Accessibility for Flex and AIR Applications

Michael A. Jordan
Modernista!
@majornista

Daniel Koestler
Adobe
@antiChipotle

#adobemax155

Introduction

Why me?

- Working in Flash since version 5
- Took early interest in Flash accessibility after it was introduced with version 6
- Projects for Adobe around Flash and Flex accessibility
  1. FLVPlayback component skins with support for captioning
  2. Keyboard, screen-reader accessibility, and caption transcripts in FLVPlayback component in Flash CS4
  3. Accessible Video Demo as part of WCAG 2.0 Implementation Report
     http://www.w3.org/WAI/GL/WCAG20/implementation-report/implementation?implementation_id=68
  4. Documentation for flash.accessibility.AccessibilityImplementation class
     http://blogs.adobe.com/accessibility/flex/

Why do Flash accessibility?

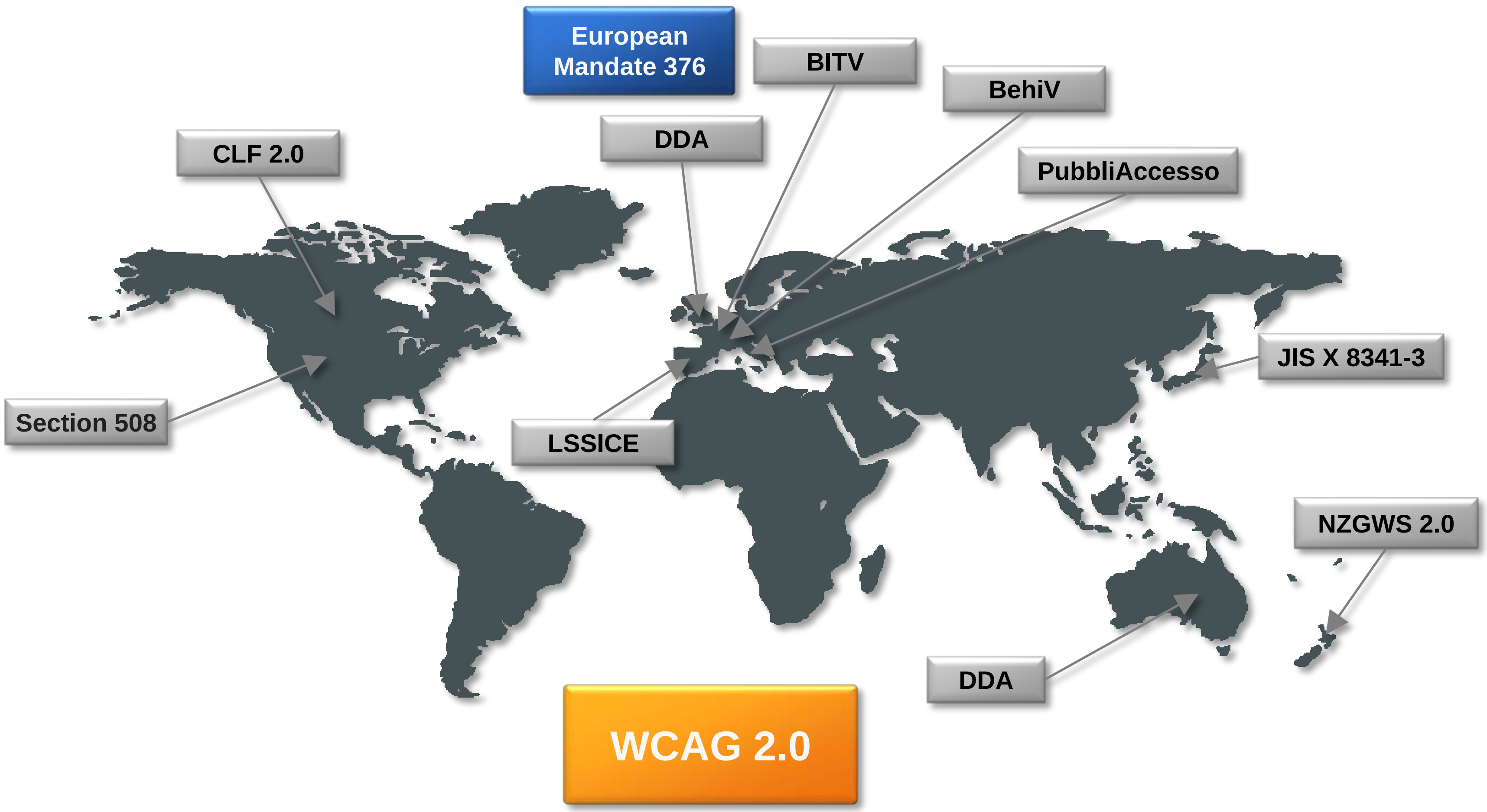
Moral Argument

“We don’t block out people for conditions they cannot readily change.”

