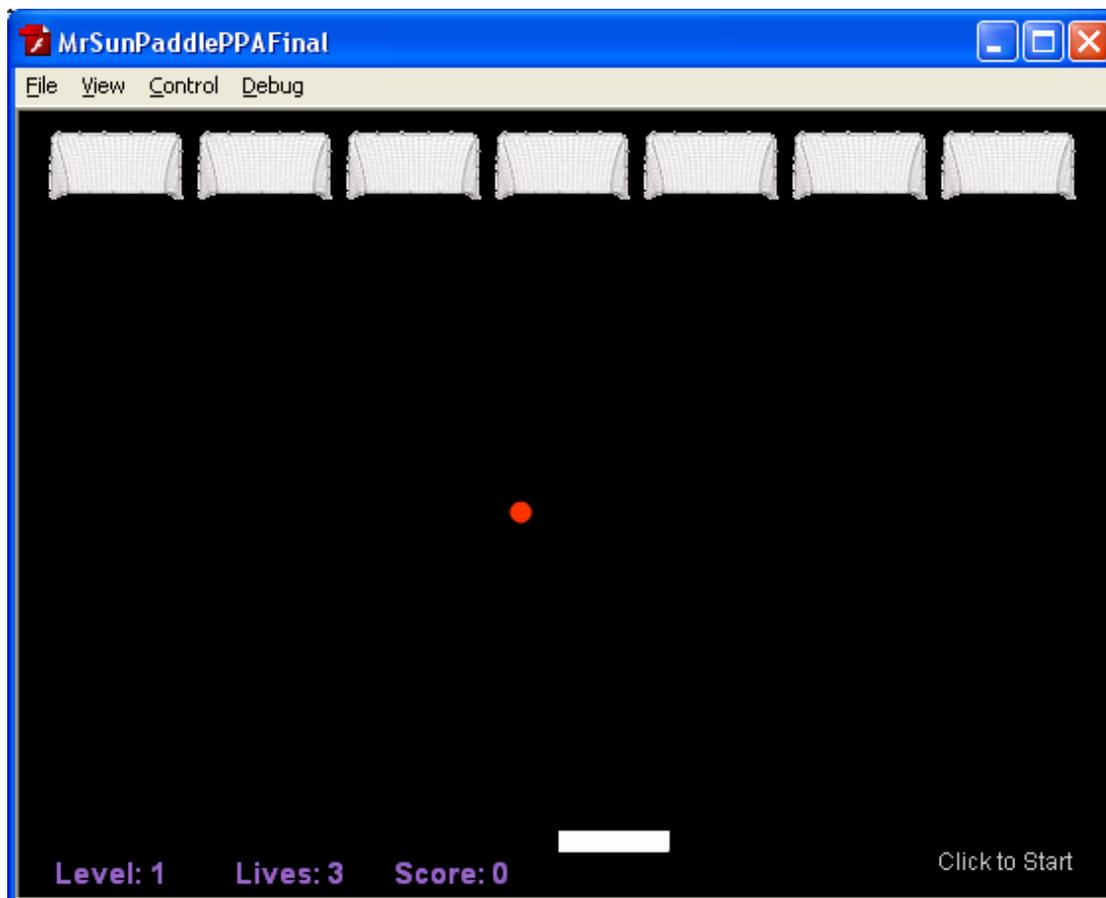


Flash Paddle Ball Game (Converted later into Soccer)

In Paddle Ball, the player controls a paddle at the bottom of the screen, which he or she can move left and right with the mouse. The main active element in the game is a ball, which can bounce off the side, walls, and top, but passes through the bottom of the screen if the paddle isn't there.

Here is what a finished Paddle Ball “Break the Bricks” game looks like:



Mr Sun's Flash blog is a superb site well worth visiting. It has detailed step by step instructions on how to make the above Paddle Ball Game at:

<http://www.blogcatalog.com/blog/mr-sun-studios>

Go to the above Mr Sun web page and then search at top right hand corner search box on “Brick Breaker Game AS 3”. It should then find his tutorial. Note that we actually have to click on the heading of the first listed item: “Create a Brick Breaker Game AS 3” to be taken into the actual tutorial. Once we are in the Tutorial, all parts are available by clicking at the top of page links.

Note also that “Mr Sun” has an AS 2 and AS 3 version of the game, so it is important to be on the AS 3.0 sets of instructions for what we are doing.

Flash Paddle Ball Game (Converted later into Soccer)

So the first project is to build the “Brick Breaker” Game by following Kenny Sun’s excellent multi-part set of instructions.

What follows here, are some notes from our experience of doing the build.

Then later on in this document, we discuss how to change the completed “Brick Breaker” game into a soccer goals game.

Note that as we were building Mr Sun’s project, our version of the Game was really slow, and so we had to increase the Flash document’s frames per second from fps = 12 to fps = 20.

Instead of using Mr Sun’s 70 pixels wide x 15 high “Brick”, as the top of screen target, we found a soccer goal image on the web to use.

We then took the image into Adobe Fireworks to make the background transparent. We also resized the soccer goal image to be 70 wide and 35 high**. ** This height value was determined by Fireworks, since we resized with “Constrain Proportions” ticked.

