```
/*********************************************************************************/
//
// 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:
// CenterCourtMen.as or CenterCourtWomen.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 = "";
msg = "";

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++;
        }

        xmlReady = true;
}

// if loading from another builder, do not redisplay login screen
if (eval("cusNum") != null)
{
        _cusNum = parseInt(cusNum);
        if (_cusNum >= 0)
        {
                thexml = new XML();
                thexml.onLoad = getSavedList;
                thexml.load("load.asp");
        }
        else
        {
                xmlReady = true;
        }

        start = getTimer();

        // wait until login screen loaded to bypass
        stop();
}
else
{
        // move onto login screen
        play();
}
```

```
/*******************************************************************************/
//
// 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
//    c. 40 body - 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 "CenterCourtMen.as"
#include "CenterCourtWomen.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;
GREY = 16;

// font constants
GOTHIC = 1;
FULL = 2;
NIKEBLOCK = 3;
INLINE = 4;
DOUBLE = 5;

// style constants
STRAIGHT = 1;
ROTATE = 2;
SKEWED = 3;
BOTTOM = 4;

// application constants
TWILL = 1;
PRINT = 2;

// fabric constants
SHEEN = 0;
MESH = 1;
BRITE = 2;

// waistband constants
KNIT = 0;
NEEDLE = 1;

// accent constants
OUTLINE = 1;
SHADOW = 2;

MAXCHARS = 13;
FRAMEOFFSET = 5;
CODEA = "A".charCodeAt(0);
CODEZ = "Z".charCodeAt(0);

fabricStyles = new Array(
        "SHIMMERSHEEN","MICROMESH", "PRO-BRITE"
);

nameStyles = new Array(
        "none","STRAIGHT","ARCHED","VERTICAL ARCHED","TOP & BOTTOM"
);

fontNames = new Array(
        "none",
        "GOTHIC",
        "FULL BLOCK",
        "NIKE TRUE BLOCK",
        "NIKE INLINE",
        "DOUBLE BLOCK"
);

fontDesc = new Array(
        "none",
        "GOTHIC Solid/Outline/Shadow",
        "FULL BLOCK Solid/Outline/Shadow",
        "NIKE TRUE BLOCK Solid/Outline/Shadow",
        "NIKE INLINE Solid/Inline\nAvailable Printed Only",
        "DOUBLE BLOCK Solid/Outline\nAvailable Printed Only"
);

fontName = new Array(
        "none",
        "gothic", "full", "nikeblock", "inline", "double"
);

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

colorHex = new Array(
        -1,
        0xffffff, 0x000000, 0x330000, 0x000066,
        0x330066, 0x003399, 0x6699CC, 0x006600,
        0x006633, 0x660033, 0x990000, 0xCC0000,
        0xFF6600, 0xFFCC33, 0xCCCC99, 0x999999
);

colorItems = new Array(
        "bodyColor","side1Color","side2Color","side3Color","trim1Color","trim2Color","trim3Color",
        "numMain","numAccentColor","nameMain","nameAccentColor","homePlateColor"
);

function countLetters(theName)
{
        var numLetters = 0;
        numLetters = theName.length;
        for (var i=numLetters-1; i>=0; i--)
        {
                if (theName.charAt(i)==' ')
                        numLetters--;
        }
        return numLetters;
}

function markupPrice(theNumber)
{
        return Math.round(theNumber*20125)/10000;
}

function formatPrice(theNumber)
{
        theNumber = Math.round(theNumber*100);
        if ((theNumber%100) == 0)
                return "$"+theNumber/100+".00";
        else if ((theNumber%10) == 0)
                return "$"+theNumber/100+"0";
        else
                return "$"+theNumber/100;
}

function calculatePrice()
{
        logosPrice = _homePlate ? homePlatePrice : 0;
        jerseyPrice = garmentPrice[_uniformStyle][_fabric];
        if (hasWaistband())
                shortsPrice = garmentPrice[_uniformStyle][_fabric+3*(_waistband+1)];
        else
                shortsPrice = garmentPrice[_uniformStyle][_fabric+3];

        // print: 2 loc front + back
        // twill: 4 numbers = 2 front + 2 back
        if (_numApp == PRINT)
                numPrice = numPrintPrice[(_numFont>=INLINE)?1:0][(_numAccent==NONE)?0:1] * 2;
        else
                numPrice = numTwillPrice[_numFrontSize][_numAccent]*2 + numTwillPrice[_numBackSize][_numAccent]*2;

        // print: top + bottom
        // twill: num letters
        if (_nameApp == PRINT)
```

```
                namePrice = namePrintPrice[(_nameAccent==NONE)?0:1] * ((_nameStyle == BOTTOM)?2:1);
        else
        {
                numLetters = countLetters(_nameTop) + ((_nameStyle == BOTTOM)?countLetters(_nameBottom):0);
                namePrice = nameTwillPrice[_nameSize][_nameAccent] * numLetters;