—Chris Heilmann
Yahoo!

Why do Flash accessibility?

Laws and Standards



Why do Flash accessibility?

Laws and Standards

- WCAG 2

The Web Content Accessibility Guidelines documents explain how to make Web content accessible to people with disabilities.

There are 12 guidelines organized around 4 principles.

Why do Flash accessibility?

Laws and Standards

- WCAG 2

1. Perceivable

    - Provide **text alternatives** for non-text content.
    - Provide **captions and alternatives** for audio and video content.
    - Make content **adaptable**; and make it **available** to assistive technologies.
    - Use **sufficient contrast** to make things easy to see and hear.

Why do Flash accessibility?

Laws and Standards

- WCAG 2

1. Perceivable
   - Provide **text alternatives** for non-text content.
   - Provide **captions and alternatives** for audio and video content.
   - Make content **adaptable**; and make it **available** to assistive technologies.
   - Use **sufficient contrast** to make things easy to see and hear.
2. Operable
   - Make all functionality **keyboard accessible**.
   - Give users **enough time** to read and use content.
   - Do not use content that causes **seizures**.
   - Help users **navigate and find** content.

Why do Flash accessibility?

Laws and Standards

- WCAG 2

1. Perceivable
   - Provide **text alternatives** for non-text content.
   - Provide **captions and alternatives** for audio and video content.
   - Make content **adaptable**; and make it **available** to assistive technologies.
   - Use **sufficient contrast** to make things easy to see and hear.
2. Operable
   - Make all functionality **keyboard accessible**.
   - Give users **enough time** to read and use content.
   - Do not use content that causes **seizures**.
   - Help users **navigate and find** content.
3. Understandable
   - Make text **readable and understandable**.
   - Make content appear and operate in **predictable** ways.
   - Help users **avoid and correct mistakes**.

Why do Flash accessibility?

Laws and Standards

- WCAG 2

1. Perceivable
   - Provide **text alternatives** for non-text content.
   - Provide **captions and alternatives** for audio and video content.
   - Make content **adaptable**; and make it **available** to assistive technologies.
   - Use **sufficient contrast** to make things easy to see and hear.
2. Operable
   - Make all functionality **keyboard accessible**.
   - Give users **enough time** to read and use content.
   - Do not use content that causes **seizures**.
   - Help users **navigate and find** content.
3. Understandable
   - Make text **readable and understandable**.
   - Make content appear and operate in **predictable** ways.
   - Help users **avoid and correct mistakes**.
4. Robust
   - Maximize **compatibility** with current and future technologies.

Why do Flash accessibility?

Making things work for people

- Technically satisfying
- If you were to test-drive a car that worked as poorly, in certain reasonable scenarios, as the Flash web site that marketed it, would you buy the car?
- Competition from other technologies.
  - Javascript: Dojo, jQuery, YUI all implementing WAI-ARIA
  - Silverlight: No seriously.
  - With AIR, desktop applications.

Brief History of Flash Accessibility

Before Flash 6

- Not much.
- Developers had to “roll their own” rudimentary keyboard focus and tab order.
- No support for assistive technology.

Brief History of Flash Accessibility

Flash 6 (March 2002), Actionscript 1 and 2

- **`tabIndex`**, **`tabEnabled`**, and **`tabChildren`**
- Support for assistive technology through Microsoft Active Accessibility (MSAA) API
- **`_accProps`**
- **`Accessibility.isActive()`**
- **`Accessibility.updateProperties()`**
- **`Accessibility.sendEvent()`**
- **`System.capabilities.hasAccessibility`**
- Accessible components in AS1, AS2 and Flex 1.5
- Undocumented **`_accImpl`** object

Brief History of Flash Accessibility

Actionscript 3

- **`flash.accessibility.*`** package
- **`flash.accessibility.Accessibility`**
- **`_accProps`** becomes **`flash.accessibility.AccessibilityProperties`**
- Undocumented **`_accImpl`** becomes recently documented **`flash.accessibility.AccessibilityImplementation`**

Brief History of Flash Accessibility

AIR 2.0

Introduces support for assistive technologies through MSAA for AIR applications built in Flex or Flash.