Later on when we made the bricks two and three rows thick for “levels”, we might have problems, because our soccer goal height is 35, but Mr Sun’s AS 3.0 code is based on 15 pixel high “bricks”. With his code, our Row 2 Game Level 2 set of Goals overlap with the Level 1 Goals.

We therefore had to alter his code, in the “makeLvl” function y coordinate like this:

```
if(lvlArray[currentLvl-1][i] == 1){
    //creating a variable which holds the brick
instance
    // The brick (Goal) is set up as a Class with its
own
    // separate Brick.AS file.
    var brick:MovieClip = new Brick();
    //setting the brick's coordinates via the i
variable and brickRow
    brick.x = 15+(i-brickRow*7)*75;
    brick.y = 10+brickRow*40;
```

We found Part 5 of the Game the hardest to build, because at the end of a successful game tested in Flash it actually gives several errors like :
TypeError: Error #1010: A term is undefined and has no properties.
BUT

If we download Mr Sun’s Zip file and play his Part 5 game, it gives this same error... and so actually nothing is wrong with our own Part 5 build ☺

(A similar thing happens at the end of a losing game)

Flash Paddle Ball Game (Converted later into Soccer)

HOWEVER, these errors only display when we Ctrl-Enter test the game while editing our .Fla file. If we are playing the exported .swf file, everything is a-ok, and the game ends on the screen quite ok.

We also had problems with Part 5 because we left the code for the beginCode function as the old Part 4 function like this :

```
function beginCode():void {
```

and this gave us the run time error of:

```
ArgumentError: Error #1063: Argument count mismatch on  
MrSunPaddlePPA_fla::MainTimeline/beginCode(). Expected 0, got 1.
```

BUT

Because an Event Listener has been added in Part 5 for a mouse click to start the game, we need to have the beginCode function like coded this in part 5:

```
function beginCode(event:MouseEvent):void{
```

Mr Sun did have this change mentioned in his Part 5 instructions, but we must have been a bit careless and missed making this important change.

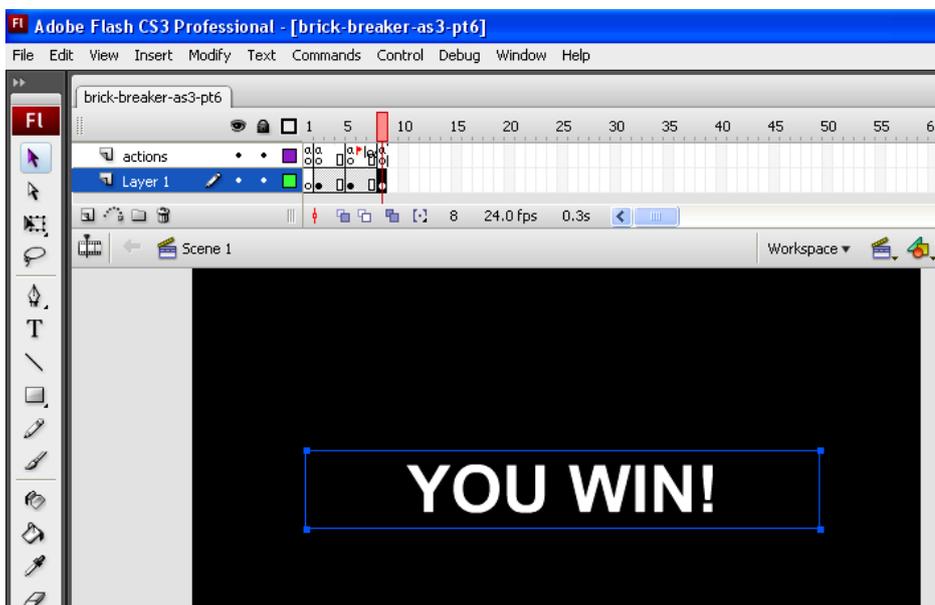
We eventually found what was wrong by comparing his solution code directly with our code, line by line.

A few “WTH” moments were also had while editing in the Actionscript window, when for reasons unknown, the panel went full size. After this happened, we could not scroll down to the bottom of the code anymore. The fix for this problem was to simply do **Window > Workspace > Default**.

Part 6 of Mr Sun’s Tutorial

Part 6 of the Tutorial does not give a lot of details, and so it is probably best to download Mr Sun’s solution zip files. This download link is at the bottom of the tutorial. Save the zip files and use WinRAR to unzip them.