        }

        // markup
        jerseyPrice = markupPrice(jerseyPrice);
        shortsPrice = markupPrice(shortsPrice);
        numPrice = markupPrice(numPrice);
        namePrice = markupPrice(namePrice);
        logosPrice = markupPrice(logosPrice);
        totalPrice = jerseyPrice + shortsPrice + numPrice + namePrice + logosPrice;

        jerseyPrice = formatPrice(jerseyPrice);
        shortsPrice = formatPrice(shortsPrice);
        numPrice = formatPrice(numPrice);
        namePrice = formatPrice(namePrice);
        logosPrice = formatPrice(logosPrice);
        totalPrice = formatPrice(totalPrice);
}

function getColor(src)
{
        theColor = eval("_" + src);
        if (theColor == null)
                theColor = 0;
        eval(src + "Text") = colorText[theColor];
        return theColor;
}

function setObjectColor(img, theColor)
{
        img._visible = (theColor == NONE) ? false : true;
        // deselect none and set item's color
        var newColor = new Color(img);
        if (theColor > 0)
                newColor.setRGB(colorHex[theColor]);
        else
                newColor.setRGB(0xcccccc);
}

function replaceColor(oldColor, newColor)
{
        for (var i=0; i<colorItems.length; i++)
                if ((eval("_"+colorItems[i]) == oldColor) && (oldColor != NONE))
                        setColor(colorItems[i],newColor);
}

function setTeamColor(src, theColor)
{
        // update global variable (same as object name)
        var oldColor = eval("_" + src);
        eval("_" + src) = theColor;

        // update description, none, and color
        eval(src + "Text") = colorText[theColor];
        if (theColor == NONE)
        {
                eval(src).hilite._visible = false;
        }
        else
        {
                eval(src).hilite._visible = true;
                eval(src+"None").gotoAndStop("off");
        }

        // update variables
        var theNum = src.charAt(9);
        var isDuplicate = false;
        for (var i=1; i<=4; i++)
        {
                if ((eval("_teamColor"+i) == theColor) && (i != theNum) && (theColor != NONE))
                {
                        isDuplicate = true;
                }
        }

        if (!isDuplicate)
        {
                teamColors = new Array(0,0,0,0);
                teamColorMsg = "";
                replaceColor(oldColor,theColor);

                var i=0;
                // consolidate duplicates and none selected
                for (var j=0; j<4; j++)
                {
                        var theColor = eval("_teamColor"+(j+1));
                        if (theColor != NONE)
                        {
                                teamColors[i] = theColor;
                                i++;
                        }
                }

                // update visuals
                for (var j=0; j<4; j++)
                {
                        var theColor = teamColors[j];
                        var button = selection.buttonArray[j];
                        button.theColor = theColor;
                        setObjectColor(button,teamColors[j]);
                        selection.frameArray[j]._visible = (theColor != NONE);
                }
                return true;
        }
        else
        {
                teamColorMsg = "Sorry, duplicate colors not allowed\nwhen selecting team colors.";
                eval("_" + src) = oldColor;
                eval(src + "Text") = colorText[oldColor];
                if (oldColor == NONE)
                {
                        eval(src).hilite._visible = false;
                        eval(src+"None").gotoAndStop("on");
                }
                else
                {
                        eval(src).hilite._visible = true;
                }
                return false;
        }
}

function setColor(src, theColor)
{
        // update global variable (same as object name)
        eval("_" + src) = theColor;

        // update description, none, and color
        if (theColor == NONE)
        {
                eval(src).hilite._visible = false;
        }
        else
        {
                eval(src).hilite._visible = true;
                eval(src+"None").gotoAndStop("off");
        }

        if (src.indexOf("Accent")>0)
        {
                if (src.indexOf("num")==0)
                {
                        prefix = "num";
                        theAccent = _numAccent;
                }
                else
                {
                        prefix = "name";
                        theAccent = _nameAccent;
