

IT452 Advanced Web and Internet Systems

Set 7: Events and Animation

Animation and Events

- Animation – attention, explanation
- Capturing user events
- Education and games
- We will use jQuery to make coding easier

Example game: math functions
“swath of destruction”

Preliminaries

```
<style type="text/css">
  /* Margin and padding on body element can introduce errors in determining
     element position and are not recommended; we turn them off. */

  body { margin:0; padding:0 }

  #foo { width:10px; height:10px; background-color:#00f }
</style>

<script type="text/javascript"
  src="http://ajax.googleapis.com/ajax/libs/jquery/1.7/jquery.min.js"></script>
```

falling.html – An event loop

```
// This function is periodically called by the timer.  It makes things move
function mainLoop() {
    moveBlock();
}

// (x,y) coordinates of the block
var blockPos = [200,200];

// Move the block by a little bit
function moveBlock() {
    blockPos[0] = blockPos[0] + 1;
    blockPos[1] = blockPos[1] + 2;
    $('#foo').offset({ left: blockPos[0], top: blockPos[1] });
}

// Do necessary setup on first load
function init() {
    // Call main loop every 50 milliseconds
    setInterval("mainLoop()", 50);

    // On every click, call the move() function
    $(document).click(moveOnClick);
}

// When we click, move box to that location
function moveOnClick(e) {
    blockPos[0] = e.pageY;
    blockPos[1] = e.pageX;
    // Move the block to the click point
    $('#foo').offset({ left: blockPos[0], top: blockPos[1] });
}
```

falling.html (HTML portion)

```
<body>
```

```
    <h1>Using an event loop</h1>
```

```
    <!-- The actual block that moves around -->
```

```
    <div id="foo">
```

```
    </div>
```

```
    <!-- The main game area -->
```

```
    <div style="width:500px; height:200px; background-color:red">
```

```
    </div>
```

```
</body>
```

jQuery Event Listeners

1. Use a selector to choose the object
2. Decide which event type to use
 1. Click
 2. Mouseover
 3. Blur
 4. Keypress
 5. Etc.
3. Add a function reference or anonymous function

How do we add a key listener to the div?

```
$('#bigdiv').keypress(handleClick);
```

bounce.html (1)

```
var bigdivPos  = [100,100];
var bigdivSize = [500,200]

function moveBlock() {
    // Get the block's current position.
    blockPos[0] += blockVeloc[0];
    blockPos[1] += blockVeloc[1];
    // Move the block to its new position.
    $('#foo').offset({ left: blockPos[0], top: blockPos[1] });

    keepItemInsideBox(bigdivPos, bigdivSize, blockPos, blockSize, blockVeloc);
}

function init() {
    // Save the main container's rendered position on the window.
    var offset = $('#bigdiv').offset();
    bigdivPos[0] = offset.left;
    bigdivPos[1] = offset.top;

    // Call main loop every 50 milliseconds
    setInterval("mainLoop()", 50);

    // On every click, call the move() function
    $(document).click(moveOnClick);
}
```

bounce.html (2)

```
// Checks to see if the block has hit the edge of the bigdiv. If so, bounce off
// If we get outside the box, this will also force us to move back in.
function keepItemInsideBox(boxPos, boxSize, itemPos, itemSize, itemVeloc)  {

    // Check to see if we hit a vertical edge
    var edgeHit = calcVerticalEdgeHit(boxPos, boxSize, itemPos, itemSize);
    if ( (edgeHit == HIT_LEFT) && (itemVeloc[0] < 0) ) {
        // if too far left, make sure we go right
        itemVeloc[0] *= -1;
    }
    if ( (edgeHit == HIT_RIGHT) && (itemVeloc[0] > 0) ) {
        // if too far right, make sure we go left
        itemVeloc[0] *= -1;
    }

    // Check to see if we hit a vertical edge
    var edgeHit = calcHorizontalEdgeHit(bigdivPos, bigdivSize, blockPos, blockSize);
    if ( (edgeHit == HIT_TOP) && (itemVeloc[1] < 0) ) {
        // if too far up, make sure we go down
        itemVeloc[1] *= -1;
    }
    if ( (edgeHit == HIT_BOTTOM) && (itemVeloc[1] > 0) ) {
        // if too far down, make sure we go up
        itemVeloc[1] *= -1;
    }

}
```

