```
Security.allowDomain("*");
Security.allowInsecureDomain("*");
Security.allowDomain(this.root.loaderInfo.loaderURL)
import flash.external.*;
stage.scaleMode ="noScale";
var urlPath:String = new String();
var urlSender = this.loaderInfo.url;
var urlNext:String = new String();
function geturlhttp() {
urlPath = ExternalInterface.call("window.location.href.toString");
}
geturlhttp();
var urlString = new String;
urlString = urlSender;
var curseWrd:RegExp = /survey.swf/;
urlString = urlString.replace(curseWrd, "ai_start.php?dltesting=1&site=1&mode=xml");
urltxt.text ='I came from \n' + urlSender + '\n\n'+ 'I am be loaded into \n' + urlPath
                +  '\n\n' +'I am Getting from \n' + urlString +  '\n\n' +'I will next goto \n' + urlNext
                +  '\n\n' + survey;
import flash.events.MouseEvent;
import flash.events.TextEvent;
import flash.events.*;
import flash.display.*;
import fl.controls.RadioButton;
import fl.controls.CheckBox;
import fl.controls.RadioButtonGroup;
import flash.text.TextFieldAutoSize;
import flash.text.GridFitType;
import fl.containers.ScrollPane;
import flash.text.TextFormat;
import flash.net.URLRequest;
import flash.text.Font;

var survey:XML = <survey/>;
XML.ignoreWhitespace = true;
var surveyLoader:URLLoader = new URLLoader();
surveyLoader.addEventListener(Event.COMPLETE, onLoadXML);
//var xmlReq:URLRequest = new URLRequest(urlString);
var xmlReq:URLRequest = new URLRequest("../xml/re538452.xml");
surveyLoader.load(xmlReq);
var anslist:Array = new Array ();
var radArray:Array = new Array();
var txtArray:Array = new Array();
var chkArray:Array = new Array();
var txtString:String = new String();
var questionFields:TextField = new TextField();
```

```
var answer:TextField = new TextField();
var errorFields:TextField = new TextField();
var option:TextField = new TextField();
var checkB:CheckBox = new CheckBox();
var radioB:RadioButton = new RadioButton();
var rGroup:RadioButtonGroup = new RadioButtonGroup("group");
var txtAns:Object = new Object();
var key:Object = new Object();
var emUnit:uint = 8;
var txtFrmt:TextFormat = new TextFormat();
var errFrmt:TextFormat = new TextFormat();
var surveyFont:SurveyFont = new SurveyFont();
var errFont:ErrFont = new ErrFont();
errFrmt.font = errFont.fontName;
errFrmt.color = 0xeeb661;
errFrmt.bold = false;
errFrmt.leading = 5;
errFrmt.letterSpacing = 1.5;
txtFrmt.font = surveyFont.fontName;
txtFrmt.color = 0x0b2743;
txtFrmt.bold = true;
txtFrmt.italic = false;
txtFrmt.underline = false;
txtFrmt.size = 15;
txtFrmt.leading = 3;
txtFrmt.letterSpacing = 0.5;
var pos = int;
pos = 0;
var pad = int;
var lastPos:int;
lastPos = 0;
var typeUI = Boolean;

stage.stageFocusRect = false;
urltxt.focusRect =false;
urltxt.embedFonts=true;
function onLoadXML(e:Event):void
{
    survey = new XML(surveyLoader.data);
    var qNum:int;
    typeUI = true;
    qNum = survey.itemList.question.length();
    err.visible=false;
    logo.visible=true;
    //===============Image Loading======================
    /*var backldr:Loader = new Loader();
```

```
93       clear_mc.addChild(backldr);
94       var backurl:String = "../../"+ survey.imgPath + "background.jpg";
95       var backReq:URLRequest = new URLRequest(backurl);
96       backldr.load(backReq);
97
98      var logoLdr:Loader = new Loader();
99      logo.addChild(logoLdr);
100      var logourl:String = "../../"+ survey.imgPath + "logo.jpg";
101      var logoReq:URLRequest = new URLRequest(logourl);
102      logoLdr.load(logoReq);
103
104     var submitldr:Loader = new Loader();
105      submit.addChild(submitldr);
106      var submiturl:String = "../../"+ survey.imgPath + "submit.jpg";
107      var submitReq:URLRequest = new URLRequest(submiturl);
108      submitldr.load(submitReq);
109
110     var closeLdr:Loader = new Loader();
111     closeBtn.addChild(closeLdr);
112      var closeurl:String = "../../"+ survey.imgPath + "close.jpg";
113      var closeReq:URLRequest = new URLRequest(closeurl);
114      closeLdr.load(closeReq);
115
116     var errorldr:Loader = new Loader();
117      err.addChild(errorldr);