                }

                if (theAccent==OUTLINE)
                {
                        setColor(prefix+"Outline",theColor);
                        setColor(prefix+"Shadow",NONE);
                }
                else if (theAccent==SHADOW)
                {
                        setColor(prefix+"Outline",NONE);
                        setColor(prefix+"Shadow",theColor);
                }
                else
                {
                        setColor(prefix+"Outline",NONE);
                        setColor(prefix+"Shadow",NONE);
                }
        }
        else if (src.indexOf("num") == 0)
        {
                // layer = main, outline, or shadow
                var layer = src.substr(3);

                // update font + back numbers color + visible
                setObjectColor(eval("numFrontImg." + layer),theColor);
                setObjectColor(eval("numBackImg." + layer),theColor);
        }
        else if (src.indexOf("name") == 0)
        {
                // layer = main, outline, or shadow
                var layer = src.substr(4);

                // update font numbers color + visible
                for (var i=1; i<=MAXCHARS; i++)
                {
                        setObjectColor(eval("nameImg.top.char" + i + "." + layer),theColor);
                        setObjectColor(eval("nameImg.bottom.char" + i + "." + layer),theColor);
                }
        }
        else if (src.indexOf("home") == 0)
        {
                setObjectColor(uniform.homePlateShow.letter,theColor);
        }
        else
        {
                // update uniform item's color
                setObjectColor(eval("uniform." + src + "Img"),theColor);

                // update swoosh to contrast with body
                if (src == "bodyColor")
                {
                        // update uniform item's color
                        if ((theColor == WHITE) || (theColor >= GOLD))
                                setObjectColor(swoosh,BLACK);
                        else
                                setObjectColor(swoosh,WHITE);

                        // home plate color matches body
                        setObjectColor(uniform.homePlateShow.body,theColor);
                }
        }

        if (isZoomed) zoomed.update();
}

function getFont(src)
{
        var theFont = eval("_" + src);
        eval(src + "Text") = fontDesc[theFont];
        return fontName[theFont];
}

function setFont(src, theFont)
{
        // update global variable (same as object name)
        eval("_" + src) = theFont;

        // update description
        eval(src + "Text") = fontDesc[theFont];

        var frameName = fontName[theFont];
        if (src.indexOf("num") == 0)
        {
                // update uniform number font for front + back on all layers
                // main, outline, shadow
                for (props in numFrontImg)
                {
                        eval("numFrontImg." + props).gotoAndStop(frameName);
                        eval("numBackImg." + props).gotoAndStop(frameName);
                }

                if (theFont >= INLINE)
                {
                        _numApp = PRINT;
                        if (_numAccent == SHADOW)
                        {
                                _numAccent = NONE;
                                _numAccentColor = NONE;
                        }
                }
        }
        else if (src.indexOf("name") == 0)
        {
                // update each letter in team names top and bottom
                for (i=1; i<=MAXCHARS; i++)
                {
                        eval("nameImg.top.char" + i).gotoAndStop(frameName);
                        eval("nameImg.bottom.char" + i).gotoAndStop(frameName);
                }

                // due to awkward embedding of layers for team name
                // may need to refresh font, color, and name
                setName("nameTop",_nameTop);
                setName("nameBottom",_nameBottom);

                setColor("nameMain",_nameMain);
                setColor("nameAccentColor",_nameAccentColor);

                if (theFont >= INLINE)
                {
                        _nameApp = PRINT;
                        if (_nameAccent == SHADOW)
                        {
                                _nameAccent = NONE;
                                _nameAccentColor = NONE;
                        }
                }
        }
        else if (src.indexOf("home") == 0)
        {
                // home plate font matches
