The Device Motion API

INTRODUCTION

This section presents an API rather similar in its use to the device orientation API from the previous chapters.

The deviceMotion API deals with accelerations instead of just orientations.

Use cases proposed by the specification are:

- **Controlling a game**: a gaming Web application monitors the device's orientation and interprets tilting in a certain direction as a means to control an on-screen sprite.
- **Gesture recognition**: a Web application monitors the device's acceleration and applies signal processing in order to recognize certain specific gestures. For example, using a shaking gesture to clear a web form.
- **Mapping**: a mapping Web application uses the device's orientation to correctly align the map with reality.

BASIC USAGE

```
function handleMotionEvent(event) {
   var x =event.accelerationIncludingGravity.x;
   var y =event.accelerationIncludingGravity.y;
   var z =event.accelerationIncludingGravity.z;
   // Process ...
}
window.addEventListener("devicemotion",handleMotionEvent, true);
```

The deviceMotion API is rather straightforward and is very similar to the orientation API except that it returns more than just the rotation information, it also returns acceleration information about the device current motion. The acceleration is in three parts: acceleration along the X axis, the Y axis and the Z axis. Each value is in meters per second squared (m/s^2). The acceleration is returned as an "acceleration event" that has two properties:`accelerationIncludingGravity` and `acceleration`, which excludes the effects of gravity. There are two different values because some devices might be able to exclude the effect of gravity if equipped with a gyroscope. Indeed there is acceleration due implicitly to gravity, see also this: Acceleration of Gravity on Earth...

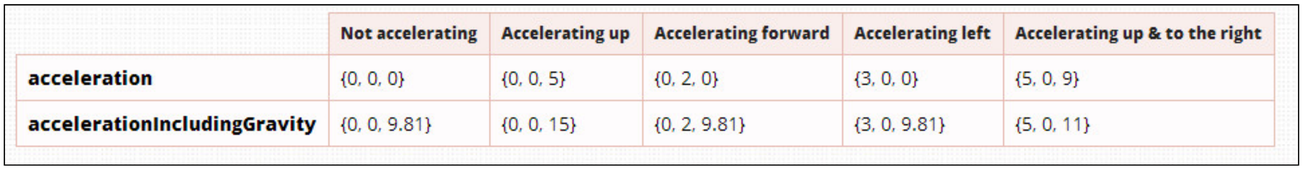
So if the device doesn't have a gyroscope, the `acceleration`property will be null. In this case you have no choice but to use the`accelerationIncludingGravity` property. Note that all IOS devices so far have a gyroscope.

BASICS ABOUT ACCELERATION

The device motion event is a superset of the device orientation event; it returns data about the rotation information and alsoacceleration information about the device. The acceleration data is returned in three axes: x, y and z. These are measured in meters per second squared (m/s^2). Because some devices might not have the hardware to exclude the effect of gravity, the event returns two properties, `accelerationIncludingGravity` and`acceleration.` The latter excludes the effects of gravity (when this is the case, the acceleration data will be null).

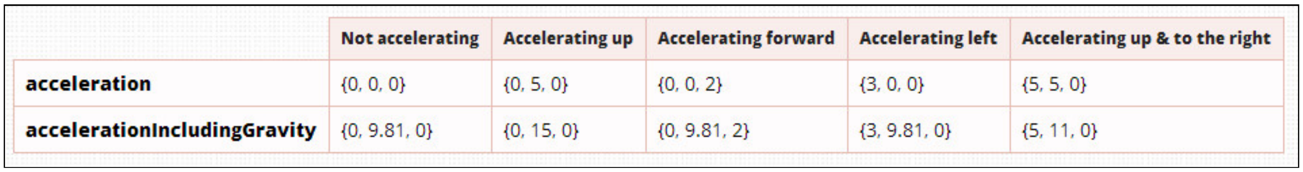
Example of acceleration values

If a laptop is in its normal position with the screen facing up, the data returned would be (info taken from:http://www.html5rocks.com/en/tutorials/device/orientation):