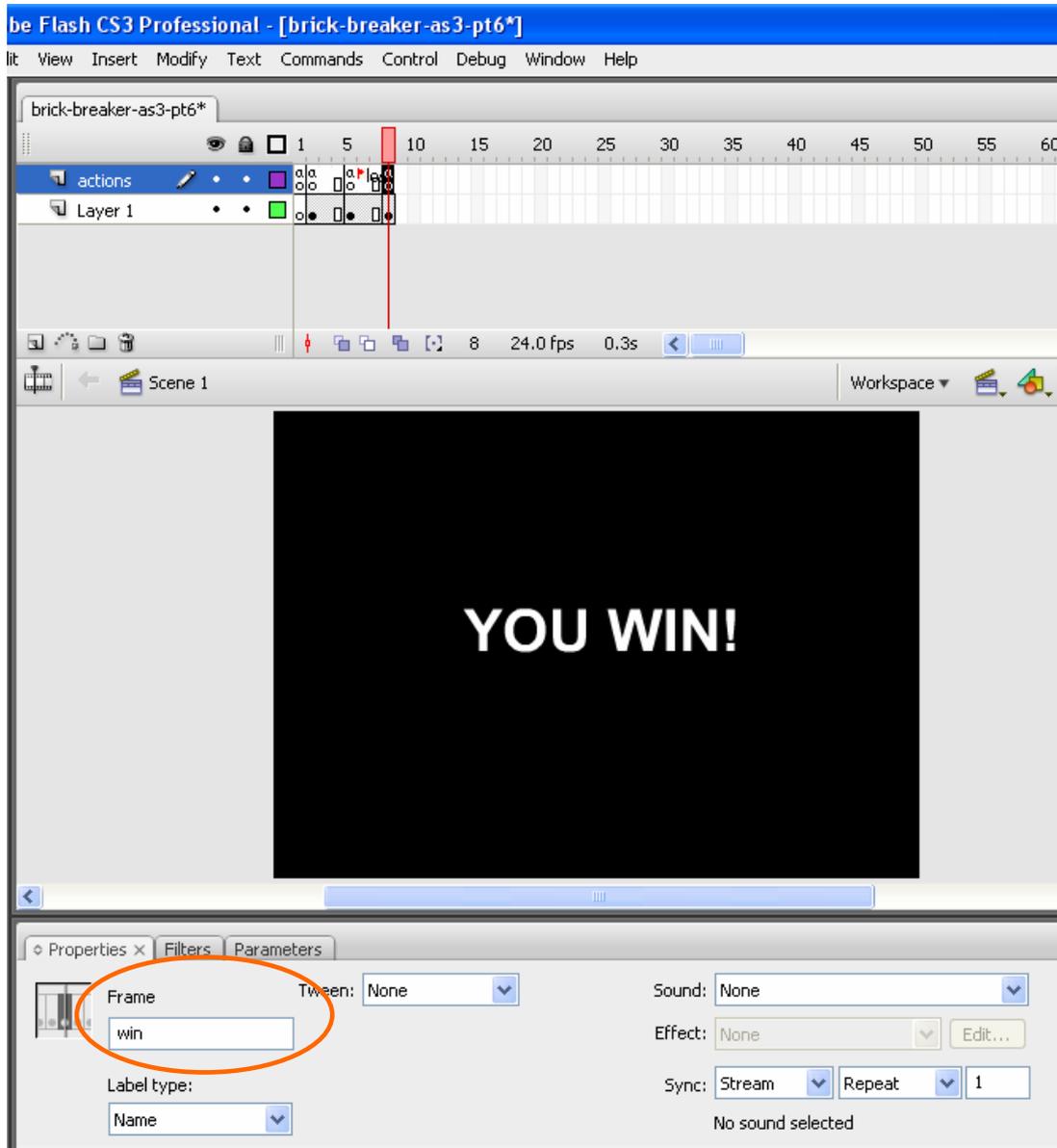
Part 6 adds a Winner Splash Screen at frame 8, and so we end up with this:



Flash Paddle Ball Game (Converted later into Soccer)

In frame 8 (of the Actionscript Layer) we have to make sure that we label the frame as “win” so that it can connect up to the Actionscript code.

Eg. The setup is like this:



In frame 8 win on the Actionscript Layer, we then put the following lines of code: to display the score, and to wait for a click to restart a new game

```
//Display the player's final score  
txtScore.text = "Final Score: "+score;  
// wait for a mouse click to start a new game  
stage.addEventListener(MouseEvent.CLICK, resetGame);
```

Flash Paddle Ball Game (Converted later into Soccer)

Part 6 Changes to Frame 2 Main Game Actionsript

The first lot of code we add into the existing Frame 2 Actionsript is the Dynamic text “Click Here to Start” for the start of each level of the Game. We have to build a dynamic text field onto our stage for the text to go in; and this is explained quite well in Mr Sun’s instructions.

At the end of the beginCode function we add in:

```
//Remove the "Click to Start" Text from the screen  
txtStart.text = '';
```

Add the dynamic text setting in the existing “checkLevel” function, whenever we start a new level, as shown in blue below:

```
function checkLevel(event:Event):void{  
    //checking if the bricks are all gone  
    if(brickAmt == 0){  
        //reset the level by increasing the level  
        currentLvl ++;  
        //and re-running makeLvl  
        makeLvl();  
        //set the dynamic text field words  
        txtStart.text = "Click to Start";  
        //then reset the ball's and paddle's position  
        mcBall.x = 150;
```

Finally, set the dynamic text at the end of our frame 2 AS code by adding in the code shown in bold blue below:

```
//Wait until the mouse clicks, before beginning the game  
stage.addEventListener(MouseEvent.CLICK, beginCode);  
// Set up the bricks for the level the player is on  
makeLvl();  
//set the dynamic text field words  
txtStart.text = "Click to Start";
```