                uniform.homePlateShow.letter.gotoAndStop(frameName);
                setChar(uniform.homePlateShow.letter, _homePlateLetter);
        }

        if (isZoomed) zoomed.update();
        calculatePrice();
}

function setChar(src, theLetter)
{
//trace("setChar "+src+"="+theLetter);
        var upper = theLetter.toUpperCase();
        var code = upper.charCodeAt(0);
        if ((code >= CODEA) && (code <= CODEZ))
        {
                src.outline.gotoAndStop(code-CODEA+1);
                src.main.gotoAndStop(code-CODEA+1);
                src.shadow.gotoAndStop(code-CODEA+1);
        }
        else
        {
                src.outline.gotoAndStop(CODEZ);
                src.main.gotoAndStop(CODEZ);
                src.shadow.gotoAndStop(CODEZ);
        }
}

function getName(src)
{
        return eval("_"+src);
}

function setName(src, theName)
{
        eval("_" + src) = theName;

        if (src == "saveName") return;

        // src = top or bottom
        loc = (src == "nameTop") ? "top" : "bottom";
        imgString = "nameImg." + loc;

        // skewed & rotate place characters depending on odd & even lengths
        eval(imgString).gotoAndStop((theName.length%2)+1);
        eval("zoomed."+imgString).gotoAndStop((theName.length%2)+1);

        // pad spaces before to center text
        for (j=1; j<=Math.floor((MAXCHARS-theName.length))/2; j++)
        {
                setChar(eval(imgString + ".char" + (j)), " ");
                setChar(eval("zoomed."+imgString + ".char" + (j)), " ");
        }

        // the actual name string
        for (i=0; i<theName.length; i++)
        {
                setChar(eval(imgString + ".char" + (j+i)), theName.charAt(i));
                setChar(eval("zoomed."+imgString + ".char" + (j+i)), theName.charAt(i));
        }

        // pad spaces after to center text
        for (j+=i; j<=MAXCHARS; j++)
        {
                setChar(eval(imgString + ".char" + (j)), " ");
                setChar(eval("zoomed."+imgString + ".char" + (j)), " ");
        }

        calculatePrice();
}

function getStyle()
{
        return _nameStyle*FRAMEOFFSET;
}

function setStyle(theStyle)
{
        _nameStyle = theStyle;

        nameStyle.gotoAndStop(theStyle*FRAMEOFFSET);
        nameImg.gotoAndStop(theStyle*FRAMEOFFSET);

        if (theStyle==BOTTOM)
        {
                nameTop.label = "TOP";
                nameBottom._visible = true;
        }
        else
        {
                nameTop.label = "NAME";
                nameBottom._visible = false;
        }

        setFont("nameFont",_nameFont);
        setSize("nameSize",_nameSize);
}

function getShow(src)
{
        var toShow = eval("_" + src);
        eval(src + "Show")._visible = toShow;
        return toShow;
}

function setShow(src,toShow)
{
        eval("_" + src) = toShow;
        eval("uniform." + src + "Show")._visible = toShow;
        if (src == "homePlate")
        {
                homePlateSelectedLetter.setEnabled(toShow);
                homePlateColor.setEnabled(toShow);
                homePlateFont.setEnabled(toShow);

                zoomed.uniform.homePlateShow._visible = toShow;
                calculatePrice();
        }
}

function getApp(src)
{
        return eval("_"+src);
}

function setApp(src,theApp)
{
        eval("_"+src) = theApp;
        calculatePrice();
}

function getFabric(src)
{
        return _fabric;
}

function setFabric(src,theFabric)
```

```
{
        _fabric = theFabric;
        calculatePrice();
}

function getAccent(src)
{
        return eval("_" + src);
}

function setAccent(src,theAccent)
{
        eval("_" + src) = theAccent;
        setColor(src+"Color",eval("_"+src+"Color"));
        calculatePrice();
}

function getSize(src)
{
        return eval("_" + src);
}

function setSize(src,theSize)
{
        eval("_" + src) = theSize;

        // clear other buttons as radio
        if (src == "numFrontSize")
        {
                zoomed.numFrontImg._xscale = zoomed.numFrontImg._yscale =
                numFrontImg._xscale = numFrontImg._yscale = theSize/8*100;
                if (theSize == 4) numFrontSize5.gotoAndStop("off");
                if (theSize == 5) numFrontSize4.gotoAndStop("off");
        }
        else if (src == "numBackSize")
        {
                numBackImg._xscale = numBackImg._yscale = theSize/8*100;
                if (theSize == 6) numBackSize8.gotoAndStop("off");
                if (theSize == 8) numBackSize6.gotoAndStop("off");
        }
        else if (src == "nameSize")
        {
                zoomed.nameImg.top._xscale = zoomed.nameImg.top._yscale =