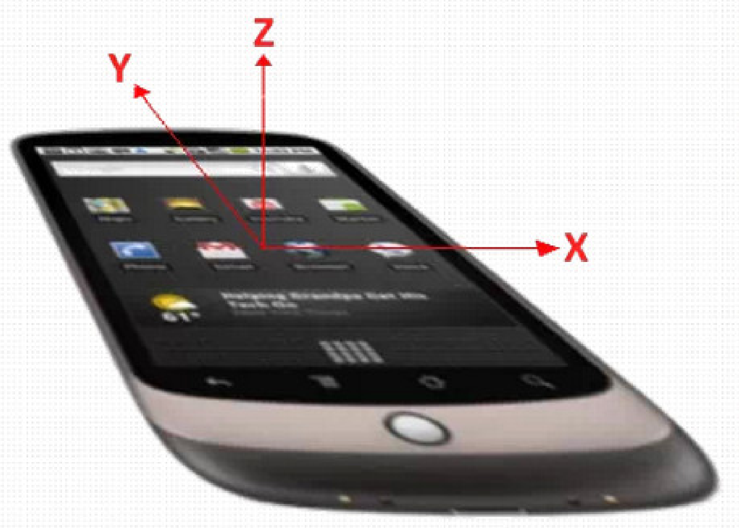
	Not accelerating	Accelerating up	Accelerating forward	Accelerating left	Accelerating up & to the right
acceleration	{0, 0, 0}	{0, 0, 5}	{0, 2, 0}	{3, 0, 0}	{5, 0, 9}
accelerationIncludingGravity	{0, 0, 9.81}	{0, 0, 15}	{0, 2, 9.81}	{3, 0, 9.81}	{5, 0, 11}

A mobile phone rotated along the x-axis so the screen is perpendicular to its normal position would return:



	Not accelerating	Accelerating up	Accelerating forward	Accelerating left	Accelerating up & to the right
acceleration	{0, 0, 0}	{0, 5, 0}	{0, 0, 2}	{3, 0, 0}	{5, 5, 0}
accelerationIncludingGravity	{0, 9.81, 0}	{0, 15, 0}	{0, 9.81, 2}	{3, 9.81, 0}	{5, 11, 0}

Remember the coordinate system for a mobile phone:



Common steps

The principle of use is the same as for the orientation API:

1. Test if the API is supported by the browser,
2. Add a listener for '`devicemotion`' events,
3. Get the acceleration values from the DOM event that has been passed to the listener,
4. Process the data.

Common processing with acceleration values

Test the value of the acceleration.z property: If > 0 then the device is facing up, otherwise it is facing down. This can be useful if you want to play heads or tails with your phone ;-)

```
    // For example, if acceleration.z is > 0 then the phone is facing up
```

```
var facingUp = -1;
if (acceleration.z > 0) {
  facingUp = +1;
}
```

Compute the angle corresponding to the Left / Right and Front / Back tilts. This example comes from:http://www.html5rocks.com/en/tutorials/device/orientation and uses the `accelerationIncludingGravity` property of the event.

```
function deviceMotionHandler(eventData) {
   // Grab the acceleration including gravity from the results
   var acceleration =eventData.accelerationIncludingGravity;
   // Convert the value from acceleration to degrees
   // acceleration.x|y is the acceleration according
   //  to gravity, we'll assume we're on Earth and divide
   // by 9.81 (earth gravity) to get a percentage value,
   // and then multiply that by 90 to convert to degrees.
   var tiltLR = Math.round(((acceleration.x)/ 9.81) * -90);
   var tiltFB = Math.round(((acceleration.y +9.81) / 9.81) * 90 * facingUp);
   // ... do something
}
```

Compute the vertical (direction of the sky) (this extract comes from a complete example further down this page)...

```
...
var angle = Math.atan2(accel.y,accel.x);
var canvas =document.getElementById('myCanvas');
var ctx = canvas.getContext('2d');

ctx.moveTo(50,50);
// Draw sky direction in the canvas
ctx.lineTo(50-50*Math.cos(angle),50+50*Math.sin(angle));
ctx.stroke();
```