bounce.html (3)

```
// See if the item hit a vertical edge by going too far left or right.  
// NOTE: we assume it was basically inside the box to begin with  
function calcVerticalEdgeHit(boxPos, boxSize, itemPos, itemSize) {  
    // check for hit on left side  
    var boxLeft    = boxPos[0];  
    var boxRight   = boxPos[0] + boxSize[0];  
    var itemLeft   = itemPos[0];  
    var itemRight  = itemPos[0] + itemSize[0];  
  
    if (itemLeft < boxLeft)  
        return HIT_LEFT;  
    else if (itemRight > boxRight)  
        return HIT_RIGHT;  
}  
  
// See if the item hit a horizontal edge by going too far up or down.  
// NOTE: we assume it was basically inside the box to begin with  
function calcHorizontalEdgeHit(boxPos, boxSize, itemPos, itemSize) {  
    // check for hit on left side  
    var boxTop     = boxPos[1];  
    var boxBot     = boxPos[1] + boxSize[1];  
    var itemTop    = itemPos[1];  
    var itemBot    = itemPos[1] + itemSize[1];  
  
    if (itemTop < boxTop)  
        return HIT_TOP;  
    else if (itemBot > boxBot)  
        return HIT_BOTTOM;  
}
```

Multiblocks.html

```
// (x,y) coordinates of the block
var block1Pos    = [200,200];
var block1Size   = [10,10]
var block1Veloc = [1, 2];

var block2Pos    = [300,300];
var block2Size   = [10,10]
var block2Veloc = [2, -1];

function mainLoop() {
    moveBlock('block1', block1Pos, block1Size, block1Veloc);
    moveBlock('block2', block2Pos, block2Size, block2Veloc);
}

// Move the block by a little bit
function moveBlock(name, pos, size, veloc) {
    pos[0] = pos[0] + veloc[0];
    pos[1] = pos[1] + veloc[1];
    // Move the block to its new position.
    $("#" + name).offset({ left: pos[0], top: pos[1] });

    keepItemInsideBox(bigdivPos, bigdivSize, pos, size, veloc);
}

...
<body class="yui-skin-sam" >

<h1>Using an event loop</h1>

<!-- The actual block that moves around -->
<div id="block1" class="myblock"></div>
<div id="block2" class="myblock"></div>
```

Exercise

1. How would you make the blocks twice as FAST?

Make setTimeout 25ms instead of 50

2. How else can the blocks move twice as FAST?

Double the
velocity

3. How would you make the blocks twice as BIG?

CSS

Game.html (1)

```
// missile launcher
var launcherPos  = [350,300];
var launcherSize = [ 20, 20]

// a missile.  Put off board initially
var offBoardPos  = [800, 0];
var missilePos   = offBoardPos;
var missileSize  = [50,50];
var missileVeloc = null;

// Do necessary setup on first load
function init() {
    // Save the main container's rendered position on the window.
    var offset = $('#bigdiv').offset();
    bigdivPos[0] = offset.left;
    bigdivPos[1] = offset.top;

    // Call main loop every 25 milliseconds
    setInterval("mainLoop()", 25);

    // On every click, call the fire function
    $(document).click(fireAtClick);

    // Create the blocks
    createDiv(block1Pos, block1Size, "myblock", "block1");
    createDiv(block2Pos, block2Size, "myblock", "block2");

    // Create the missile launcher
    createDiv(launcherPos, launcherSize, "launcher", "launcher");

    // missile initially invisible
    createDiv (missilePos, missileSize, "invisible", "missile1");
}
```

Game.html (2)

```
// Create a div with the given pos, size, and class
function createDiv(pos, size, myclass, myid) {      Create by string
    // create basic div
    var newobj = $("<div id=''" + myid + "'> </div>");

    $('body').append(newobj);

    newobj.addClass(myclass).css("width",size[0]+"px").css(
    "height",size[1]+"px");                                Chaining!!!

    // move it to right place
    newobj.offset({ left: pos[0], top: pos[1] });

}

// Update the score!
function incrementScore(val) {
    score = score + val;
    $('#scorebox').text("Score: "+score);
}
```

Create by string
HTML!