                nameImg.top._xscale = nameImg.top._yscale = theSize/4*100;
                zoomed.nameImg.bottom._xscale = zoomed.nameImg.bottom._yscale =
                nameImg.bottom._xscale = nameImg.bottom._yscale = theSize/4*100;
                if (theSize == 3) nameSize4.gotoAndStop("off");
                if (theSize == 4) nameSize3.gotoAndStop("off");
        }

        calculatePrice();
}

function getLogo()
{
        return _homePlateLetter;
}

function setLogo(theLetter)
{
        _homePlateLetter = theLetter;
        setChar(uniform.homePlateShow.letter,theLetter);
        setChar(zoomed.uniform.homePlateShow.letter,theLetter);
}

function getWaistband(src)
{
        return _waistband;
}

function setWaistband(src, theWaistband)
{
        _waistband = theWaistband;
        calculatePrice();
}

function zoom(isVisible)
{
        isZoomed = isVisible;
        if (isZoomed)
        {
                zoomed.update();
                zoomed._visible = true;
                zoomButtonSmall._visible = true;
                zoomButtonSmall.gotoAndStop("out");
                zoomButton.gotoAndStop("out");
        }
        else
        {
                zoomed._visible = false;
                zoomButtonSmall._visible = false;
                zoomButtonSmall.gotoAndStop("in");
                zoomButton.gotoAndStop("in");
        }
}

function previewTemplate(uniformStyle)
{
        _uniNum = 0;
        index = -1;
        template.gotoAndStop(uniformStyle);
        template._visible = true;
        uniform._visible = false;

        if (uniformStyle.indexOf("away")>0)
                setObjectColor(template.swoosh,WHITE);
        else
                setObjectColor(template.swoosh,BLACK);
}

function initTemplate()
{
        uniform.gotoAndStop(template._currentFrame);

        _saveName = "TEAM-NAME";
        _saveSchool = "";
        _saveCity = "";
        _saveState = "";

        isSaved = false;

        _nameTop = "TEAM NAME";
        _nameBottom = "BOTTOM";
        _homePlateLetter = "T";

        gotoAndStop("colors");
        calculatePrice();
}

function initialize()
{
        uniformName = "\"" + uniformText[_uniformStyle] + "\"";
        uniformNameFull = uniformName + " GAME UNIFORM";
        hasStockMsg._visible = hasStock(_uniformStyle);
}

function previewUser()
{
        uniform._visible = true;

        uniform.gotoAndStop(uniformFrames[_uniformStyle]+"-user");
        uniform._visible = true;
        template._visible = false;

        for (var i=0; i<colorItems.length; i++)
                setColor(colorItems[i],eval("_" + colorItems[i]));

        setStyle(_nameStyle);
        setFont("numFont",_numFont);
        setFont("homePlateFont",_homePlateFont);
        setLogo(_homePlateLetter);

        setSize("numFrontSize",_numFrontSize);
        setSize("numBackSize",_numBackSize);

        setShow("homePlate",_homePlate);
        setShow("rumpLogo",_rumpLogo);
        setShow("shortLogo",_shortLogo);
}

function initUser(aUniform)
{
        index = aUniform.index;

        _uniNum = aUniform.uniNum;
        _saveName = aUniform.saveName;
        _saveSchool = aUniform.saveSchool;
        _saveCity = aUniform.saveCity;
        _saveState = aUniform.saveState;

        isSaved = true;
        if ((_saveName.length == 0) ||
                (_saveSchool.length == 0) ||
                (_saveCity.length == 0) ||
                (_saveState.length < 2))
        {
                isSaved = false;
        }

        _teamColor1 = aUniform.teamColor1;
        _teamColor2 = aUniform.teamColor2;
        _teamColor3 = aUniform.teamColor3;
        _teamColor4 = aUniform.teamColor4;

        _uniformStyle = aUniform.uniformStyle;
        _waistband = aUniform.waistband;
        _fabric = aUniform.fabric;
        _bodyColor = aUniform.bodyColor;
        _side1Color = aUniform.side1Color;
        _side2Color = aUniform.side2Color;
        _side3Color = aUniform.side3Color;
        _trim1Color = aUniform.trim1Color;
        _trim2Color = aUniform.trim2Color;
        _trim3Color = aUniform.trim3Color;
        _numFont = aUniform.numFont;
        _numFrontSize = aUniform.numFrontSize;
        _numBackSize = aUniform.numBackSize;
        _numMain = aUniform.numMain;
        _numAccentColor = aUniform.numAccentColor;
        _numAccent = aUniform.numAccent;
        _numApp = aUniform.numApp;
        _nameStyle = aUniform.nameStyle;
        _nameTop = aUniform.nameTop;
        _nameBottom = aUniform.nameBottom;
        _nameFont = aUniform.nameFont;
        _nameSize = aUniform.nameSize;
