Notes 16: MovingMC Class

In the proceeding example we had an array of balls but only one set of velocities. We could have separate velocities variables for each, but what if we have hundreds of balls? We could have an array of velocities, and an array of balls, but it probably makes more sense to have Moving MovieClip Class and then create an array of MovingMC objects. With each object we can associate relevant members and functions. Now that we know about depths we understand the need to guarantee unique names and depths, we can do that with this class also.

The following file contains a class definition:

MovingMC.as

Look at the code. A few highlights:

Members include most things you would want:

private var x:Number ;
private var y:Number ;   // current x,y location
private var xv:Number ;   // x velocity
private var yv:Number ;   // y velocities
private var width:Number ;
private var height:Number ;
private var internalName:String ;
private var depth:Number ;
private var mc:MovieClip ;

Now look at the constructor. Look how it uses getNextHighestDepth() to ensure unique internal names and depths:

    // constructor
    function MovingMC(x:Number, y:Number, xv:Number, yv:Number,
                       linkName:String) {
        var nextDepth:Number = _root.getNextHighestDepth() ;
        internalName = "mc" + nextDepth ;
        mc = _root.attachMovie(linkName,internalName,nextDepth) ;
        setX(x) ;
        setY(y) ;
        setXV(xv) ;
        setYV(yv) ;
    }
The class has all the usual get/set methods, but it also has updateX() and updateY() methods to make it easier to use:

```java
public function updateX() : Void {this.x += this.xv ; this.mc._x = this.x ; }
public function updateY() : Void {this.y += this.yv ; this.mc._y = this.y ; }
```

Given this class, lets use it to create a bunch of bouncing balls. Run the following (F12) and also look at the code. Note, you MUST have MovingMC in the same directory as this .fla.

**a_objectArray_2.fla**

```javascript
var numBalls:Number = 5 ;
var balls:Array = new Array() ; // an array of Moving MC object

var tempX:Number ;
var tempY:Number ;
var tempXV:Number ;
var tempYV:Number ;
var MAX_XV:Number = 8 ;
var MAX_YV:Number = 12 ;

for (var i = 0 ; i < numBalls ; i++) {
    tempX = Math.floor( Math.random() * (Stage.width - 100) ) ;
    tempY = Math.floor( Math.random() * (Stage.height - 100) ) ;
    tempXV = ( Math.random() * MAX_XV - (MAX_XV / 2) ) ;
    tempYV = ( Math.random() * MAX_YV - (MAX_YV / 2) ) ;
    balls[i] = new MovingMC(tempX,tempY,tempXV,tempYV,"yellowBall")
;
}

onEnterFrame = function() {
    for (var i = 0 ; i < numBalls ; i++) {
        balls[i].updateX() ;
        balls[i].updateY() ;

        if (balls[i].getY() > (Stage.height-balls[i].getHeight()))
            balls[i].setYV( balls[i].getYV() * -1) ;
        if (balls[i].getY() < 0)
            balls[i].setYV( balls[i].getYV() * -1) ;
        if (balls[i].getX() > (Stage.width-balls[i].getWidth()))
            balls[i].setXV( balls[i].getXV() * -1) ;
    }
}
```
if (balls[i].getX() < 0)
    balls[i].setXV( balls[i].getXV() * -1 )

That is it, the entire code. The MovingMC code makes sure one does not goof up the
names. Now, there is nothing to stop you from adding code using attachMovie() that
overwrites one of these balls, but if you follow the model of using
getNextHighestDepth() to create the name and depth, you will be fine.