```
/*****************************************************************************/
//
// UNIFORM BUILDER README:
//
// The script in this frame performs only some basic inits & processing.
// In general, the bulk of the code is contained in frame("script") #25.
// Some additional code is scattered in some of the buttons and a few of
// the controls.
//
// FILES REQUIRED @ RUNTIME:
// login.asp, load.asp, delete.asp, save.asp, urls.txt
//
// FILES REQUIRED @ GENTIME:
// HardwoodMen.as or HardwoodWomen.as
//
// Pricing and code specific to generating men's or women's version of the
// builder is stored externally in the .as files mentioned above.
// The library folder "_gender-details" contains symbols for men/women.
// The frame("script") contains documentation on how to generate men/women
// versions of each builder.
//
/*****************************************************************************/

// load urls for button links: membersLink, dealersLink, builderLink
loadVariables("urls.txt", this);

user = "";
password = "";

timeout = 30000;
start = getTimer();
index = -1;
xmlReady = false;
savedUniforms = null;
totalBytes = _root.getBytesTotal();
savedUniforms = new Array();
uniformName = "";

CODE0 = "0".charCodeAt(0);
CODE9 = "9".charCodeAt(0);
function getUniform(node)
{
        var theUniform = node.attributes;
        for (props in theUniform)
        {
                var theName = "theUniform."+props;
                var theValue = eval(theName);
                if ((theValue.charCodeAt(0) >= CODE0) && (theValue.charCodeAt(0) <= CODE9))
                        eval(theName) = parseInt(theValue);
                else if (theValue == "true")
                        eval(theName) = true;
                else if (theValue == "false")
                        eval(theName) = false;
        }

        return theUniform;
}

function getSavedList()
{
        if ((!this.loaded) || (this.status != 0))
        {
                // error in xml, move on w/o saved list
                xmlReady = true;
                return;
        }

        var top = this.lastChild;
        while (top.nodeName != "savedList")
        {
                top = top.previousSibling;
        }

        top = top.firstChild;
        i=0;
        while (top.nodeName == "uniform")
        {
                var theUniform = getUniform(top);
                theUniform.index = i;
                savedList.addItem(theUniform);
                _root.savedUniforms.push(theUniform);
                top = top.nextSibling;
                i++;
        }
```

```
 85
 86          xmlReady = true;
 87  }
 88
 89  // if loading from another builder, do not redisplay login screen
 90  if (eval("cusNum") != null)
 91  {
 92          _cusNum = parseInt(cusNum);
 93          if (_cusNum >= 0)
 94          {
 95                  thexml = new XML();
 96                  thexml.onLoad = getSavedList;
 97                  thexml.load("load.asp");
 98          }
 99          else
100          {
101                  xmlReady = true;
102          }
103
104          start = getTimer();
105
106          // wait until login screen loaded to bypass
107          stop();
108  }
109  else
110  {
111          // move onto login screen
112          play();
113  }
```

```
/*****************************************************************************/
//
// GENERATING MENS/WOMENS SPECIFIC BUILDER:
//
// 1. Comment/Uncomment men/women .as script include below
// 2. Swap instance in frames.
//    a. 25 script - banner, uniform
//    b. 30 start - template, templateList, unispecs
//    c. 35 colors - zoomed.uniform (maybe hidden behind help)
//    Hint: check the use count to ensure only correct symbols are used
// 3. Generate .swf then rename to appropriate
//
/*****************************************************************************/

#include "HardwoodMen.as"
//#include "HardwoodWomen.as"

// color constants
NONE = 0;
WHITE = 1;
BLACK = 2;
BROWN = 3;
NAVY = 4;
PURPLE = 5;
ROYAL = 6;
POWDER = 7;
GREEN = 8;
KELLY = 9;
MAROON = 10;
CARDINAL = 11;
SCARLET = 12;
ORANGE = 13;
GOLD = 14;
VEGAS = 15;
SILVER = 16;

// font constants
SYRACUSE = 1;
FULL = 2;

// style constants
STRAIGHT = 1;
ROTATE = 2;

MAXCHARS = 9;
FRAMEOFFSET = 5;
CODEA = "A".charCodeAt(0);
CODEZ = "Z".charCodeAt(0);

nameStyles = new Array(
        "none","STRAIGHT","ARCHED"
);

fontNames = new Array(
        "none","SYRACUSE","FULL BLOCK"
);

fontName = new Array(
        "none",
        "syracuse", "full"
);

colorText = new Array(
        "NONE",
        "WHITE", "BLACK", "BROWN", "NAVY", "PURPLE", "ROYAL", "POWDER BLUE", "DARK GREEN",