        _nameMain = aUniform.nameMain;
        _nameAccentColor = aUniform.nameAccentColor;
        _nameAccent = aUniform.nameAccent;
        _nameApp = aUniform.nameApp;
        _homePlateFont = aUniform.homePlateFont;
        _homePlateColor = aUniform.homePlateColor;
        _homePlateLetter = aUniform.homePlateLetter;
        _homePlate = aUniform.homePlate;
        _shortLogo = aUniform.shortLogo;
        _rumpLogo = aUniform.rumpLogo;

        initialize();
        previewUser();
}

//for (stuff in _root) { if (stuff.charAt(0)=='_') trace(stuff); }
//initialize setup based on defaults
var _teamColor1,_teamColor2,_teamColor3,_teamColor4;
var _uniformStyle,_fabric,_waistband,_bodyColor,_side1Color,_side2Color,_side3Color,_trim1Color,_trim2Color,_trim3Color;
var _numFont,_numFrontSize,_numBackSize,_numMain,_numAccentColor,_numAccent,_numApp;
var _nameStyle,_nameTop,_nameBottom,_nameFont,_nameSize,_nameMain,_nameAccentColor,_nameAccent,_nameApp;
var _homePlateColor,_homePlateFont,_homePlateLetter,_homePlate,_shortLogo,_rumpLogo;
var _uniNum, _saveName;

gotoAndStop("start");
```

```
/******************************************************************************/
/* MENS PRICING/CODES                                                        */
/******************************************************************************/

isMen = true;

homePlatePrice = 2.40;

// garmentPrice [uniform][jersey..shorts + fabric]
garmentPrice = [
        [ 00.00 ],
        [ 29.00, 33.00, 31.00, 29.00, 36.00, 38.00 ],
        [ 31.00, 37.00, 32.00, 44.00, 51.00, 52.00, 38.00, 45.00, 47.00 ],
        [ 30.00, 35.00, 33.00, 43.00, 46.00, 48.00 ],
        [ 28.00, 34.00, 30.00, 36.00, 41.00, 43.00 ],
        [ 30.00, 36.00, 33.00, 40.00, 45.00, 47.00 ],
        [ 27.00, 32.00, 27.00, 30.00, 39.00, 41.00 ],
        [ 23.00, 29.00, 24.00, 31.00, 37.00, 39.00 ],
        [ 31.00, 38.00, 32.00, 42.00, 48.00, 50.00 ],
        [ 30.00, 37.00, 31.00, 29.00, 36.00, 38.00 ],
        [ 31.00, 36.00, 32.00, 28.00, 35.00, 37.00 ],
        [ 33.00, 38.00, 33.00, 44.00, 57.00, 59.00 ],
        [ 25.00, 29.00, 25.00, 39.00, 56.00, 59.00 ]
];

garmentCode = [
        [ "(DELETED)" ],
        [ "NBGM01888", "NBGM01953", "NBGM03607", "NRGM01898", "NRGM01957", "NRGM03615" ],
        [ "NBGM01889", "NBGM01954", "NBGM03610", "NRGM01899", "NRGM01958", "NRGM03618", "NRGM02023", "NRGM02024", "NRGM04290" ],
        [ "NBGM01891", "NBGM01956", "NBGM03606", "NRGM01902", "NRGM01960", "NRGM03614" ],
        [ "NBGM01895", "NBGM01892", "NBGM03608", "NRGM01906", "NRGM01903", "NRGM03616" ],
        [ "NBGM01894", "NBGM01893", "NBGM03609", "NRGM01905", "NRGM01904", "NRGM03617" ],
        [ "NBGM06642", "NBGM06644", "NBGM06643", "NRGM06645", "NRGM06647", "NRGM06646" ],
        [ "NBGM02094", "NBGM01897", "NBGM03611", "NRGM02115", "NRGM02116", "NRGM03619" ],
        [ "NBGM03690", "NBGM03688", "NBGM03689", "NRGM03696", "NRGM03694", "NRGM03695" ],
        [ "NBGM03693", "NBGM03691", "NBGM03692", "NRGM03699", "NRGM03697", "NRGM03698" ],
        [ "NBGM06654", "NBGM06656", "NBGM06655", "NRGM06657", "NRGM06659", "NRGM06658" ],
        [ "NBGM06636", "NBGM06638", "NBGM06637", "NRGM06639", "NRGM06641", "NRGM06640" ],
        [ "NBGM06648", "NBGM06650", "NBGM06649", "NRGM06651", "NRGM06653", "NRGM06652" ]
];

// [font][solid..accent]
numPrintPrice = [[ 2.30, 4.30 ],[ 2.40, 4.90 ]];
namePrintPrice = [ 2.30, 4.30 ];

// [size][solid,outline,shadow]
numTwillPrice = [
        [],[],[],[],
        [ 1.40, 2.40, 3.00 ],
        [ 1.50, 2.80, 3.20 ],
        [ 1.60, 3.00, 3.50 ],
        [],
        [ 2.10, 3.90, 5.20 ]
];

nameTwillPrice = [
        [],[],[],
        [ 1.40, 2.50, 2.60 ],
        [ 1.50, 2.80, 3.00 ]
];

uniformText = [
        "(DELETED)", "RED STORM", "CARDINAL","HARTFORD", "TUCSON",
        "COWBOY", "OWL", "(DELETED)", "BLUE DEVIL", "LEXINGTON",
        "INDY","CHAMPAIGN","JAYHAWK","LONGHORN"
];

uniformFrames = [
        "(Cinci)", "John", "SU", "UConn", "AZ",
        "OK", "Temple", "American", "Duke", "KY",
        "IN","Ill", "KS", "TX"
];

// uniform constants
STJOHN = 1;
STANFORD = 2;
UCONN = 3;
ARIZONA = 4;
OKLAHOMA = 5;
DUKE = 8;
KENTUCKY = 9;

CINCINNATI = 0;      // 01/03 deleted
```

```
AMERICAN = 7;        // 01/03 deleted
TEMPLE = 6;          // 01/03 updated
INDIANA = 10;        // 01/03 added
ILLINOIS = 11;       // 01/03 added
KANSAS = 12;         // 01/03 added
TEXAS = 13;          // 01/03 added

function hasWaistband()
{
        if ((_uniformStyle == CINCINNATI) || (_uniformStyle == STANFORD))
                return true;
        else
                return false;