We need to add a variable with the others at the start of the AS code in frame 2 to store the score like this:

```
//Store the score of the game  
var score:int = 0;
```

In the makeLvl function, we need to add the code shown in bold blue below:

```
function makeLvl():void{ //Places bricks onto Level  
    //check if indeed there are any more levels left  
    if(currentLvl > lvlArray.length){  
        //the game is over now and we survived all the levels  
        //and have therefore won the Game.  
        gameOver = true;
```

Flash Paddle Ball Game (Converted later into Soccer)

```
//remove all the Game Play Event listeners
mcPaddle.removeEventListener(Event.ENTER_FRAME, movePaddle);
mcBall.removeEventListener(Event.ENTER_FRAME, moveBall);
removeEventListener(Event.ENTER_FRAME, checkLevel);
removeEventListener(Event.ENTER_FRAME, updateTextFields);
//go to the won the game frame
gotoAndStop("win");
}
//finding the array length of the lvl code
//The index has to be currentLvl-1 because:
//array indexes start on 0 and our lvl starts at 1
//our level will always be 1 higher than the actual index of the array
var arrayLength:int = lvlArray[currentLvl-1].length;
//the current row of bricks we are creating
var brickRow:int = 0;
```

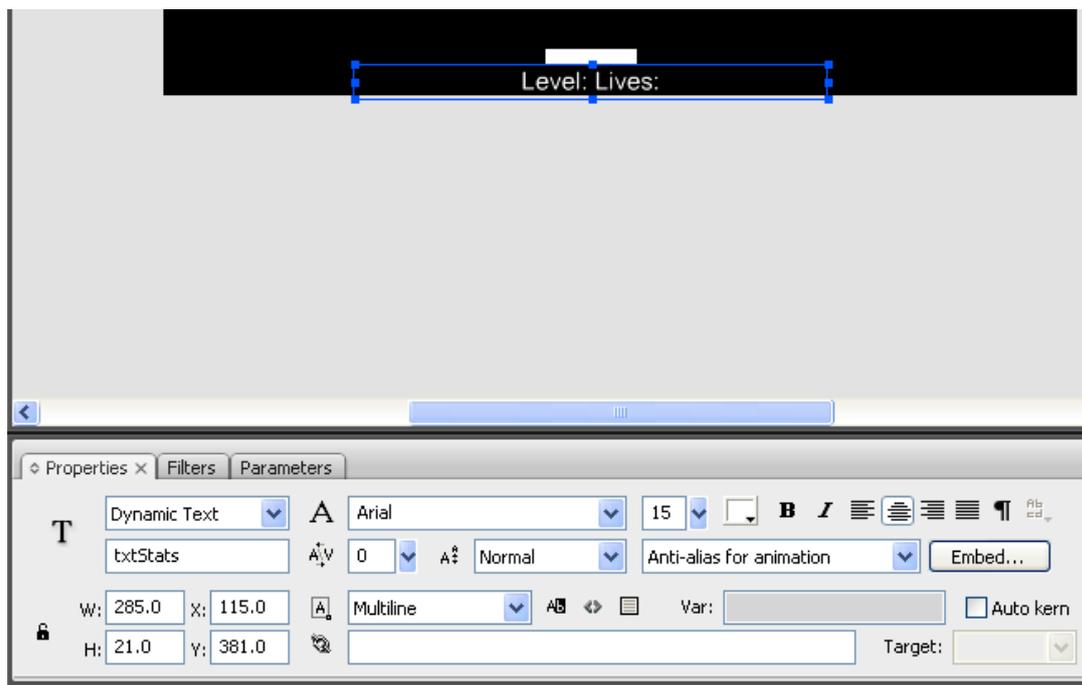
We add a new function near the end of our code (under the existing checkLevel function) that is coded like this:

```
function updateTextFields(event:Event):void{
    txtStats.text = "Level: "+currentLvl+" Lives: "+lives+" Score:
    "+score;
}
```

We then add an event listener to go with this function, in the bottom area of our AS code like this:

```
//if the mouse clicks, then begin the game
stage.addEventListener(MouseEvent.CLICK, beginCode);
//setting the text's word
txtStart.text = "Click To Begin";
//Update the text fields as we pass through each frame of the game
addEventListener(Event.ENTER_FRAME, updateTextFields);
//making the level
makeLvl();
```

We also have to add a dynamic text box to the bottom of our frame 2 stage that is setup like this:



Flash Paddle Ball Game (Converted later into Soccer)

Part 6 Changes to “Brick.AS” Actionscript

We have to update the score in the Brick (Soccer Goal) Class code in “Brick.AS”, since we score point each time we hit a brick (Kick a Goal) .