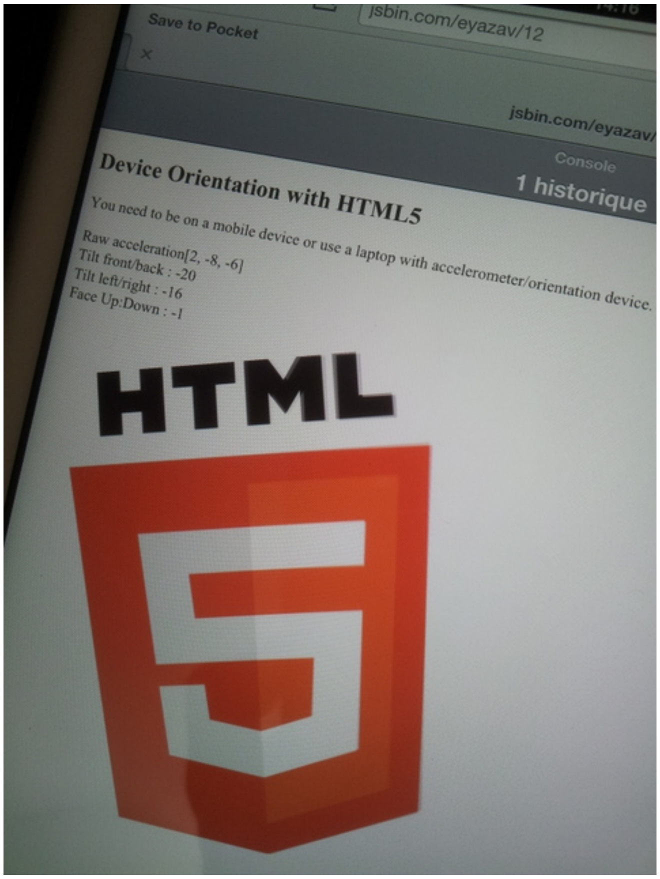
Use acceleration values to move a ball on the screen of a tablet when the tablet is tilted front / back or left / right (complete example later on)...

```
...
ball.x += acceleration.x;
ball.y += acceleration.y;
...
```

COMPLETE EXAMPLES

Move the HTML5 logo

Online example at JsBin, if using a mobile device, use this URL.



Device Orientation with HTML5

You need to be on a mobile device or use a laptop with accelerometer/orientation device.

Raw acceleration[2, -8, -6]
Tilt front/back : -20
Tilt left/right : -16
Face Up:Down : -1

Code from this example:

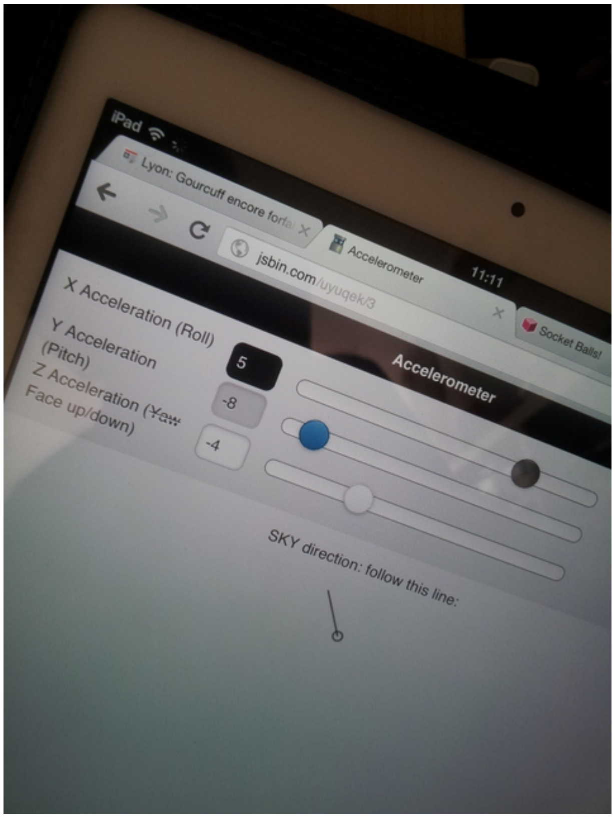
```
<!doctype html>
<html>
   <head></head>
   <body>
      <h2>Device Orientation with HTML5</h2>
      You need to be on a mobile device or use a laptop with
accelerometer/orientation
      device.
      <p>
      <div id="rawAccel"></div>
      <div id="tiltFB"></div>
      <div id="tiltLR"></div>
      <div id="upDown"></div>
      <imgsrc="http://www.html5rocks.com/en/tutorials/device/orientation/html5_logo.png"
id="imgLogo" class="logo">
      <script type="text/javascript">
         if (window.DeviceMotionEvent !=undefined) {
         console.log("DeviceMotion is supported");

         window.addEventListener('devicemotion',function(eventData) {
            // Grab the acceleration including gravity from the results
            var acceleration =eventData.accelerationIncludingGravity;
            // Display the raw acceleration data
            var rawAcceleration = "[" +Math.round(acceleration.x) + ", 
" +Math.round(acceleration.y)
            + ", " +Math.round(acceleration.z) + "]";
            // Z is the acceleration in the Z axis, and if the device
            // is facing up or down
            var facingUp = -1;
            if (acceleration.z > 0) {
               facingUp = +1;
            }
            // Convert the value from acceleration to degrees
            // acceleration.x|y is the acceleration according to gravity,
            //  we'll assume we're on Earth and divide
            // by 9.81 (earth gravity) to get a percentage value,
            // and then multiply that by 90 to convert to degrees.
            var tiltLR =Math.round(((acceleration.x) / 9.81) * -90);
            var tiltFB =Math.round(((acceleration.y + 9.81) / 9.81) *90 * facingUp);
            document.querySelector("#rawAccel").innerHTML=
                               "Raw acceleration" + rawAcceleration;
            document.querySelector("#tiltFB").innerHTML =
                               "Tilt front/back : " + tiltFB;
            document.querySelector("#tiltLR").innerHTML =
                               "Tilt left/right : " + tiltLR;
            document.querySelector("#upDown").innerHTML =
                               "Face Up:Down : " + facingUp;
            updateLogoOrientation(tiltLR,tiltFB);
         }, false);
      } else {
        alert("Not supported on your device or browser. Sorry.");
      }
      function updateLogoOrientation(tiltLR,tiltFB) {
         // USE CSS3 rotations for rotating the HTML5 logo
         //for webkit browser
         document.getElementById("imgLogo").style.webkitTransform =
         "rotate(" + tiltLR + "deg) rotate3d(1,0,0, " + (tiltFB * -1) + "deg)";
         //for HTML5 standard-compliance
         document.getElementById("imgLogo").style.transform =
         "rotate(" + tiltLR + "deg) rotate3d(1,0,0, " + (tiltFB * -1) + "deg)";
      }
   </script>
  </body>
</html>
```

Interesting example that uses jQuery mobile

This example comes from:http://www.emanueleferonato.com/2011/09/05/playing-with-javascript-iphone-and-devicemotion-event-listener/

It shows how the X and Y acceleration values can be used for indicating the sky's direction (vertical), and how the Z acceleration is in fact an indicator for the face up / face down orientation of the device.

This example has been adapted and put on jsbin.com so that you can tweak it: http://jsbin.com/uyuqek/4/edit



X Acceleration (Roll) 5

Y Acceleration (Pitch) -8

Z Acceleration (Yaw Face up/down) -4

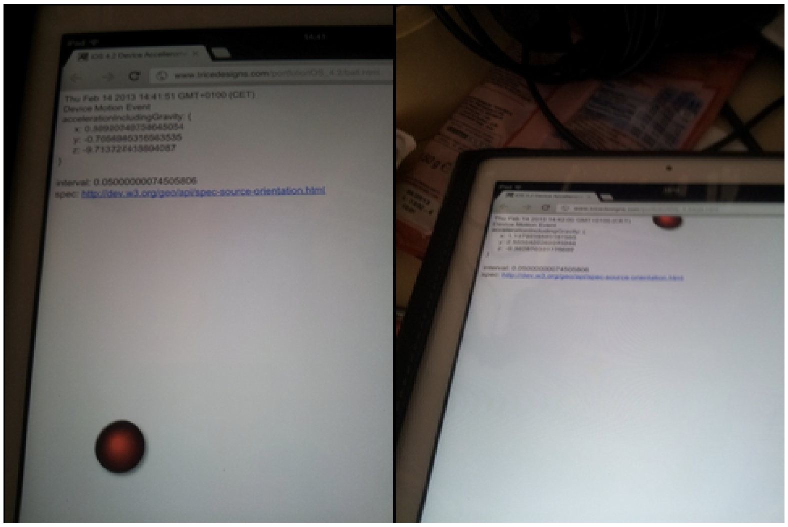
SKY direction: follow this line:

Code from the example:

```
<html>
   <head>
     <meta http-equiv="content-type"content="text/html; charset=utf-8">
     <meta name="viewport" content="user-scalable=no, width=device-width" />
     <link rel="stylesheet"
           href="http://code.jquery.com/mobile/1.0b2/jquery.mobile-
1.0b2.min.css" />
     <script type="text/javascript"
             src ="http://code.jquery.com/jquery-1.6.2.min.js">
     </script>
     <script type="text/javascript"
             src ="http://code.jquery.com/mobile/1.0b2/jquery.mobile-1.0b2.min.js">
     </script>
     <script type="text/javascript">
        $(document).ready(function(){
            window.addEventListener("devicemotion",onDeviceMotion,false);
        });
        function onDeviceMotion(event){
            var ctx =document.getElementById("c").getContext("2d");
            var accel =event.accelerationIncludingGravity;
            $("#sliderX").val(Math.round(accel.x)).slider("refresh");
            $("#sliderY").val(Math.round(accel.y)).slider("refresh");
            $("#sliderZ").val(Math.round(accel.z)).slider("refresh");
            // sky direction
            var angle =Math.atan2(accel.y,accel.x)
            ctx.clearRect(0,0,100,100);
            ctx.beginPath();
            ctx.arc(50,50,5,0,2*Math.PI,false);
            ctx.moveTo(50,50);
            // Draw sky direction
            ctx.lineTo(50-50*Math.cos(angle),50+50*Math.sin(angle));
            ctx.stroke();
        }
     </script>
   </head>
   <body>
      <div data-role="page" id = "intropage">
        <div data-role="header">
           <h1>Accelerometer</h1>
        </div>
        <div data-role="content">
           <label for="sliderX">X Acceleration (Roll)</label>
           <input type="range" name="sliderX"id="sliderX"
                  value="0" min="-10"max="10" data-theme="a" />
           <label for="sliderY">Y Acceleration (Pitch)</label>
           <input type="range" name="sliderY"id="sliderY"
                  value="0" min="-10"max="10" data-theme="b" />

           <label for="sliderZ">Z Acceleration (<strike>Yaw</strike>
              Face up/down)
           </label>
           <input type="range" name="sliderZ"id="sliderZ"
                  value="0" min="-10"max="10" data-theme="c" />
        </div>
        <p style = "text-align:center">SKY direction:
                                      follow this line:</p>
        <div style = "text-align:center;margin-top:10px;">
           <canvas id="c" width="100"height="100"></canvas>
        </div>

     </div>
   </body>
</html>
```

MOVE A BALL ON THE SCREEN

Try this example at JsBin : if using a mobile device, use this URL instead!



Thu Feb 14 2013 14:41:51 GMT+0100 (CET)
Device Motion Event
accelerationIncludingGravity: {
x: 0.98920048758645054
y: -0.70648450316963535
z: -9.71372741380404087
}

interval: 0.050000000074505806
spec: http://dev.w3.org/geo/api/spec-source-orientation.html