        "KELLY", "MAROON", "CARDINAL", "SCARLET", "ORANGE", "BRIGHT GOLD", "VEGAS GOLD", "SILVER"
);

// color options
bodyColorOptions = [ WHITE, BLACK, NAVY, PURPLE, ROYAL, GREEN, NONE, MAROON, CARDINAL, SCARLET, GOLD, VEGAS ];
letteringOptions = [ WHITE, BLACK, NAVY, PURPLE, ROYAL, POWDER, GREEN, MAROON, CARDINAL, SCARLET, ORANGE, GOLD, VEGAS, SILVER ];

colorHex = new Array(
        -1,
        0xFFFFFF, 0x000000, 0x330000, 0x000033,
        0x330033, 0x003366, 0x6699CC, 0x003300,
        0x006633, 0x660033, 0x990000, 0xCC0000,
        0xFF6600, 0xFFCC00, 0xCCCC66, 0x999999
);

colorItems = new Array(
        "bodyColor","trimColor","nameMain","nameOutline"
);
```

```
 84
 85 function numColors()
 86 {
 87         return (_nameMain == NONE) ? 0 : ((_nameOutline == NONE) ? 1 : 2);
 88 }
 89
 90 function calculatePrice()
 91 {
 92         jerseyPrice = garmentPrice[isShirt()?1:0][numColors()];
 93         shortsPrice = garmentPrice[isShirt()?1:0][3];
 94
 95         // format by adding $xx.00
 96         totalPrice = formatPrice(jerseyPrice + shortsPrice);
 97         jerseyPrice = formatPrice(jerseyPrice);
 98         shortsPrice = formatPrice(shortsPrice);
 99 }
100
101 function formatPrice(theNumber)
102 {
103         theNumber = Math.round(theNumber*100);
104         if ((theNumber%100) == 0)
105                 return "$"+theNumber/100+".00";
106         else if ((theNumber%10) == 0)
107                 return "$"+theNumber/100+"0";
108         else
109                 return "$"+theNumber/100;
110 }
111
112 function getColor(src)
113 {
114         theColor = eval("_" + src);
115         if (theColor == null)
116                 theColor = 0;
117         eval(src + "Text") = colorText[theColor];
118         return theColor;
119 }
120
121 function setObjectColor(img, theColor)
122 {
123         img.theColor = theColor;
124         img._visible = (theColor == NONE) ? false : true;
125         // deselect none and set item's color
126         var newColor = new Color(img);
127         if (theColor > 0)
128                 newColor.setRGB(colorHex[theColor]);
129         else
130                 newColor.setRGB(0xcccccc);
131 }
132
133 function setColor(src, theColor)
134 {
135         // update global variable (same as object name)
136         eval("_" + src) = theColor;
137
138         // update description, none, and color
139         eval(src + "Text") = colorText[theColor];
140
141         // update description, none, and color
142         if (theColor == NONE)
143         {
144                 eval(src).hilite._visible = false;
145                 eval(src+"None").gotoAndStop("on");
146         }
147         else
148         {
149                 eval(src).hilite._visible = true;
150                 eval(src+"None").gotoAndStop("off");
151         }
152
153         if (src.indexOf("name") == 0)
154         {
155                 // layer = main or outline
156                 var layer = src.substr(4);
157
158                 // update font numbers color + visible
159                 for (var i=1; i<=MAXCHARS; i++)
160                 {
161                         setObjectColor(eval("nameImg.top.char" + i + "." + layer),theColor);
162                 }
163
164                 // update font + back numbers color + visible
165                 if (isShirt())
166                         setObjectColor(eval("numImg." + layer),NONE);
167                 else
```

```
168                         setObjectColor(eval("numImg." + layer),theColor);
169
170                 // if no main color, outline must be NONE also
171                 _isBlank = (_root._nameMain == _root.NONE)
172
173                 if ((layer == "Main") && (_nameMain == NONE))
174                 {
175                         setColor("nameOutline",NONE);
176                         nameOutline.setEnabled(false);
177                 }
178                 else
179                 {
180                         nameOutline.setEnabled(true);
181                 }
182
183                 calculatePrice();
184         }
185         else
186         {
187                 // update uniform item's color
188                 setObjectColor(eval("uniform." + src + "Img"),theColor);
189
190                 // update swoosh to contrast with body
191                 if (isShirt())
192                 {
193                         setObjectColor(swoosh,NONE);
194                 }
195                 else if ((_bodyColor == WHITE) || (_bodyColor >= GOLD))
196                 {
197                         if ((_trimColor == WHITE) || (_trimColor >= GOLD))
198                         {