118      var errorurl:String = "../../"+ survey.imgPath + "error.png";
119      var errorReq:URLRequest = new URLRequest(errorurl);
120      errorldr.load(errorReq);*/
121     //=======================================
122
123     var mc:MovieClip = new MovieClip();
124         addChild(mc);
125     for (var k:uint = 0; k < qNum; k++)
126     {
127         var thisQuest:Object = new Object();
128         thisQuest = survey.itemList.question[k];
129         questionFields = new TextField();
130         questionFields.type = TextFieldType.DYNAMIC;
131         questionFields.x = 10;
132         questionFields.y = mc.y + k + mc.height;
133         if (typeUI == false){
134             questionFields.y = mc.y + k + mc.height + 50;
135         }
136         else{
137             }
138         questionFields.autoSize = TextFieldAutoSize.LEFT;
```

```
        questionFields.width = 400;
        questionFields.multiline = true;
        questionFields.wordWrap = true;
        questionFields.embedFonts=true;
        if (thisQuest.@type == "htmlElement")
        {
            questionFields.text = "";
        }
        else if (thisQuest.@type == "error")
        {
            questionFields.text = "";
            questionFields.text = survey.itemList.question.@html_text[e];
            doERROR(thisQuest);
        }
        else
        {
        questionFields.htmlText = survey.itemList.question. @ question_text[k];
        }
        mc.addChild(questionFields);
        questionFields.setTextFormat(txtFrmt);
        vScroll.source = mc;
        vScroll.x = 0;
        vScroll.y = 80;
        vScroll.height = 195;
        vScroll.width = 480;
        vScroll.update();
        vScroll.verticalScrollPosition = 0;
        if (thisQuest. @ type == 'textLine')
        {
            doTEXT(thisQuest);
        }
        else if (thisQuest. @ type == 'textArea')
        {
            doTEXT(thisQuest);
        }
        else if (thisQuest. @ type == 'multChoiceCheckbox')
        {
            doCHECK(thisQuest);
        }
        else if (thisQuest.@type=='multChoiceRadio')
        {
            doRADIO(thisQuest);
        }
        else if (thisQuest.@type=='multChoiceDetailRadio')
        {
            doRADIO(thisQuest);
```

```
185         }
186         else if (thisQuest.@type=='htmlElement')
187         {
188             doHTML(thisQuest);
189         }
190         else
191         {
192         }
193         function doHTML(thisQuest):void
194         {
195             if (thisQuest. @ type == "htmlElement")
196             {
197                 option = new TextField();
198                 option.type = TextFieldType.DYNAMIC;
199                 for (var h:uint = 0; h < thisQuest.length(); h++)
200                 {
201                     option.htmlText = thisQuest.@html_text;
202                     option.x = questionFields.height;
203                     option.y = questionFields.y + questionFields.height;
204                     mc.addChild(option);
205                     option.width = option.textWidth + questionFields.height;
206                 }
207             }
208             typeUI = false;
209         }
210 
211         //========CHECK=====10/02/10===');
212         function doCHECK(thisQuest):void
213         {
214             pad = 0
215             lastPos = 0;
216 
217             if (thisQuest. @ type == "multChoiceCheckbox")
218             {
219                 var key:Object = new Object();
220                 key = survey.itemList.question[k]. @ base_html_name[c];
221                 var num:Object = new Object();
222                 pos = 0
223                         checkB = new CheckBox();
224                         checkB.setStyle("textFormat", txtFrmt);
225                         checkB.textField.multiline = true;
226                         checkB.textField.wordWrap = true;
227                         checkB.textField.width = 400;
228                         checkB.label = survey.itemList.question[k].answerRows.answer. @ row_text;
229 
230 
```

```
231                 pos = questionFields.y; // where to start calcuating base on where the question starts.
