EECS 1012: Introduction to Computer Science

October 20, 2016

Interacting with the world

- Camera capture
  - Continued from last day
- Other sensors to follow

```html
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript" src="camera.js"> </script>
</head>
<body>
<input type="file" id="userImage" accept="image/* capture="camera"/>
<button onclick="display();">Display</button>
<div id="photo"> </div>
</body>
</html>
```

```javascript
function display() {
  var x = document.getElementById('userImage');
  var f = x.files;
  if(f.length > 0) {
    var reader = new FileReader();
    reader.onload = doDisplay;
    reader.readAsDataURL(f[0]);
  }
}

function doDisplay(e){
  var src = e.target.result;
  var div = document.getElementById("photo");
  var img = document.createElement("img");
  img.src = src;
  if(img.width > img.height) {
    img.style.width ="400px";
    img.style.height = (400 * img.height / img.width) + "px";
  } else {
    img.style.height = "400px";
    img.style.width = (400 * img.width / img.height) + "px";
  }
  div.appendChild(img);
}
```

Gets a reference to the file input type

```
function display() {
  var x = document.getElementById('userImage');
  var f = x.files;
  if(f.length > 0) {
    var reader = new FileReader();
    reader.onload = doDisplay;
    reader.readAsDataURL(f[0]);
  }
}
```

files - the array of files the user choose (or camera picture)

```
function display() {
  var x = document.getElementById('userImage');
  var f = x.files;
  if(f.length > 0) {
    var reader = new FileReader();
    reader.onload = doDisplay;
    reader.readAsDataURL(f[0]);
  }
}
```

if list length not zero

```
function display() {
  var x = document.getElementById('userImage');
  var f = x.files;
  if(f.length > 0) {
    var reader = new FileReader();
    reader.onload = doDisplay;
    reader.readAsDataURL(f[0]);
  }
}
```

FileReader - reads the file

```
function display() {
  var x = document.getElementById('userImage');
  var f = x.files;
  if(f.length > 0) {
    var reader = new FileReader();
    reader.onload = doDisplay;
    reader.readAsDataURL(f[0]);
  }
}
```

When file is loaded, call this **with the file**

```
function display() {
  var x = document.getElementById('userImage');
  var f = x.files;
  if(f.length > 0) {
    var reader = new FileReader();
    reader.onload = doDisplay;
    reader.readAsDataURL(f[0]);
  }
}
```

Start reading the first file

```
function display() {
  var x = document.getElementById('userImage');
  var f = x.files;
  if(f.length > 0) {
    var reader = new FileReader();
    reader.onload = doDisplay;
    reader.readAsDataURL(f[0]);
  }
}
```

get the result (sequence of bytes)

```
function doDisplay(e){
  var src = e.target.result;
  var div = document.getElementById("photo");
  var img = document.createElement("img");
  img.src = src;
  if(img.width > img.height) {
    img.style.width ="400px";
    img.style.height = (400 * img.height / img.width) + "px";
  } else {
    img.style.height = "400px";
    img.style.width = (400 * img.width / img.height) + "px";
  }
  div.appendChild(img);
}
```

create a new img tag

```
function doDisplay(e){
  var src = e.target.result;
  var div = document.getElementById("photo");
  var img = document.createElement("img");
  img.src = src;
  if(img.width > img.height) {
    img.style.width ="400px";
    img.style.height = (400 * img.height / img.width) + "px";
  } else {
    img.style.height = "400px";
    img.style.width = (400 * img.width / img.height) + "px";
  }
  div.appendChild(img);
}
```

style the new ‘img’ and set its src

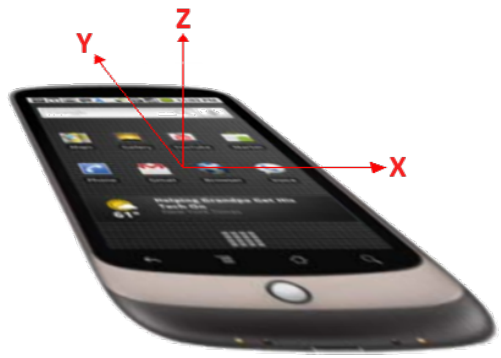
```
function doDisplay(e){
  var src = e.target.result;
  var div = document.getElementById("photo");
  var img = document.createElement("img");
  img.src = src;
  if(img.width > img.height) {
    img.style.width ="400px";
    img.style.height = (400 * img.height / img.width) + "px";
  } else {
    img.style.height = "400px";
    img.style.width = (400 * img.width / img.height) + "px";
  }
  div.appendChild(img);
}
```

and add under div (which was empty to start)

```
function doDisplay(e){
  var src = e.target.result;
  var div = document.getElementById("photo");
  var img = document.createElement("img");
  img.src = src;
  if(img.width > img.height) {
    img.style.width ="400px";
    img.style.height = (400 * img.height / img.width) + "px";
  } else {
    img.style.height = "400px";
    img.style.width = (400 * img.width / img.height) + "px";
  }
  div.appendChild(img);
}
```

Lets look at another measurement of the external world

- Device orientation
  - Most mobile devices provide a mechanism to obtain the orientation of the device
  - Bad news is that there is some variation across devices.