Accessibility in Flex

28 Accessible Flex 3 Components

- Accordion
- AdvancedDataGrid
- Alert
- Button
- CheckBox
- ColorPicker
- ComboBox
- DataGrid
- DateChooser
- DateField
- Form
- Image
- Label
- LinkButton
- List
- Menu
- MenuBar
- Panel
- RadioButton
- RadioButtonGroup
- Slider
- TabNavigator
- Text
- TextArea
- TextInput
- TitleWindow
- ToolTipManager
- Tree

Accessibility in Flex

Tab order and Reading order

`flash.display.InteractiveObject`

- **`.tabIndex`** : uint
  Specifies the tab ordering of objects in a SWF file.

- **`.tabEnabled`** : Boolean
  Specifies whether a particular object is in the tab order.

`flash.display.DisplayObjectContainer`

- **`.tabChildren`** : Boolean
  Determines whether the children of an object are tab enabled.

Accessibility in Flex

Tab order and Reading order

- In MXML:

```
<mx:VBox id="custInfo" label="Customer Info" fontWeight="bold"
    horizontalAlign="center">

    <mx:Label text="Customer Info" textAlign="center"

            tabIndex="1" />

    <mx:HBox horizontalAlign="center">

            <mx:Label text="Email Address"

                tabIndex="2" />

            <mx:TextInput id="email"

                tabIndex="3" />

            <mx:Button id="emailSubmit" label="Submit"

                tabIndex="4" />

    </mx:HBox>

</mx:VBox>
```

Accessibility in Flex

Tab order and Reading order

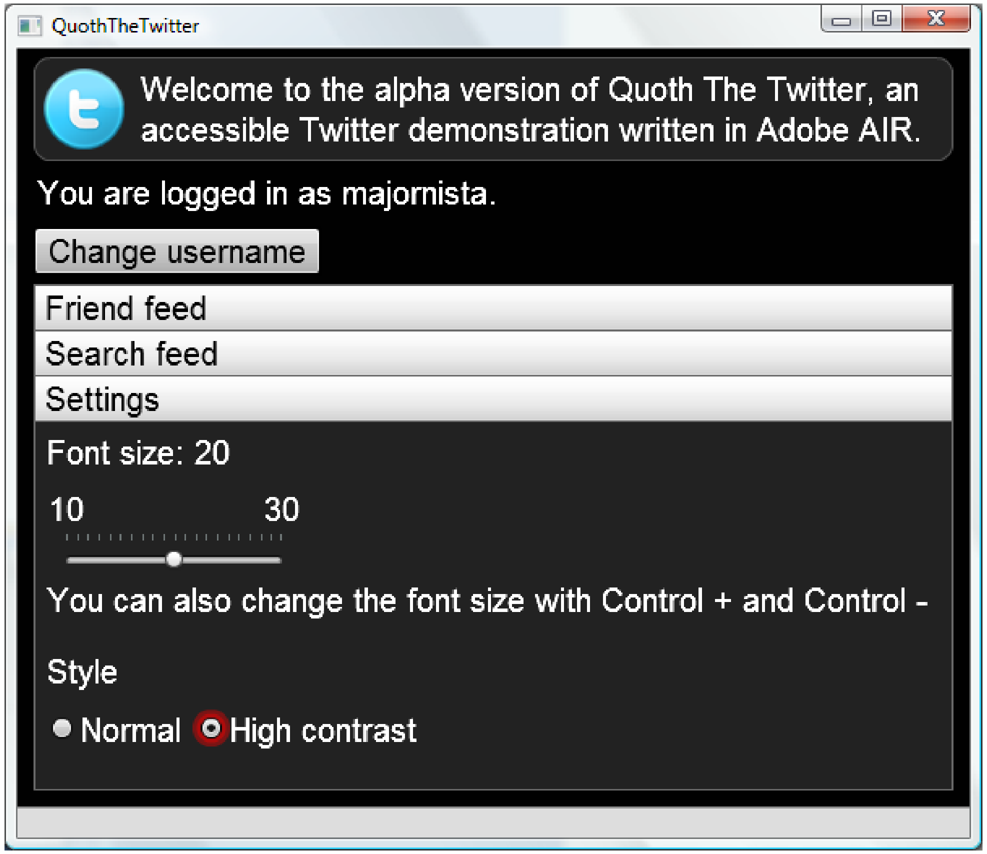
- In ActionScript:

```
private var nextTabIndex:uint = 1;

private function application_creationComplete(evt:FlexEvent):void {
    nextTabIndex = assignTabIndexes(panel as InteractiveObject);
}

private function assignTabIndexes(component:InteractiveObject):uint {
    component.tabIndex = nextTabIndex++;
    if(component is Panel) {
        use namespace mx_internal;
        var titleBar:UIComponent = Panel(component).getTitleBar() as UIComponent;
        if(titleBar) titleBar.tabIndex = nextTabIndex++;
    }
    if(component is Container){
        var children:Array = Container(container).getChildren();
        for each(var child:* in children){
            if(child is InteractiveObject){
                 nextTabIndex = assignTabIndexes(child as InteractiveObject);
            }
        }
    }
    return nextTabIndex++;
}
```