232
233                 var metric = int;
234                 metric = 27
235
236                 pos += questionFields.height + checkB.textField.numLines * checkB.textField.numLines;
237                 lastPos = pos;
238
239
240                 for (var c:uint = 0; c < thisQuest.answerRows.answer.length(); c++)
241                 {
242                     checkB.addEventListener(MouseEvent.CLICK, checked);
243                     num = thisQuest.answerRows.answer. @ html_value[c];
244                     checkB = new CheckBox();
245                     checkB.width = 400;
246                     mc.addChild(checkB);
247                     checkB.textField.multiline = true;
248                     checkB.textField.wordWrap = true;
249                     checkB.textField.width = 420;
250                     checkB.focusRect = false;
251                     checkB.setStyle("textFormat", txtFrmt);
252                     checkB.x = 10;
253                     checkB.label = survey.itemList.question[k].answerRows.answer. @ row_text[c];
254                     var hold:int = pad;
255                     pos = lastPos;
256                 //=================== 2nd ++ ====================
257                     checkB.y = pos + hold
258                     pad = checkB.height * checkB.textField.numLines  + 20;
259                     lastPos = checkB.y;
260                     mc.value = num;
261                     checkB.name = thisQuest.answerRows.answer. @ html_value[c];
262
263                     //checkB.addEventListener(MouseEvent.CLICK, checked);
264
265                     checkB.textField.embedFonts=true;
266                     mc.value = num;
267                     trace (checkB.selected);
268
269
270                 //}
271                         //=============end formatting===========
272             function checked(e: MouseEvent):void
273                 {
274                     trace();
275                     trace('checkedOn');
276                     var isOn:Boolean;
```

```
277                     var wait:String = new String;
278                     wait = "wating";
279                     //wait.push(key +'['+ e.target.name +']' + '[answer]=' + e.currentTarget.name + " " + isOn);
280                     //chkArray.push(key +'['+ e.target.name +']' + '[answer]=' + e.currentTarget.name + " " + isOn);
281                     //trace (wait);
282                     //trace ("====")
283 
284 
285                     trace(e.target.name + e.target.selected);
286 
287                     if (e.target.selected == true){
288 
289                         wait=  '['+ e.target.name +']' + '[answer]=' + e.currentTarget.name + " " + e.target.selected;
290                         chkArray.push(wait);
291                         chkArray.toString( );
292                     }
293                     else{
294                         wait.replace(key +'['+ e.target.name +']' + '[answer]=' + e.currentTarget.name + " " + e.targe
    t.selected);
295                         //chkArray.unshift(key +'['+ e.target.name +']' + '[answer]=' + e.currentTarget.name + " " +
    isOn);
296                     }
297                     trace (wait);
298                     trace (chkArray);
299 
300                     /*
301                     e.buttonDown = true;
302                     e.target.hasOwnProperty(true);
303                     trace ('e.buttonDown = '+ e.buttonDown);
304                     //checkB.selected
305 
306                             if(e.buttonDown == true)
307                             {
308                                 isOn = true;
309                                 trace ('isOn' + isOn);
310                                 checkB.toggle;
311                                 trace('toggled = '+checkB.toggle);
312                                 checkB.selected = true;
313                                 trace('selected = '+ checkB.selected);
314                                 trace('['+ e.target.name +']' + isOn);
315                                 trace(num);
316                                 checkB.removeEventListener(MouseEvent.CLICK,checked);
317                             }
318                             if ('['+ e.target.name +']' + isOn == true){
319                                 '['+ e.target.name +']' + !isOn
320                             }
```

```
321
322
323                 trace (e.currentTarget)
324                 //trace (checkB.selected);
325
326                         if (checkB.selected )
327                         {
328                             checkB.toggle;
329                             trace ('i know');
330                         }
331
332                 chkArray.push(key +'['+ e.target.name +']' + '[answer]=' + e.currentTarget.name + " " + isOn);
333                         /*if(isOn == true){
334                         isOn =!true;
335                         }
336                         if (key +'['+ e.target.name +']' + '[answer]=' + e.currentTarget.name + " " + isOn){
337
338                         }*/
339
340                         //trace('checked' + chkArray);
341
342                 checkB.removeEventListener(MouseEvent.CLICK,checked);
343
344             }
345
346
347         } }
348         typeUI = true;
349     }
350
351
352     //=============Radio======9/26/10=========================
353     function doRADIO(thisQuest):void
354     {
355         pad = 0
356         lastPos = 0;
357         if (thisQuest. @ type == "multChoiceRadio")
358         {
359             var key:Object = new Object();
360             key = survey.itemList.question[k]. @ base_html_name[r] + '[answer]' + '=';
361             rGroup = new RadioButtonGroup("group" + r);
362             pos = 0
363             radioB = new RadioButton();
364             radioB.setStyle("textFormat", txtFrmt);
365             radioB.textField.multiline = true;
366             radioB.textField.wordWrap = true;
```

```
                radioB.textField.width = 400;
                radioB.label = survey.itemList.question[k].answerRows.answer. @ row_text;
                pos = questionFields.y; // where to start calcuating base on where the question starts.