}

function hasStock()
{
        if ((_uniformStyle == ARIZONA) || (_uniformStyle == STANFORD))
                return true;
        else
                return false;
}

function getEnabled(target)
{
        if ((target._name.indexOf("home") == 0) && (target != homePlateAvailable))
                target.setEnabled(_homePlate);
        else
                target.setEnabled(true);
}

function setEnabled(target,isEnabled)
{
        target.gotoAndStop(isEnabled?"on":"off");
        target.setOptions(NULL);
}


```

login

```
on (release, keyPress "<Enter>")
{
        user = _root.user;
        password = _root.password;

        if ((user == "") || (password == ""))
        {
                _root.msg = "User and password may not be empty.";
        }
        else
        {
//              getURL("login.asp", "_blank", "GET");
                _root.msg = "Logging in ...";
                loadVariables ("login.asp", _root.loginButton, "POST");
        }
}
```

```
onClipEvent(data)
{
        CusNum = parseInt(CusNum);
        if (CusNum >= 0)
        {
                _root.msg = "Login successful.";
                _root._cusNum = CusNum;

                thexml = new XML();
                thexml.onLoad = _root.getSavedList;
                thexml.load("load.asp");
                _root.start = getTimer();

                _root.play();
        }
        else
        {
                _root.msg = "Login unsuccessful.";
        }
}
```

textfield

```
onClipEvent (load)
{
        current = _root.getName(_parent._name);
        _parent.text = current;
}

onClipEvent (enterFrame)
{
        if (current == _parent.text)
                return;
        _parent.text = _parent.text.toUpperCase();
        for (var i=0; i<_parent.text.length; i++)
        {
                if ((_parent.text.charCodeAt(i) < _root.CODEA) || (_parent.text.charCodeAt(i) > _root.CODEZ))
                {
                        _parent.text = _parent.text.substr(0,i) + " " + _parent.text.substr(i+1);
                }
        }
        current = _parent.text;
        _root.setName(_parent._name,current);
}
```

save

```
on (release)
{
        // prevents saving multiple time while connecting to server
        if (!_root.saveMsg.done)
                return;

        saveName = _root._saveName;
        saveSchool = _root._saveSchool;
        saveCity = _root._saveCity;
        saveState = _root._saveState;

        if ((saveName.length == 0) ||
                (saveSchool.length == 0) ||
                (saveCity.length == 0) ||
                (saveState.length < 2))
        {
                _root.saveMsg.gotoAndPlay("validate");
                _root.isSaved = false;
                _root.reviewButton.setEnabled(false);
                return;
        }

        _root.saveMsg.done = false;
        _root.saveMsg.gotoAndPlay("saving");

        CusNum = _root._cusNum;
        UniNum = 0;

        if ((_root.index>=0) && (saveName == _root.savedUniforms[_root.index].saveName))
                UniNum = _root.savedUniforms[_root.index].UniNum;

        if (UniNum == 0)
                _root.index = -1;

        teamColor1 = _root._teamColor1;
        teamColor2 = _root._teamColor2;
        teamColor3 = _root._teamColor3;
        teamColor4 = _root._teamColor4;

        uniformStyle = _root._uniformStyle;
        waistband = _root._waistband;
        fabric = _root._fabric;
        bodyColor = _root._bodyColor;
        side1Color = _root._side1Color;
        side2Color = _root._side2Color;
        side3Color = _root._side3Color;
        trim1Color = _root._trim1Color;
        trim2Color = _root._trim2Color;
        trim3Color = _root._trim3Color;
        numFont = _root._numFont;
        numFrontSize = _root._numFrontSize;
        numBackSize = _root._numBackSize;
        numMain = _root._numMain;
        numAccentColor = _root._numAccentColor;
        numAccent = _root._numAccent;
        numApp = _root._numApp;
        nameStyle = _root._nameStyle;
        nameTop = _root._nameTop;
        nameBottom = _root._nameBottom;
        nameFont = _root._nameFont;
        nameSize = _root._nameSize;
        nameMain = _root._nameMain;
        nameAccentColor = _root._nameAccentColor;
        nameAccent = _root._nameAccent;
        nameApp = _root._nameApp;
        homePlateFont = _root._homePlateFont;
        homePlateColor = _root._homePlateColor;
        homePlateLetter = _root._homePlateLetter;
        homePlate = _root._homePlate;
        shortLogo = _root._shortLogo;
        rumpLogo = _root._rumpLogo;

//      getURL("save.asp", "_blank", "GET");