The new code we have to add is shown in Blue :

```
private function beginClass(event:Event):void{
    //defining _root as the document root
    _root = MovieClip(root);
    //incrementing how many bricks are on the stage
    _root.brickAmt ++;
}
private function enterFrameEvents(event:Event):void{
    //checking if the player has lost
    if(_root.gameOver){
        //destroy this brick
        this.parent.removeChild(this);
        //stop running this code
        removeEventListener(Event.ENTER_FRAME,
enterFrameEvents);
    }
    //hit testing with the ball
    if(this.hitTestObject(_root.mcBall)){
        //making the ball bounce off vertically
        _root.ballYSpeed *= -1;
        //destroying this brick
        this.parent.removeChild(this);
        //stop running this code
        removeEventListener(Event.ENTER_FRAME,
enterFrameEvents);

        //decrementing the amount of bricks on stage
        _root.brickAmt --;
        //increasing the score
        _root.score += 10;
    }
}
}
```

Part 6 Changes to Frame 5 Actionscript

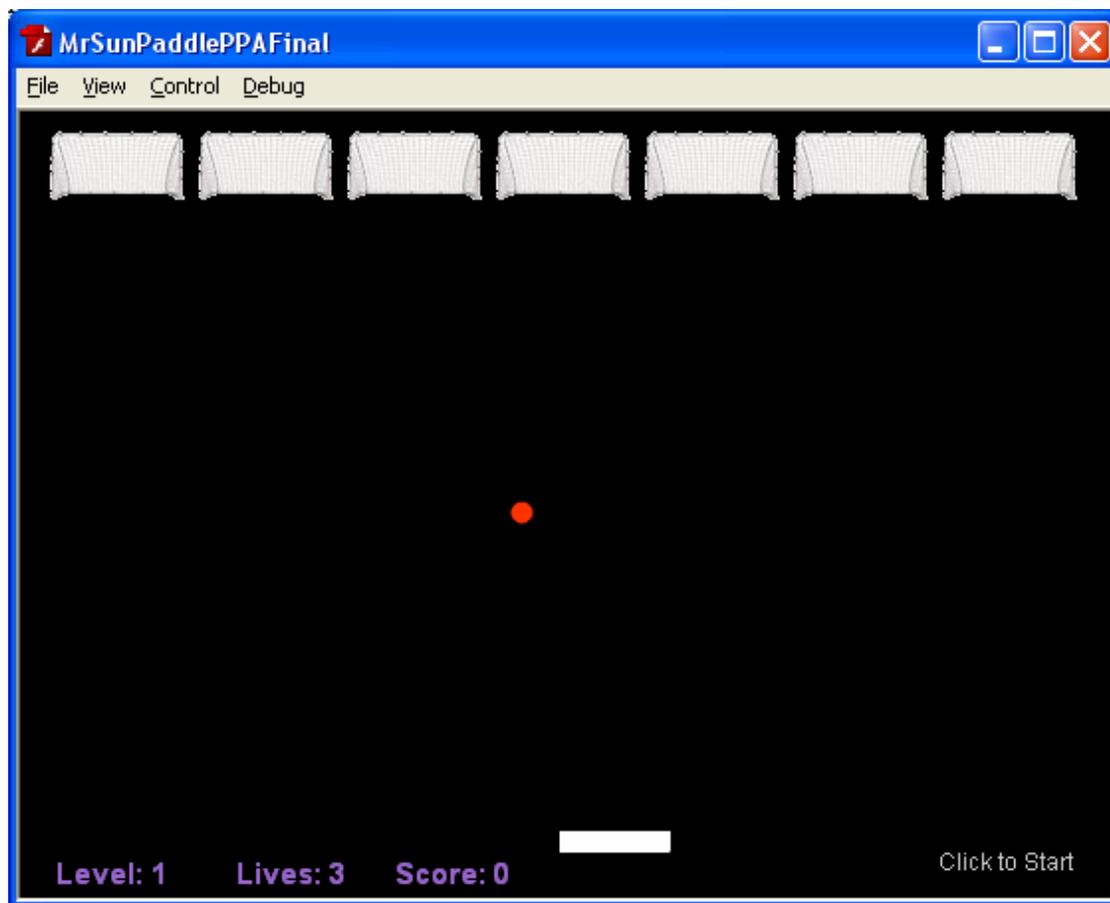
At the bottom of the Actionscript code for the “Game Over” You Lose in frame 5, we add the following code: (Add it on its own after the function)

```
//Display the final score
txtScore.text = "Final Score: "+score;
```

and to display this text, we will need to add a dynamic text box onto the frame 5 stage, and give it an instance name of “txtScore” .

Flash Paddle Ball Game (Converted later into Soccer)

Here is a printscreen of the final paddle game we made by following Mr Sun's tutorial: (except that we used slightly taller soccer goal images instead of rectangle hand drawn bricks).