                var metric = int;
                metric = 27
                pos += questionFields.height + radioB.textField.numLines * radioB.textField.numLines;
                lastPos = pos;
                for (var r:uint = 0; r < thisQuest.answerRows.answer.length(); r++)
                {
                    radioB = new RadioButton();
                    radioB.width = stage.width;
                    mc.addChild(radioB);
                    radioB.setStyle("textFormat", txtFrmt);
                    radioB.textField.multiline = true;
                    radioB.textField.wordWrap = true;
                    radioB.textField.width = 400;
                    radioB.x = 10;
                    radioB.label = survey.itemList.question[k].answerRows.answer. @ row_text[r];
                    var hold:int = pad;
                    pos = lastPos;
                //==================== 2nd ++ ====================
                    radioB.y = pos + hold
                    pad = radioB.height * radioB.textField.numLines  + 20;
                    lastPos = radioB.y;
                    radioB.textField.embedFonts=true;
                    //==============end formatting===========
                    radioB.value = r + 1;
                    radioB.group = rGroup;
                    radioB.addEventListener(MouseEvent.CLICK, radioed);
                }

                function radioed(e: MouseEvent):void
                {
                    radArray.pop();
                    radArray.push(key + e.target.value);
                    trace();
                    trace('check  ' + anslist);
                    trace('radioed  ' + radArray);
                }
            }
            typeUI = true;
        }
        function doTEXT(thisQuest):void
        {
            var key:Object = new Object();
            key = survey.itemList.question[k]. @ base_html_name[i] + '=';
```

```
413             txtAns:Object;
414             if (survey.itemList.question. @ type[k] == "textLine" || "textArea")
415             {
416                 for (var i:int = 0; i < thisQuest.length(); i++)
417                 {
418                     answer = new TextField();
419                     answer.x = 20;
420                     answer.y = Math.round(questionFields.y + questionFields.height);
421                     answer.type = TextFieldType.INPUT;
422                     answer.background = true;
423                     answer.backgroundColor =0x0b2743;
424                     answer.border = false;
425                     answer.wordWrap = true;
426                     mc.addChild(answer);
427                     answer.defaultTextFormat = txtFrmt;
428                     answer.embedFonts=true;
429                     answer.textColor = 0xeeb661;
430                     if (survey.itemList.question. @ type[k] == "textLine")
431                     {
432                         answer.width = thisQuest.@text_line_size * emUnit + 10;
433                         answer.width < stage.width;
434                         answer.height = 20;
435                     }
436                     else if (survey.itemList.question. @ type[k] == "textArea")
437                     {
438                         answer.width = thisQuest.@textarea_columns * emUnit;// 16 pixels to 1em but 8 comes closer
    here;
439                         answer.height = thisQuest.@textarea_rows * emUnit * 1.8;
440                         answer.multiline = true;
441                     }
442                     else
443                     {
444                         answer.width = 300;
445                         answer.height = 20;
446                     }
447                     answer.addEventListener(TextEvent.TEXT_INPUT, newText);
448                     answer.addEventListener(FocusEvent.FOCUS_IN,fiHandler);
449                     answer.addEventListener(FocusEvent.FOCUS_OUT,foHandler);
450                     answer.addEventListener(FocusEvent.MOUSE_FOCUS_CHANGE, foHandler);
451                     answer.addEventListener(FocusEvent.KEY_FOCUS_CHANGE, foHandler);
452                     var _answer:String;
453                     function newText(event:TextEvent):void
454                     {
455                         if (! _answer)
456                         {
457                             _answer = "";
```

```
458                         }
459                         _answer +=  event.text;
460                     }
461                     function fiHandler (e:FocusEvent):void
462                     {
463 
464                         e.currentTarget.replaceSelectedText("");
465                         trace ('text answer ' + e.currentTarget.text);
466                     }