Accessibility in Flex

Adjustable Interface

- Resizable text
- High Contrast Mode
- Save Preferences



Welcome to the alpha version of Quoth The Twitter, an accessible Twitter demonstration written in Adobe AIR.

You are logged in as majornista.

Font size: 20

You can also change the font size with Control + and Control -

Style

Normal [✓ High contrast]

Accessibility in Flex

`flash.accessibility.AccessibilityProperties`

- **`.name`** : String
  Provides a name for this display object in the accessible presentation.

- **`.description`** : String
  Provides a description for this display object in the accessible presentation.

- **`.forceSimple`** : Boolean
  If true, causes Flash Player to exclude child objects within this display object from the accessible presentation.

- **`.silent`** : Boolean
  If true, excludes this display object from accessible presentation.

- **`.shortcut`** : String
  Indicates a keyboard shortcut associated with this display object.

Accessibility in Flex

flash.accessibility.AccessibilityProperties

- In MXML:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application
    xmlns:accessibility="flash.accessibility.*"
    xmlns:mx="http://www.adobe.com/2006/mxml">

    <accessibility:AccessibilityProperties
        id="searchInputAccProps"
        name="search Adobe.com…" />

    <mx:HBox width="100%" height="100%"
         horizontalAlign="center" verticalAlign="middle">

       <mx:TextInput id="searchInput"
                  accessibilityProperties="{searchInputAccProps}" />

       <mx:Button id="searchSubmit"
         label="search" />

    </mx:HBox>

</mx:Application>
```

Accessibility in Flex

flash.accessibility.AccessibilityProperties

- In ActionScript:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute">
    <mx:Script>
    <![CDATA[

        import mx.events.FlexEvent;

        private function onCreationComplete(evt:FlexEvent):void {
            evt.target.accessibilityProperties = new AccessibilityProperties();
            evt.target.accessibilityProperties.name = "search Adobe.com…";
            Accessibility.updateProperties();
        }

    ]]>
    </mx:Script>
    <mx:HBox width="100%" height="100%" horizontalAlign="center" verticalAlign="middle" >
       <mx:TextInput id="searchInput"
         creationComplete="onCreationComplete(event)" />
       <mx:Button id="searchSubmit" label="search" />
    </mx:HBox>
</mx:Application>
```

Accessibility in Flex

`flash.accessibility.Accessibility`

- **`.active`** : Boolean
  [static] [read-only] Indicates whether a screen reader is currently active and the player is communicating with it.

- **`.updateProperties()`** : void
  [static] Tells Flash Player to apply any accessibility changes made by using the DisplayObject.accessibilityProperties property.

- **`.sendEvent(source:DisplayObject, childID:uint, eventType:uint, nonHTML:Boolean = false)`** : void
  [static] Sends an event to the Microsoft Active Accessibility API.