Chaining!!!

Text update
sooo easy!

Game.html(3)

```
// Check for a missile hit.  If hit, act like block destroyed, get rid of missile
function checkMissileHit(missileId, missilePos, missileSize, blockPos, blockSize) {
  if (!didSpheresHit(missilePos, missileSize, blockPos, blockSize))
    return;

  // we did have a hit.  Make the missile invisible and immobile
  $('#missile1').addClass("invisible");
  $('#missile1').removeClass("missile");
  missileVeloc = null;
  missilePos = offBoardPos;
  $('#'+missileId).offset({ left: missilePos[0], top: missilePos[1] });
  incrementScore(100);

  // Randomize location of new block (somewhere inside bigdiv)
  blockPos[0] = bigdivPos[0] + Math.random() * bigdivSize[0];
  blockPos[1] = bigdivPos[1] + Math.random() * bigdivSize[1];
}
```

NOTE addClass and
removeClass, can have
multiple classes on an
object

Game.html(3)

```
// Check if the spheres with given pos and sizes are touching.  
// Assumes width equal to height  
function didSpheresHit(pos1, size1, pos2, size2) {  
    // We have positions of top left corners. Convert to center positions  
    var cx1 = pos1[0] + size1[0] / 2;  
    var cx2 = pos2[0] + size2[0] / 2;  
    var cy1 = pos1[1] + size1[1] / 2;  
    var cy2 = pos2[1] + size2[1] / 2;  
  
    // find distance between centers  
    var dx = cx1 - cx2;  
    var dy = cy1 - cy2;  
    dist = Math.sqrt(dx*dx + dy*dy);  
  
    // size tracks the width, so radius would be half of that.  
    // So they touch if dist less than (width1 + width2) / 2  
    if (dist < ( (size1[0] + size2[0]) / 2) )  
        return true;  
    else  
        return false;  
}
```

Game.html (4)

```
// When we click, fire a missile at that location
function fireAtClick(e) {
    // calculate slope toward that spot
    var dx = e.pageX - launcherPos[0];
    var dy = e.pageY - launcherPos[1];

    // scale down by the overall distance.  not necc. the right math, but
    // gets it moving in the right direction
    var dist = Math.sqrt(dx*dx + dy*dy);
    dx = 2 * dx / dist;
    dy = 2 * dy / dist;

    // Make coordinates for the missile
    missileVeloc = [ dx , dy ];
    // get COPY of that array, not the same one
    missilePos    = [ launcherPos[0], launcherPos[1] ];

    // Make missile visible.
    $('#missile1').addClass("missile");
    $('#missile1').removeClass("invisible");
}
```

Further Example

- game-keypress.html – add keyboard movement
(NOTE: array keys more complex, ask if needed)

JavaScript Objects

JS Objects: alternative syntax

```
// Create the game board ("bigdiv")
bigdiv = new Object();
bigdiv.pos  = [100, 100];
bigdiv.size = [500, 200];
bigdiv.elem = createDiv(bigdiv.pos, bigdiv.size, "class_bigdiv");
```

OR

```
// Create the game board ("bigdiv")
bigdiv = {
  pos:  [100, 100],
  size: [500, 200]
};
bigdiv.elem = createDiv(bigdiv.pos, bigdiv.size, "class_bigdiv")
```

Note: {} makes empty object. [] makes empty array.

Game Design

- Think about the key elements

- Mechanics
 - Story
 - Aesthetics
 - Technology

Example game: math functions
“swath of destruction”

- JS/jQuery speed may be an issue

- Test early and often!
 - Design and test

- Start thinking about your game!

Zombie attack?