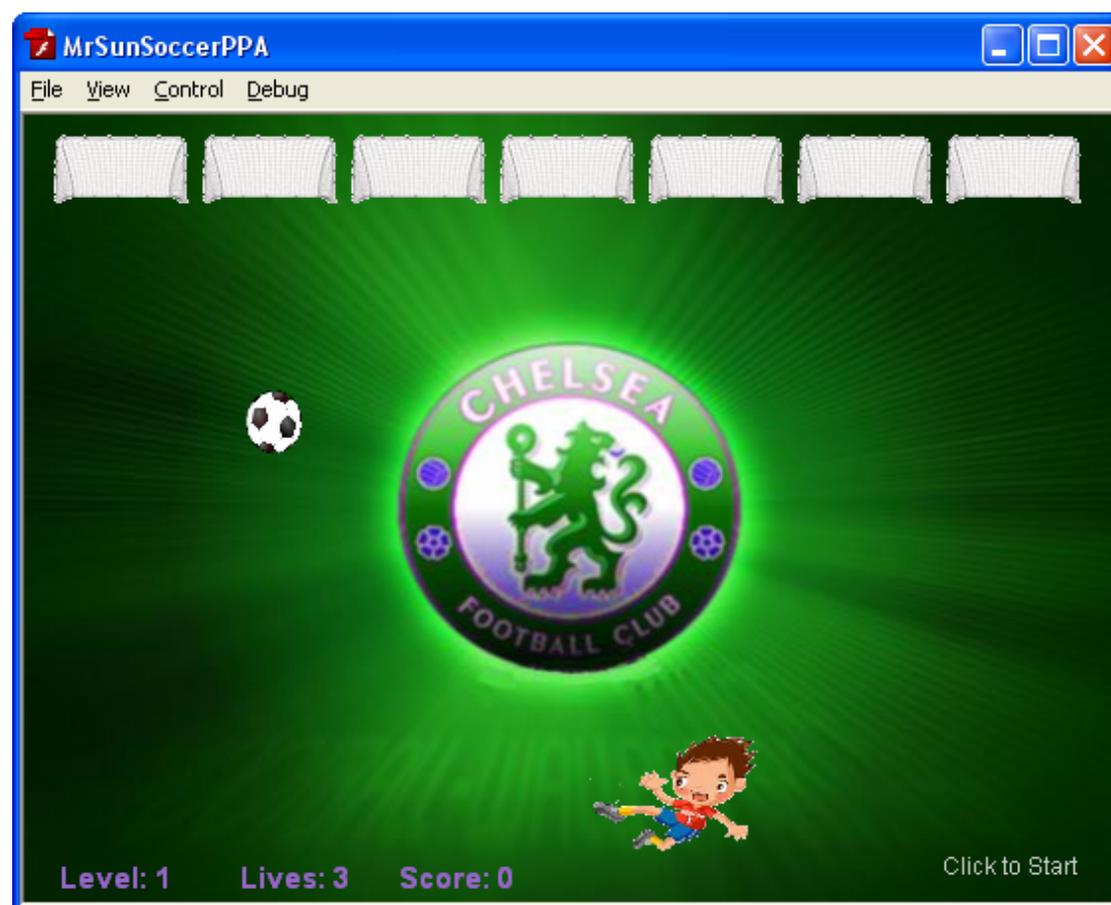
Converting the Paddle Game to a Soccer Game.

This was quite simple, as we only changed the “cosmetics” of the game, and did not alter the Actionscript 3.0 “game engine” at all.

First we searched Google images on the net and found a “Chelsea” soccer club desktop that we altered and reduced down to a 550x400 background, using Adobe Fireworks. We then placed this background on a new bottom layer in our Flash document. This background resides in frame 2 but had to be key frame (F6) copied, and then deleted out of frame 5. That way we can retain the plain black background for the Game Over screens.

We also replaced the paddle with a boy, and the ball with a soccer ball (both from the net, and adjusted in Fireworks). This basically gave us the finished soccer game shown below.

Flash Paddle Ball Game (Converted later into Soccer)



Playing the Finished Game

There is one game “glitch” we have found. And that is when we hit the ball straight up against the left or right side wall. The ball wobbles up and down along the left wall edge back and forth forever, and we cannot redirect its course. Eventually, we have to move away and let the ball hit the bottom of screen and consume all our lives so that the game can end. In a way this is kind of ok, because it is an unlucky random event that adds an element of “surprise” to the game.

This glitch is probably some kind of a Flash thing, and nothing to do with the code Mr Sun supplied. Perhaps extra code could be made to check for deflection angle = zero and then move the ball to the center to correct this ?