467                     function foHandler (e:FocusEvent):void
468                     {
469                         txtArray.pop();
470                         txtAns = answer.text;
471                         trace ('txtAns ' + txtAns);
472                         txtAns = e.currentTarget.text
473                         txtArray.push(key + txtAns);
474                         trace ('txtArray ' + txtArray );
475                         if (txtAns == '')
476                         {
477                             txtArray.pop();
478                         }
479                     }
480                 }
481                 typeUI = false;
482             }
483 
484         }
485         function doERROR(thisQuest):void
486         {
487             for (var e:uint = 0; e < qNum; e++)
488             {
489                 if (thisQuest. @ type == "error")
490                 {
491                     errorFields = new TextField();
492                     errorFields.x = 25 ;
493                     errorFields.y = questionFields.y + 15 * e + 40;
494                     errorFields.height = 20;
495                     errorFields.width = 400;
496                     errorFields.multiline = true;
497                     errorFields.wordWrap = true;
498                     errorFields.text = survey.itemList.question.@html_text[e];
499                     errorFields.setTextFormat(errFrmt);
500                     mc.addChild(errorFields);
501                     err.mouseEnabled=false;
502                     err.mouseChildren=false;
503                     err.visible=false;
```

```
                    err.visible=true;
                    err.alpha = .6;
                }
            }
        }
    }
    vScroll.refreshPane();
    vScroll.update();
    if (survey.continues.@type == "n")
    {
        submit.visible=false;
        closeBtn.visible=false;
        sub.visible=false;
        logo.visible=false;
        submit.removeEventListener(MouseEvent.CLICK, submitted);
        sub.removeEventListener(MouseEvent.CLICK, submitted);
        sub = null;
        logo = null;
        clo.x=420;
        clo.y=37;
         var doneurl:String = "../../"+ survey.imgPath + "done.jpg";
         var doneReq:URLRequest = new URLRequest(doneurl);
         //backldr.unload();
        //backldr.load(doneReq);
    }
    else{}
    urltxt.text ='I came from \n' + urlSender + '\n\n'+ 'I am be loaded into \n' + urlPath
                  +  '\n\n' +'I am Getting from \n' + xmlReq +  '\n\n' +'I will next goto \n' + urlNext
                  +  '\n\n' + survey + '\n\n'+ 'images come from \n'; //+ backurl;
    urltxt.alpha = 0;
    stage.focus = answer;
    trace(stage.focus);

}

submit.buttonMode=true;
this.setChildIndex(this.err, this.numChildren-1);
this.setChildIndex(sub, this.numChildren-1);
submit.addEventListener(MouseEvent.CLICK, submitted);
sub.addEventListener(MouseEvent.CLICK, submitted);
function submitted(e:MouseEvent):void
{
    pad = 0;
    pos = 0;
    this.setChildIndex(clear_mc, this.numChildren-1);
    this.setChildIndex(closeBtn, this.numChildren-1);
```

```
    this.setChildIndex(clo, this.numChildren-1);
    this.setChildIndex(logo, this.numChildren-1);
    this.setChildIndex(submit, this.numChildren-1);
    this.setChildIndex(err, this.numChildren-1);
    this.setChildIndex(sub, this.numChildren-1);
    this.setChildIndex(vScroll, this.numChildren-1);
    txtString = txtArray.join("&");
    anslist.push(txtString);
    anslist.push(radArray);
    anslist.push(chkArray);
    trace (anslist);

    var ansArray:Array = new Array();
    var ansString:String;
    ansArray = [];
    ansArray = anslist;
    ansArray.push("action=nextpage");
    ansString = ansArray.join("&");
    var curseWrd:RegExp = /ai_start.php/;
    trace("function = \n" + urlString.replace(curseWrd, "newai.php"));
    urlString = urlString.replace(curseWrd, "newai.php");
    trace('replacement = \n' + urlString);
    var xmlGet:URLRequest = new URLRequest(urlString + '&' +ansString);
    urlNext = urlString + '&' + ansString;
    trace('next ===' + urlNext);
    surveyLoader.load(xmlGet);
    anslist = [];
    this.txtAns = [];
    txtArray = [];
    clear_mc.x = clear_mc.y = 0;
    vScroll.refreshPane();
    vScroll.update();
    urltxt.text ='I came from \n' + urlSender + '\n\n'+ 'I am be loaded into \n' + urlPath
                  +  '\n\n' +'I am Getting from \n' + urlString +  '\n\n' +'I will next goto \n' + urlNext
                  +  '\n\n' + survey;
    errorFields.text = "";
    err.visible=false;
}
closeBtn.buttonMode= true;
closeBtn.useHandCursor = true;
closeBtn.addEventListener(MouseEvent.CLICK, exit);
function exit (e:MouseEvent):void
{
this.dispatchEvent(new Event('closeBtn_mc'));
}'
```