199                                 setObjectColor(swoosh,BLACK);
200                         }
201                         else
202                         {
203                                 setObjectColor(swoosh,_trimColor);
204                         }
205                 }
206                 else
207                         setObjectColor(swoosh,WHITE);
208
209                 setUniformDetails();
210
211                 // trim color options are determined by body color
212                 if (src == "bodyColor")
213                 {
214                         trimColor.setOptions(trimColorOptions[_bodyColor]);
215                 }
216         }
217
218         if (isZoomed) zoomed.update();
219 }
220
221 function getFont(src)
222 {
223         var theFont = eval("_" + src);
224         return fontName[theFont];
225 }
226
227 function setFont(src, theFont)
228 {
229         // update global variable (same as object name)
230         eval("_" + src) = theFont;
231
232         var frameName = fontName[theFont];
233
234         if (true)
235         {
236                 // update uniform number font for front + back on all layers
237                 // main, outline
238                 for (props in numImg)
239                 {
240                         eval("numImg." + props).gotoAndStop(frameName);
241                 }
242
243                 // update each letter in team name
244                 for (i=1; i<=MAXCHARS; i++)
245                 {
246                         eval("nameImg.top.char" + i).gotoAndStop(frameName);
247                 }
248
249                 // due to awkward embedding of layers for team name
250                 // may need to refresh font, color, and name
251                 setName("nameText",_nameText);
```

```
252
253                 setColor("nameMain",_nameMain);
254                 setColor("nameOutline",_nameOutline);
255         }
256
257         if (isZoomed) zoomed.update();
258         calculatePrice();
259 }
260
261 function setChar(src, theLetter)
262 {
263         var upper = theLetter.toUpperCase();
264         var code = upper.charCodeAt(0);
265         if ((code >= CODEA) && (code <= CODEZ))
266         {
267                 src.outline.gotoAndStop(code-CODEA+1);
268                 src.main.gotoAndStop(code-CODEA+1);
269         }
270         else
271         {
272                 src.outline.gotoAndStop(CODEZ);
273                 src.main.gotoAndStop(CODEZ);
274         }
275 }
276
277 function getName(src)
278 {
279         return eval("_"+src);
280 }
281
282 function setName(src, theName)
283 {
284         eval("_" + src) = theName;
285
286         if (src == "saveName") return;
287
288         imgString = "nameImg.top";
289
290         // skewed & rotate place characters depending on odd & even lengths
291         toFrame = (theName.length%2)+1;
292         eval(imgString).gotoAndStop(toFrame);
293         eval("zoomed."+imgString).gotoAndStop(toFrame);
294
295         // pad spaces before to center text
296         for (j=1; j<=Math.floor((MAXCHARS-theName.length))/2; j++)
297         {
298                 setChar(eval(imgString + ".char" + (j)), " ");
299                 setChar(eval("zoomed."+imgString + ".char" + (j)), " ");
300         }
301
302         // the actual name string
303         for (i=0; i<theName.length; i++)
304         {
305                 setChar(eval(imgString + ".char" + (j+i)), theName.charAt(i));
306                 setChar(eval("zoomed."+imgString + ".char" + (j+i)), theName.charAt(i));
307         }
308
309         // pad spaces after to center text
310         for (j+=i; j<=MAXCHARS; j++)
311         {
312                 setChar(eval(imgString + ".char" + (j)), " ");
313                 setChar(eval("zoomed."+imgString + ".char" + (j)), " ");
314         }
315
316         // len @ MAXCHARS, switch from even to odd last letter needs color reset
317         if (theName.length == MAXCHARS)
318         {
319                 setColor("nameMain",_nameMain);
320                 setColor("nameOutline",_nameOutline);
321         }
322
323
324         nameImg.gotoAndStop(_root.nameImg._currentFrame);
325         nameImg.top.gotoAndStop(_root.nameImg.top._currentFrame);
326 }
327
328 function getStyle()
329 {
330         return _nameStyle*FRAMEOFFSET;
331 }
332
333 function setStyle(theStyle)
334 {
335         _nameStyle = theStyle;
```

```
336
337         nameImg.gotoAndStop(theStyle*FRAMEOFFSET);
338         nameStyle.gotoAndStop(theStyle*FRAMEOFFSET);
339
340         // due to awkward embedding of layers for team name
341         // may need to refresh font, color, and name