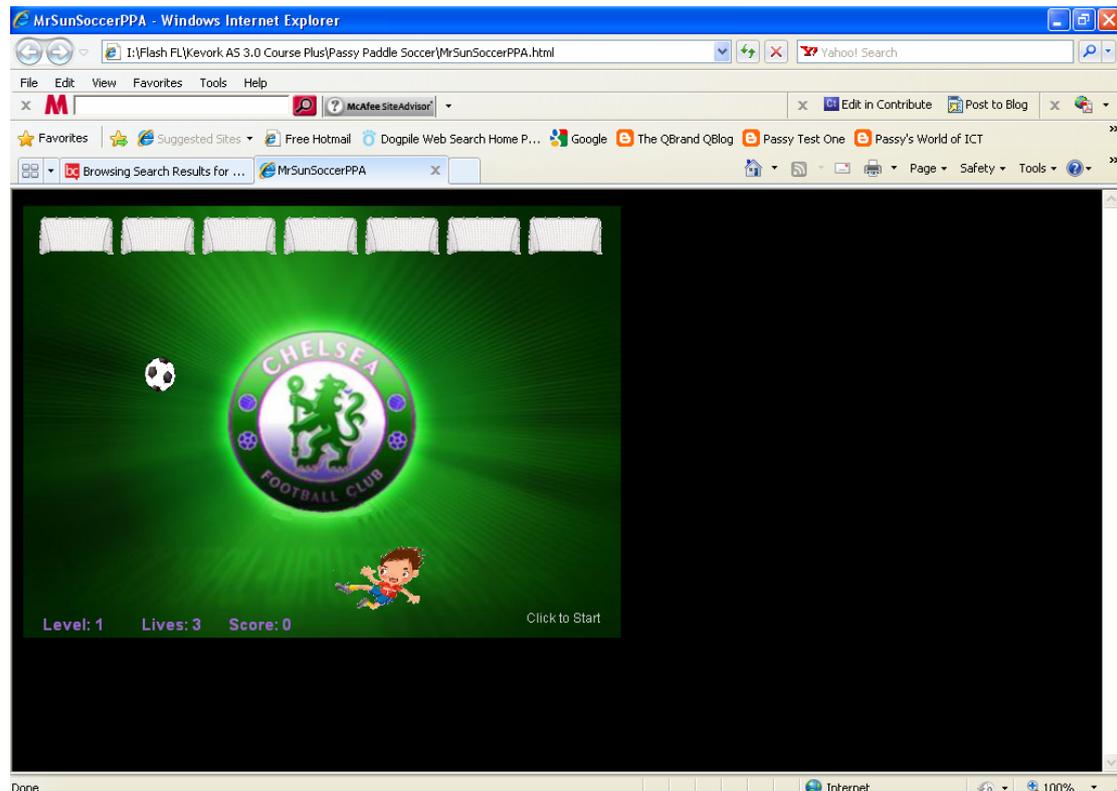
Getting the Game onto a Web Page

This was a bit tricky, but we got there in the end.

While in the Flash .FLA document, do **File > Publish Settings** and then tick the HTML option, and click on “Publish” button (and ignore the ok button) .

Flash Paddle Ball Game (Converted later into Soccer)

This will create some HTML that executes the Flash game in the top left hand corner of a webpage like this:



The HTML which embeds the game into the webpage is this :

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>MrSunSoccerPPA</title>
<script language="javascript">AC_FL_RunContent = 0;</script>
<script src="AC_RunActiveContent.js" language="javascript"></script>
</head>
<body bgcolor="#000000">
<!--url's used in the movie-->
<!--text used in the movie-->
<!--
Level: Lives:
The Final Score:
-->
<!-- saved from url=(0013)about:internet -->
<script language="javascript">
    if (AC_FL_RunContent == 0) {
        alert("This page requires AC_RunActiveContent.js.");
    } else {
        AC_FL_RunContent(
            'codebase',
            'http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=9,0,0,0',
            'width', '550',
            'height', '400',
            'src', 'MrSunSoccerPPA',
```

Flash Paddle Ball Game (Converted later into Soccer)