Code from this example:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
       <meta http-equiv="Content-Type"content="text/html; charset=UTF-8" />
       <meta name="viewport"content="width=device-width,
                                      target-densityDpi=device-dpi,
                                      initial-scale=1.0,
                                      user-scalable=no,
                                      maximum-scale=1.0">
       <title>iOS 4.2 Device Accellerometer</title>
       <style>
          body {
             font-family:Arial, Helvetica,sans-serif;
             font-size: 14px;
          }
          #board {
             position:absolute;
             left:0px;
             right:0px;
             top:0px;
             bottom:0px;
          }
          #ball {
             position:absolute;
             width: 60px;
             height: 60px;
             border-radius: 30px;
             background-image: -webkit-gradient(radial, 45% 45%, 5, 60% 60%,
                  40, from(red), color-
stop(75%, black), to(rgba(255, 255, 255,0)));
             -webkit-box-shadow: 3px 3px 5px#888;
          }
       </style>
       <scriptsrc="http://ajax.googleapis.com/ajax/libs/jquery/1.4.2/jquery.min.js">
       </script>
       <script>
          !window.jQuery &&document.write('<script src="./js/jquery.min.js">
<\/script>')
       </script>
       <script>
          var offset;
          var velocity;
          var board;
          var ball;
          var interval;
          $(document).ready(function() {
              window.addEventListener("devicemotion",onDeviceMotion, false);
              $('#timestamp').html(newDate().toString());
              $('#status').html("Ready!");
              velocity = {};
              velocity.x = 0;
              velocity.y = 0;
              offset = {};
              board = $('#board');
              ball = $('#ball');
              offset.left = (board.width() -ball.width()) / 2;
              offset.top = (board.height() -ball.height()) / 2;
              $('#ball').offset(offset);
              interval =setInterval(updateBall, 25);
          });
          function onDeviceMotion(event) {
             $('#timestamp').html(newDate().toString());
             $('#status').html("Device Motion Event");
             var eventDetails;
             try {
                 var accel =event.accelerationIncludingGravity;
                 eventDetails ="accelerationIncludingGravity: {" +
                 "<br>     x: " + accel.x +
                 "<br>     y: " + accel.y +
                 "<br>     z: " + accel.z +
                 "<br/>} <br/><br/>" +
                 "interval: " +event.interval;
                 updateVelocity(event);
             } catch (e) {
                 eventDetails = e.toString();
             }
             $('#details').html(eventDetails);
          }

          var decay = .9;
          var bounceDecay = .95;
          var maxVelocity = 100;
          function updateVelocity(event) {
             velocity.x +=event.accelerationIncludingGravity.x;
             if (Math.abs(velocity.x) >maxVelocity) {
                 if (velocity.x > 0)velocity.x = maxVelocity;
                 else velocity.x = -maxVelocity;
             }
             velocity.y +=event.accelerationIncludingGravity.y;
             if (Math.abs(velocity.y) >maxVelocity) {
                 if (velocity.y > 0)velocity.y = maxVelocity;
                 else velocity.y = -maxVelocity;
             }
          }
          function updateBall() {
             if (offset.left <= -(ball.width() / 2)) {
                velocity.x =Math.abs(velocity.x * bounceDecay);
             } else if (offset.left >=(board.width() - (ball.width() / 2))) {
                velocity.x = -Math.abs(velocity.x * bounceDecay);
             } else {
                velocity.x =parseInt(velocity.x);
                velocity.x *= decay;
             }
          if (offset.top <= -(ball.height() /2)) {
             velocity.y = -Math.abs(velocity.y * bounceDecay);
          } else if (offset.top >=(board.height() - (ball.height() / 2))) {
             velocity.y = Math.abs(velocity.y* bounceDecay);
          } else {
             velocity.y =parseInt(velocity.y);
             velocity.y *= decay;
          }
          offset.left += velocity.x;
          offset.top -= velocity.y;

          $('#ball').offset(offset);
      }
    </script>
  </head>
  <body>
     <div id="timestamp"></div>
     <div id="status"></div>
     <div id="details"></div>
     <div id="board">
         <div id="ball"></div>
     </div>spec: <ahref="http://dev.w3.org/geo/api/spec-source-
orientation.html"target="http://dev.w3.org/geo/api/spec-source-
orientation.html">http://dev.w3.org/geo/api/spec-source-orientation.html</a>
  </body>
</html>
```

EXTERNAL RESOURCES:

- From the W3C specification: http://dev.w3.org/geo/api/spec-source-orientation.html#devicemotion
- Article on html5Rocks.com about device orientation
- Article on dev.opera.com about device motion and orientation