342         setFont("nameFont",_nameFont);
343 }
344
345 function getShow(src)
346 {
347         var toShow = eval("_" + src);
348         eval(src + "Show")._visible = toShow;
349         return toShow;
350 }
351
352 function setShow(src,toShow)
353 {
354         eval("_" + src) = toShow;
355         eval("uniform." + src + "Show")._visible = toShow;
356 }
357
358 function zoom(isVisible)
359 {
360         isZoomed = isVisible;
361         if (isZoomed)
362         {
363                 zoomed.update();
364                 zoomed._visible = true;
365                 zoomButtonSmall._visible = true;
366                 zoomButtonSmall.gotoAndStop("out");
367                 zoomButton.gotoAndStop("out");
368         }
369         else
370         {
371                 zoomed._visible = false;
372                 zoomButtonSmall._visible = false;
373                 zoomButtonSmall.gotoAndStop("in");
374                 zoomButton.gotoAndStop("in");
375         }
376 }
377
378 function previewTemplate(uniformStyle)
379 {
380         _uniNum = 0;
381         index = -1;
382         template.gotoAndStop(uniformStyle);
383         template._visible = true;
384         uniform._visible = false;
385 }
386
387 function initTemplate()
388 {
389         _nameText = "TEAM NAME";
390         _saveName = "TEAM-NAME";
391         _saveSchool = "";
392         _saveCity = "";
393         _saveState = "";
394
395         isSaved = false;
396
397         firstStep();
398         calculatePrice();
399 }
400
401 function initialize()
402 {
403         uniformName = "\"" + uniformText[_uniformStyle] + "\"";
404         uniformType = isShirt() ? "SHOOTING SHIRT" : "GAME UNIFORM";
405         uniformNameFull = uniformName + " " +  uniformType;
406
407         trimColorOptions = new Array();
408         trimColorOptions[WHITE] = [ BLACK, NAVY, PURPLE, ROYAL, GREEN, NONE, NONE, MAROON, CARDINAL, SCARLET, GOLD,
    VEGAS ];
409         trimColorOptions[BLACK] = [ WHITE, PURPLE, ROYAL, GREEN, MAROON, NONE, NONE, CARDINAL, SCARLET, GOLD, VEGAS
    ];
410         trimColorOptions[VEGAS] = [ WHITE, BLACK, NAVY, PURPLE, ROYAL, NONE, NONE, GREEN, MAROON, CARDINAL, SCARLET
    ];
411         trimColorOptions[GOLD]  = [ WHITE, BLACK, NAVY, PURPLE, ROYAL, NONE, NONE, GREEN, MAROON, CARDINAL, SCARLET
    ];
412         trimColorOptions[SCARLET] = [ WHITE, BLACK, NAVY, ROYAL, GOLD, VEGAS ];
413         trimColorOptions[ROYAL] = [ WHITE, BLACK, SCARLET, GOLD, VEGAS ];
414         trimColorOptions[PURPLE] = [ WHITE, BLACK, GOLD, VEGAS ];
415         trimColorOptions[NAVY] = [ WHITE, SCARLET, GOLD, VEGAS ];
```

```
416         trimColorOptions[GREEN] = [ WHITE, BLACK, GOLD, VEGAS ];
417         trimColorOptions[CARDINAL] = [ WHITE, BLACK, GOLD, VEGAS ];
418         trimColorOptions[MAROON] = [ WHITE, BLACK, GOLD, VEGAS ];
419
420         block._visible = !isShirt();
421         swoosh._visible = !isShirt();
422 }
423
424 function previewUser()
425 {
426         unispecs.gotoAndStop(uniformFrames[_uniformStyle]);
427         uniform.gotoAndStop(uniformFrames[_uniformStyle]+"-user");
428         uniform._visible = true;
429         template._visible = false;
430
431         for (var i=0; i<colorItems.length; i++)
432                 setColor(colorItems[i],eval("_" + colorItems[i]));
433
434         setStyle(_nameStyle);
435 }
436
437 function initUser(aUniform)
438 {
439         index = aUniform.index;
440
441         _uniNum = aUniform.uniNum;
442         _saveName = aUniform.saveName;
443         _saveSchool = aUniform.saveSchool;
444         _saveCity = aUniform.saveCity;
445         _saveState = aUniform.saveState;
446
447         isSaved = true;
448         if ((_saveName.length == 0) ||
449                 (_saveSchool.length == 0) ||
450                 (_saveCity.length == 0) ||
451                 (_saveState.length < 2))
452         {
453                 isSaved = false;
454         }
455
456         _uniformStyle = aUniform.uniformStyle;
457         _bodyColor = aUniform.bodyColor;
458         _trimColor = aUniform.trimColor;
459         _nameMain = aUniform.nameMain;
460         _nameOutline = aUniform.nameOutline;
461         _nameFont = aUniform.nameFont;
462         _nameText = aUniform.nameText;
463         _nameStyle = aUniform.nameStyle;
464
465         initialize();
466         previewUser();
467 }
468
469 //for (stuff in _root) { if (stuff.charAt(0)=='_') trace(stuff); }
470 //initialize setup based on defaults
471 var _uniformStyle,_bodyColor,_trimColor;
472 var _nameMain,_nameOutline,_nameFont,_nameText,_nameStyle;
473 var _uniNum, _saveName;
474
475 gotoAndStop("start");
```