window.addEventListener(‘deviceorientation’, fn);
function fn(e) { … }

alpha - rotation about z
beta - rotation about x
gamma - rotation about y

Some complexities to make this work on different devices

Same with acceleration

```
<!DOCTYPEhtml>
<html>
<head>
<script src="tilt.js" type="text/javascript"> </script>
</head>
<body>
  Alpha <span id="alpha">ALPHA</span><br>
  Beta <span id="beta">ALPHA</span><br>
  Gamma <span id="gamma">ALPHA</span><br>
  X <span id="x"></span><br>
  Y <span id="y"></span><br>
  Z <span id="z"></span><br>
</body>
</html>
```

```
function tilt(e) {
  var z = document.getElementById("alpha");
  z.innerHTML = e.alpha;
  z = document.getElementById("beta");
  z.innerHTML = e.beta;
  z = document.getElementById("gamma");
  z.innerHTML = e.gamma;
}

function motion(e) {
  var z = document.getElementById("x");
  z.innerHTML = e.accelerationIncludingGravity.x;
  z = document.getElementById("y");
  z.innerHTML = e.accelerationIncludingGravity.y;
  z = document.getElementById("z");
  z.innerHTML = e.accelerationIncludingGravity.z;
}

window.addEventListener('deviceorientation', tilt);
window.addEventListener('devicemotion', motion);
```

So lets make an app

- A ‘don’t touch’ application
- Device displays an image, and if moved will update the screen (or play a sound, or …)

So problems to solve…

- Detect motion
  - Know how to do that now
- Sleep until start
  - We have used the timer callback before (think lab 2)

Need to determine a threshold

- So we need to know what the average acceleration is and choose a threshold (say 1.2 x normal acceleration)
  - Need to measure this
- Need to do things differently
  - When the alarm has been triggered
  - When we are in ‘alarm mode’ versus ‘non-alarm mode’

alarmTriggered, do nothing

```
function motion(e) {
  var x = e.accelerationIncludingGravity.x;
  var y = e.accelerationIncludingGravity.y;
  var z = e.accelerationIncludingGravity.z;
  var acc = x * x + y * y + z * z;

  var p  = document.getElementById("debug");
  p.innerHTML = "alarmon " + alarmOn + "<br>";
  if(alarmTriggered) {
    p.innerHTML += "alarm triggered<br>";
  } else {
    if(alarmOn) {
      p.innerHTML += "threshold " + threshold + " acc " + acc + "<br>";
      alarmTriggered = acc > threshold;
    } else {
      suma = suma + acc;
      sumn = sumn + 1;
    }
  }
}
```

alarmTriggered, is false

```
function motion(e) {
  var x = e.accelerationIncludingGravity.x;
  var y = e.accelerationIncludingGravity.y;
  var z = e.accelerationIncludingGravity.z;
  var acc = x * x + y * y + z * z;

  var p  = document.getElementById("debug");
  p.innerHTML = "alarmon " + alarmOn + "<br>";
  if(alarmTriggered) {
    p.innerHTML += "alarm triggered<br>";
  } else {
    if(alarmOn) {
      p.innerHTML += "threshold " + threshold + " acc " + acc + "<br>";
      alarmTriggered = acc > threshold;
    } else {
      suma = suma + acc;
      sumn = sumn + 1;
    }
  }
}
```

alarm mode, wait for acceleration to exceed threshold

```
function motion(e) {
  var x = e.accelerationIncludingGravity.x;
  var y = e.accelerationIncludingGravity.y;
  var z = e.accelerationIncludingGravity.z;
  var acc = x * x + y * y + z * z;

  var p  = document.getElementById("debug");
  p.innerHTML = "alarmon " + alarmOn + "<br>";
  if(alarmTriggered) {
    p.innerHTML += "alarm triggered<br>";
  } else {
    if(alarmOn) {
      p.innerHTML += "threshold " + threshold + " acc " + acc + "<br>";
      alarmTriggered = acc > threshold;
    } else {
      suma = suma + acc;
      sumn = sumn + 1;
    }
  }
}
```

not in alarm mode, figure out acceleration

```
function motion(e) {
  var x = e.accelerationIncludingGravity.x;
  var y = e.accelerationIncludingGravity.y;
  var z = e.accelerationIncludingGravity.z;
  var acc = x * x + y * y + z * z;

  var p  = document.getElementById("debug");
  p.innerHTML = "alarmon " + alarmOn + "<br>";
  if(alarmTriggered) {
    p.innerHTML += "alarm triggered<br>";
  } else {
    if(alarmOn) {
      p.innerHTML += "threshold " + threshold + " acc " + acc + "<br>";
      alarmTriggered = acc > threshold;
    } else {
      suma = suma + acc;
      sumn = sumn + 1;
    }
  }
}
```

alarm mode set, disable

```
function pressed() {
  var p = document.getElementById("button");
  if(alarmOn) {
    alarmOn = false;
    suma = 0;
    sumn = 0;
    threshold = 0;
    alarmTriggered = false;
    p.innerHTML ="ENABLE ALARM";
  } else { // arm the alarm
    alarmOn = true;
    threshold = 1.2 * suma / sumn;
    alarmTriggered = false;
    p.innerHTML = "DISABLE ALARM";
  }
}
```

alarm mode not set, enable

```
function pressed() {
  var p = document.getElementById("button");
  if(alarmOn) {
    alarmOn = false;
    suma = 0;
    sumn = 0;
    threshold = 0;
    alarmTriggered = false;
    p.innerHTML ="ENABLE ALARM";
  } else { // arm the alarm
    alarmOn = true;
    threshold = 1.2 * suma / sumn;
    alarmTriggered = false;
    p.innerHTML = "DISABLE ALARM";
  }
}
```