```
        'quality', 'high',
        'pluginspage',
'http://www.macromedia.com/go/getflashplayer',
        'align', 'middle',
        'play', 'true',
        'loop', 'false',
        'scale', 'showall',
        'wmode', 'window',
        'devicefont', 'false',
        'id', 'MrSunSoccerPPA',
        'bgcolor', '#000000',
        'name', 'MrSunSoccerPPA',
        'menu', 'true',
        'allowFullScreen', 'false',
        'allowScriptAccess', 'sameDomain',
        'movie', 'MrSunSoccerPPA',
        'salign', ''
    ); //end AC code
}
</script>
</noscript>
    <object classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swf
lash.cab#version=9,0,0,0" width="550" height="400"
id="MrSunSoccerPPA" align="middle">
    <param name="allowScriptAccess" value="sameDomain" />
    <param name="allowFullScreen" value="false" />
    <param name="movie" value="MrSunSoccerPPA.swf" /><param
name="loop" value="false" /><param name="quality" value="high"
/><param name="bgcolor" value="#000000" />    <embed
src="http://www.passyworld.com/passySWFs/MrSunSoccerPPA.swf"
loop="false" quality="high" bgcolor="#000000" width="550"
height="400" name="MrSunSoccerPPA" align="middle"
allowScriptAccess="sameDomain" allowFullScreen="false"
type="application/x-shockwave-flash"
pluginspage="http://www.macromedia.com/go/getflashplayer" />
    </object>
</noscript>
</body>
</html>
```

But when we played the game on the webpage, it did not progress to Level 2 when we finish Level 1 and so is not good.

So we tried regenerating it with “Loop” not ticked in the HTML play settings, because maybe the problem is that it wants to keep looping through Level 1 ?

This created the HTML shown above that has loop = false in it.

THIS FIXED THE PROBLEM, and the game now progresses up through the levels ok.

Flash Paddle Ball Game (Converted later into Soccer)

Getting the Game onto our Blog Page

The basic idea here is that we have to put all the components on a separate “Real” website that we have, and make sure they are all in the same folder. First we tried putting a new folder up on www.passyworld.com, and in it put the HTML and the game SWF file, and then see if we can execute this page within an “iframe” window on our Blog.

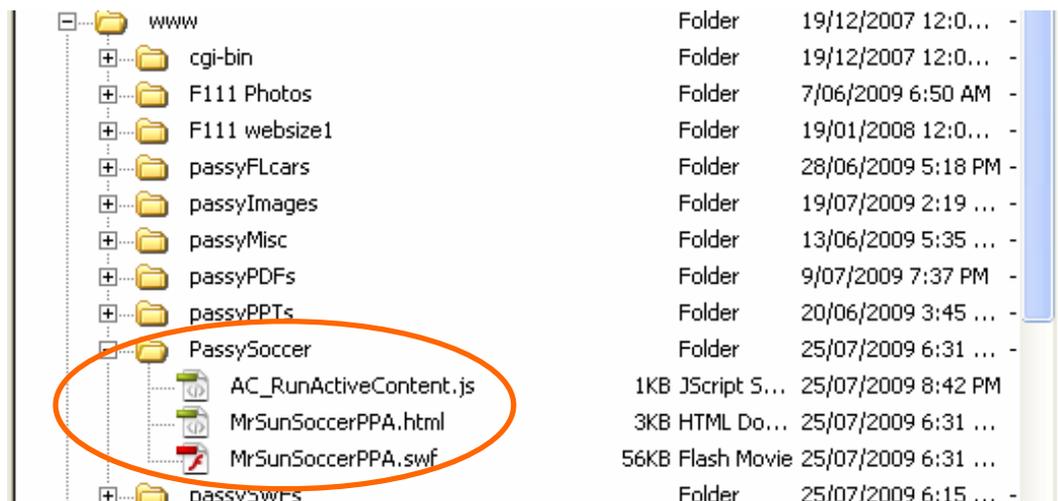
This was the address on Passy World:

<http://www.passyworld.com/PassySoccer/MrSunSoccerPPA.html>

Nothing happens when we run the page, because it says the page needs “AC_RunActiveContent.JS”

There was actually one of these .js files created in the same folder where the HTML and SWF were made by Flash. So we obtained this file, and uploaded it into a new “PassySoccer” folder on the websites remote server like this.

So the special website folder now has these 3 items”



The game runs, but we still have the Problem with the levels not working, if we do not do View > Refresh in Explorer between restarts of the Game. (or View > Reload in Firefox).

However, if we start a clean run, then we can have levels work ok.

What makes it annoyingly challenging to get to the end of the game, is the “glitching” that sometimes happens in the left and right hand extremities of the playing area. Once we have the ball stuck in the straight up and down glitch, we can only finish the game by allowing all our lives to be lost.

Finally, to set up on the Blog, an “iframe” window into the webpage at:

<http://www.passyworld.com/PassySoccer/MrSunSoccerPPA.html>

we put the following HTML code into our Blog:

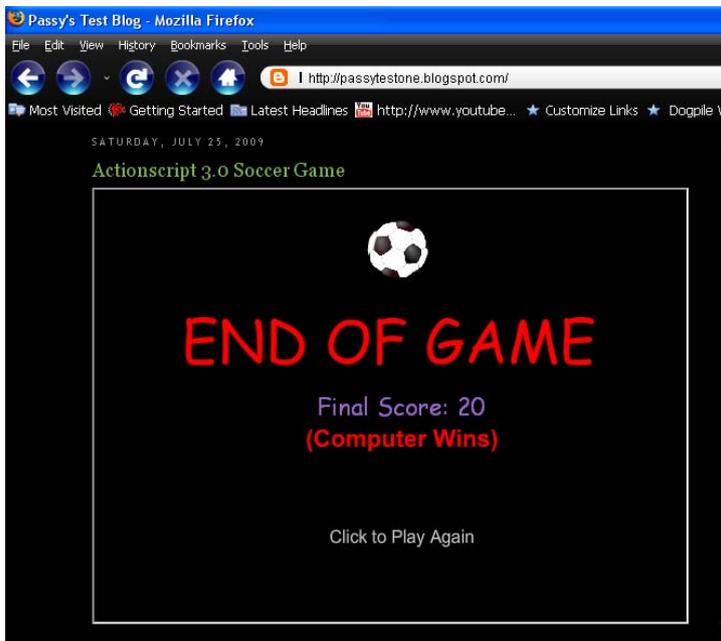
Flash Paddle Ball Game (Converted later into Soccer)

```
<div style="text-align: left;"><iframe marginheight="0"
src="http://www.passyworld.com/PassySoccer/MrSunSoccerPPA.html"
marginwidth="0" scrolling="no" width="550" frameborder="0"
height="400"></iframe>
</div>
```

We then made a slight change to narrow the width, and put a 1 pixel border around the game, which makes it look better by changing to :

```
width="545" frameborder="1"
```

It then works fine on the Blog, and the 5 pixel reduction in width is not at all noticeable. The end of game screens look a lot better with the border around them as well:



So that's it, a fun little soccer game.

Further down the track it would be nice to have the ball speed increase as we go up the levels, and also have keyboard keys that can move the player.

I will also definitely have to try out some more of Mr Sun's fabulous Flash Game tutorials at: <http://www.blogcatalog.com/blog/mr-sun-studios>

But that's it for now,

Enjoy,
Passy