        loadVariables("save.asp", _root.saveButton, "POST");
}
```

```
onClipEvent(data)
{
        saved = (saved.indexOf("true")>=0);
        UniNum = parseInt(UniNum);

        _root.saveMsg.saved = saved?"true":"false";
        _root.saveMsg.done = true;
        _root.isSaved = true;
        _root.reviewButton.setEnabled(true);

        if (saved)
        {
                theUniform = new Array();
                for (var prop in this)
                {
                        eval("theUniform."+prop) = eval(prop);
                }

                if (_root.index < 0)
                {
                        _root.index = theUniform.index = _root.savedUniforms.length;
                        _root._uniNum = theUniform.UniNum = UniNum;
                        _root.savedUniforms.push(theUniform);
                }
                else
                {
                        theUniform.index = _root.index;
                        _root.savedUniforms[_root.index] = theUniform;
                }
        }
}
```

print

```
stop();
if (!uniform._visible) previewUser();

function catalogFontName(font, accent)
{
        theFontName = fontNames[font];
        if (font == DOUBLE)
                theFontName = "FULL BLOCK OUTLINE";
        else if (accent == SHADOW)
        {
                if (font == GOTHIC)
                        theFontName = "ATHLETIC GOTHIC SHADOW";
                else if (font == NIKEBLOCK)
                        theFontName = "ATHLETIC SHADOW";
                else if (font == FULL)
                        theFontName = "FULL BLOCK SHADOW";
        }

        theFontName += " ";
        theFontName += (accent == NONE) ? "1C" : "2C";
        return theFontName;
}

function getSpecs()
{
// uniform specs
//styleSpec = "Style / "+uniformText[_uniform];
jerseySpec = "Jersey Style / #"+garmentCode[_uniformStyle][_fabric];
shortsSpec = "Shorts Style / #"+garmentCode[_uniformStyle][_fabric+3*(_waistband+1)];
fabricSpec = "Fabric / "+fabricStyles[_fabric];
bodySpec = "Color / "+colorText[_bodyColor];
insertSpec = "Inserts / "+colorText[_side1Color]+
        (_side2Color?"-"+colorText[_side2Color]:"");
trimSpec = "Trims / "+colorText[_trim1Color]+
        (_trim2Color?"-"+colorText[_trim2Color]:"");

// number specs
numFontSpec = "Font / "+catalogFontName(_numFont,_numAccent);
numSizeSpec = "Size / Front-"+_numFrontSize+"\" Back-"+_numBackSize+"\"";
numAppSpec = "Application / "+((_numApp==PRINT)?"PRINT":"TACKLE TWILL");
numColorSpec = "Colors / "+colorText[_numMain]+
        (_numAccentColor?"-"+colorText[_numAccentColor]:"");

// team name specs
nameSpec = "Name / "+_nameTop+
        (_nameStyle==BOTTOM?" "+_nameBottom:"");
nameStyleSpec = "Style / " + nameStyles[_nameStyle];
nameFontSpec = "Font / "+catalogFontName(_nameFont,_nameAccent);
nameSizeSpec = "Size / "+_nameSize+"\"";
nameAppSpec = "Application / "+((_nameApp==PRINT)?"PRINT":"TACKLE TWILL");
nameColorSpec = "Colors / "+colorText[_nameMain]+
        (_nameAccentColor?"-"+colorText[_nameAccentColor]:"");

// custom logo specs
homeSpec = "Homeplate / "+(_homePlate?(colorText[_homePlateColor]+"-"+_homePlateLetter):"NO");
homeFontSpec = "Font / "+(_homePlate?fontNames[_homePlateFont]+" SOLID":"NONE");
logoSpec = "Logos / Shorts-"+(_shortLogo?"YES":"NO")+" Rump-"+(_rumpLogo?"YES":"NO");
}

getSpecs();
```

color palette

```
stop();

function setEnabled(isEnabled)
{
       gotoAndStop(isEnabled?"on":"off");

       buttonArray = new Array(color1, color2, color3, color4);
       frameArray = new Array(frame1, frame2, frame3, frame4);
       theColor = _root.getColor(_name);
       selected = null;

       for (i=0; i<buttonArray.length; i++)
       {
              button = buttonArray[i];
              button.theColor = _root.teamColors[i];
              _root.setObjectColor(button,_root.teamColors[i]);

              if (_root.teamColors[i] == _root.NONE)
              {
                     button._visible = false;
                     frameArray[i]._visible = false;
              }
              else if (theColor == _root.teamColors[i])
              {
                     button._visible = true;
                     frameArray[i]._visible = true;
                     selected = button;
              }
       }

       if (isEnabled && selected != null)
       {
              hilite._visible = true;
              hilite._x = selected._x;
              hilite._y = selected._y;
              text = _root.colorText[theColor];
       }
       else
       {
              hilite._visible = false;
       }
}

_root.getEnabled(this);
```