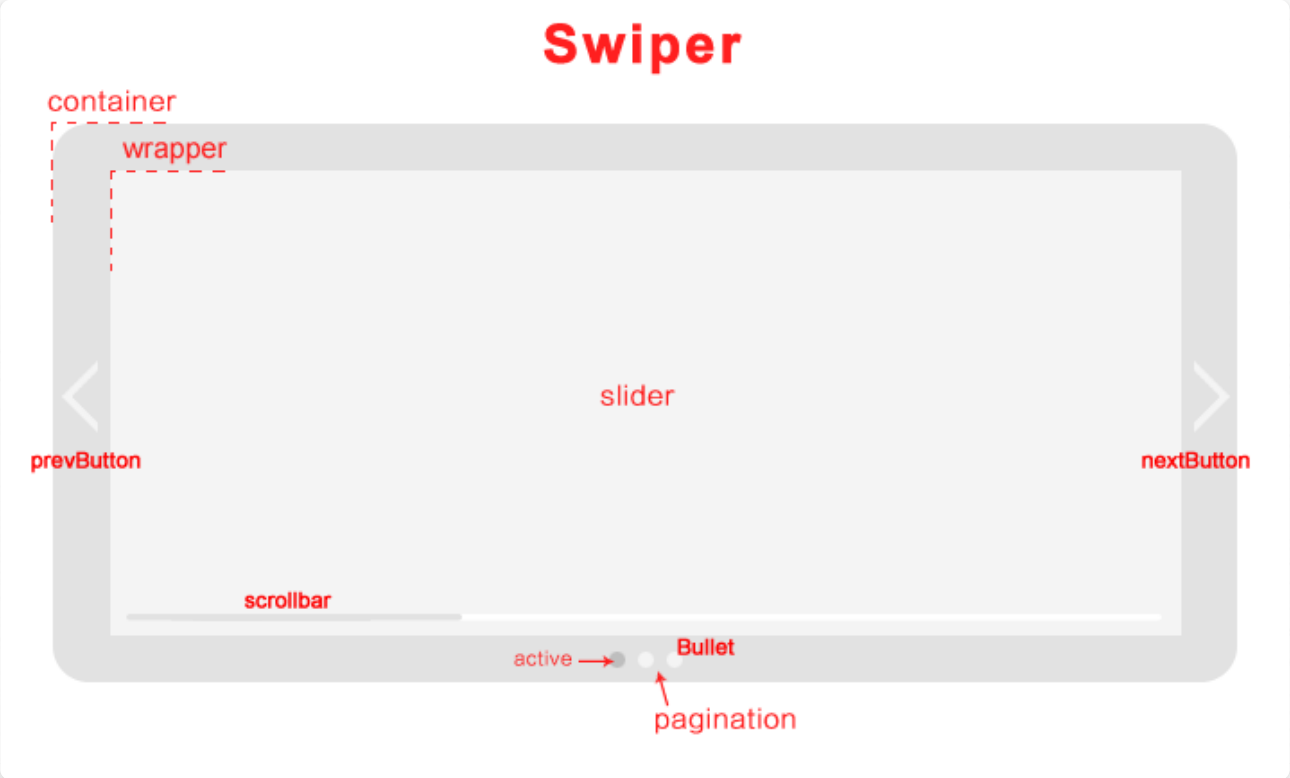
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
        <style type="text/css">
            #div1 {
                width: 100px;
                height: 100px;
                background-color: red;
                position: absolute;
            }
        </style>
    </head>
    <body>
        <div id="div1"></div>
    </body>
    <script src="js/touch-0.2.14.min.js"></script>
    <!--<script src="http://code.baidu.com/touch-0.2.14.min.js"></scrip
t>-->
    <script type="text/javascript" >

        // 给div1绑定开始触摸事件
        touch.on('#div1', 'touchstart', function(ev){
            // 取消系统事件
            ev.preventDefault();
            console.log("开始拖动");
            });


    var dx, dy;
    // 给div1绑定拖动事件
    touch.on('#div1', 'drag', function(ev){
        dx = dx || 0;
        dy = dy || 0;
        console.log("当前x值为:" + dx + ", 当前y值为:" + dy +".");
        var offx = dx + ev.x + "px";
        var offy = dy + ev.y + "px";
        this.style.webkitTransform = "translate3d(" + offx + "," + offy
 + ",0)";
    });

    touch.on('#div1', 'dragend', function(ev){
        dx += ev.x;
        dy += ev.y;
        console.log("结束");
    });

    </script>
</html>
```

更多使用方法示例：http://touch.code.baidu.com/examples.html

swiper.js

与touch.js不同，swiper.js虽然也是用来处理用户手势的但是他更专注于处理用户的滑动手势，可以通过swiper.js来实现各种酷炫的滑动效果。

swiper.js的结构及名词解释



名词	翻译	描述
Swiper	滑动、切换	整个滑动对象，有时特指滑块释放后仍然正向移动
container	容器	Swiper的容器，里面包括滑动块（slides）的封套
wrapper	封套	触控的对象，可触摸区域，移动的块的集合，过渡
slider	滑块	切换的块中的一个，可以包含文字、图片、html元
pagination	分页器	指示slide的数量和当前活动的slide
active	活动的，激活的	可视的(visible)slide是活动的，当可视slide不止一个
callback	回调函数	在某些情况下触发

swiper.js的用法

1、引入swiper.min.js和swiper.min.css

这两个文件需要下载到本地，然后引入。js文件负责处理用户滑动手势，css文件负责渲染效果

下载地址：
swiper.min.js:http://www.swiper.com.cn/download/index.html#file7

swiper.min.css: http://www.swiper.com.cn/download/index.html#file5

更多介绍请参照：http://www.swiper.com.cn/usage/index.html

2、创建滚动视图及绑定事件

以下demo只是部分配置，详细配置请带着学生参见官方文档：http://www.swiper.com.cn/api/index.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8">
        <title></title>
         <!--加载插件-->
        <link rel="stylesheet" type="text/css" href="css/swiper-3.3.1.m
in.css"/>
        <script type="text/javascript" src="js/swiper-3.3.1.min.js"></s
cript>
        <style type="text/css">
            /*设置样式*/
            .swiper-container {
                width: 300px;
                height: 200px;
                background-color: bisque;
            }

            .swiper-slide {
                width: 300px;
                height: 200px;
                background: green;
                font-size: 50px;
                color: white;
                text-align: center;

            }

        </style>
    </head>

    <body>
```

```
        // 配置结构
        <div class="swiper-container">
            <div class="swiper-wrapper">
                <div class="swiper-slide">slide1</div>
                <div class="swiper-slide" >slide2</div>
                <div class="swiper-slide">slide3</div>
            </div>

            // 添加两个箭头
            <div class="swiper-button-prev"></div>
            <div class="swiper-button-next"></div>
        </div>

    </body>

    <script type="text/javascript">
        var mySwiper = new Swiper('.swiper-container', {

            // 轮播图滚动的相关设置
            initialSlide: 2, // 设置初始的是第几个图片，初始下标是0
            autoplay:500,   // 设置自动轮播时，停留在每张图片上的时间
//          autoplayDisableOnInteraction:false  // 用户手动滑动轮播图时，是否停止自动轮播事件。默认是true

            // 当前页面指示器相关设置

            // 前进、后退按钮（该项和autoplayDisableOnInteraction不能同时设置，因为他们的实现效果类似）
            prevButton:'.swiper-button-prev',
            nextButton:'.swiper-button-next',


        })

    </script>

</html>
```