Accessibility in Flex

Video and Audio Players

- Captions for all prerecorded audio and synchronized media
- Video controls should be perceivable and operable
- Audio controls should be perceivable and operable

Accessibility in Flex

Video and Audio Players

[Demo]

Building Accessible Custom Controls

First Steps

- Extend existing accessible components where possible
or build on what's already there
- Start with keyboard accessibility
  - Build to expected keyboard behavior
  http://msdn.microsoft.com/en-us/library/ms971323.aspx#atg_keyboardshortcuts_dialog_box_and_common_controls_shortcut_keys
- Focus management

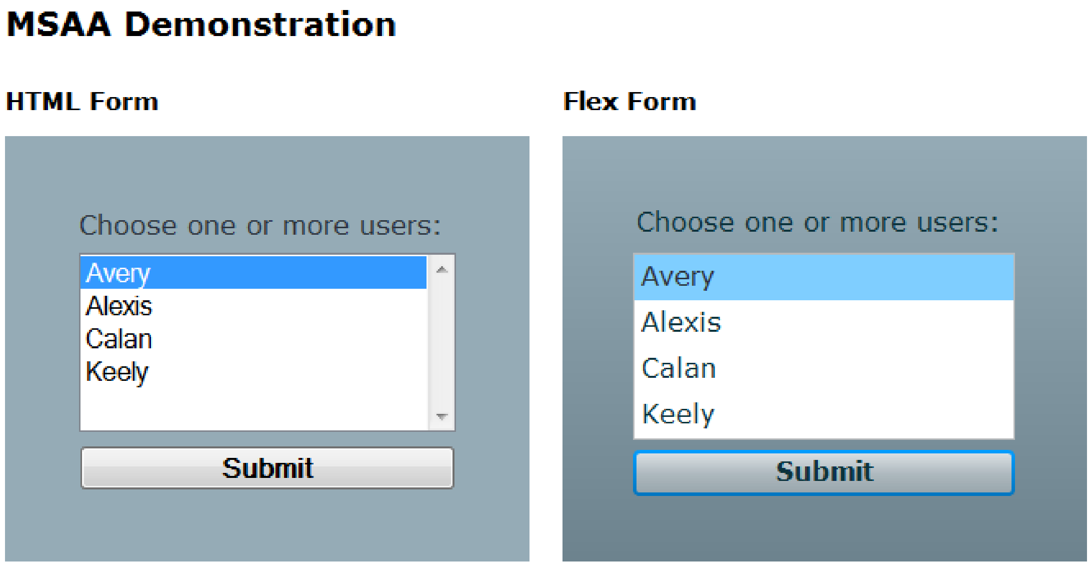
FlashPlayer and Microsoft Active Accessibility

How MSAA Works

- MSAA provides descriptive and standardized way for applications and screen readers to communicate on Windows operating systems.
- MSAA conveys an object model of an application.
- Get the tools **Active Accessibility 2.0 SDK Tools**: http://www.microsoft.com/downloads/details.aspx?familyid=3755582A-A707-460A-BF21-1373316E13F0&displaylang=en

FlashPlayer and Microsoft Active Accessibility

How MSAA Works



MSAA Demonstration

HTML Form

Choose one or more users:

- Avery
- Alexis
- Calan
- Keely

Submit

Flex Form

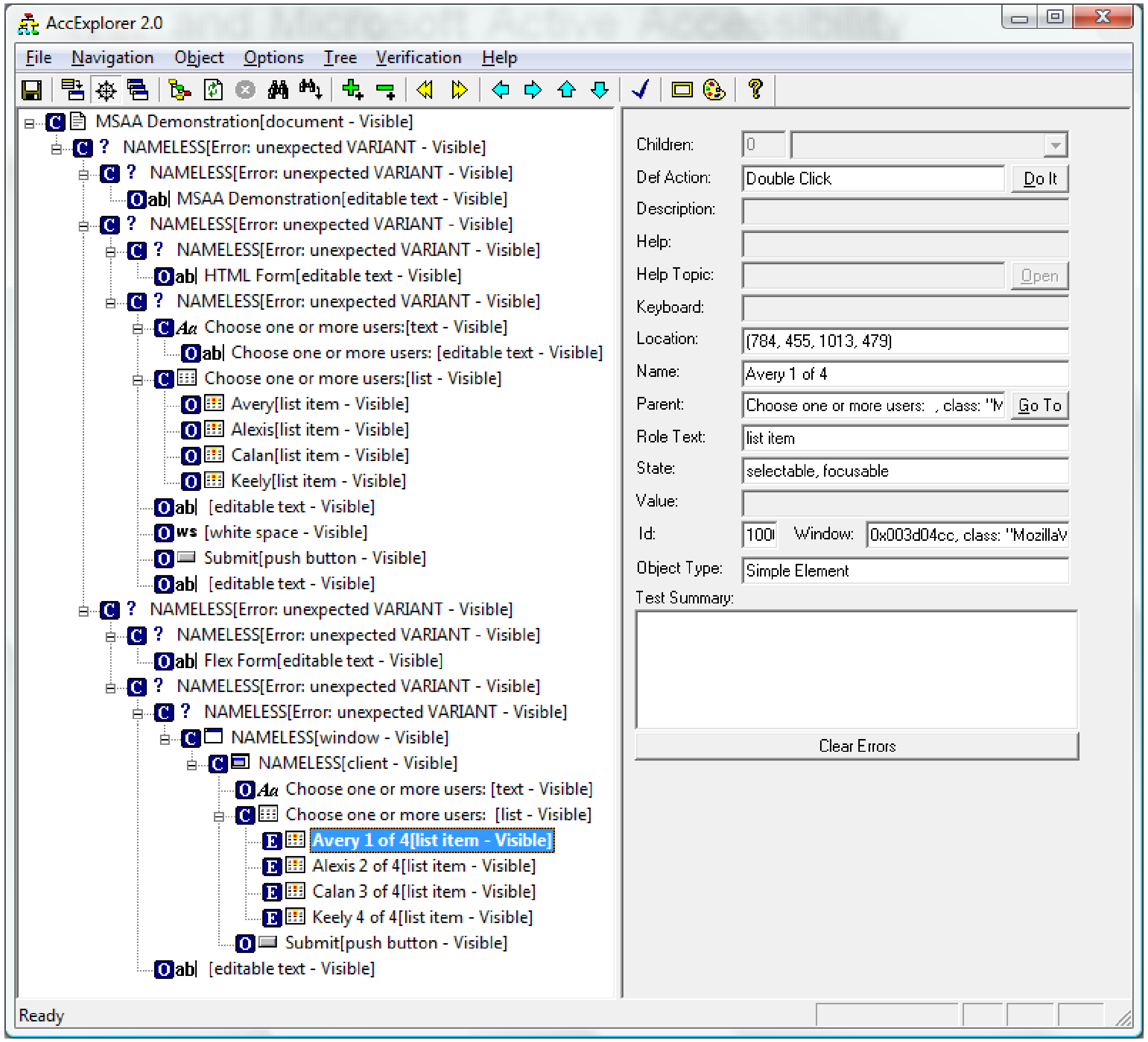
Choose one or more users:

- Avery
- Alexis
- Calan
- Keely

Submit

FlashPlayer and Microsoft Active Accessibility

How MSAA Works


AccExplorer 2.0

```
MSAA Demonstration[document - Visible]
  NAMELESS[Error: unexpected VARIANT - Visible]
    NAMELESS[Error: unexpected VARIANT - Visible]
      MSAA Demonstration[editable text - Visible]
    NAMELESS[Error: unexpected VARIANT - Visible]
      NAMELESS[Error: unexpected VARIANT - Visible]
        HTML Form[editable text - Visible]
      NAMELESS[Error: unexpected VARIANT - Visible]
        Choose one or more users:[text - Visible]
          Choose one or more users: [editable text - Visible]
        Choose one or more users:[list - Visible]
          Avery[list item - Visible]
          Alexis[list item - Visible]
          Calan[list item - Visible]
          Keely[list item - Visible]
        [editable text - Visible]
        [white space - Visible]
        Submit[push button - Visible]
        [editable text - Visible]
    NAMELESS[Error: unexpected VARIANT - Visible]
      NAMELESS[Error: unexpected VARIANT - Visible]
        Flex Form[editable text - Visible]
      NAMELESS[Error: unexpected VARIANT - Visible]
        NAMELESS[Error: unexpected VARIANT - Visible]
          NAMELESS[window - Visible]
            NAMELESS[client - Visible]
              Choose one or more users: [text - Visible]
              Choose one or more users:  [list - Visible]
                Avery 1 of 4[list item - Visible]
                Alexis 2 of 4[list item - Visible]
                Calan 3 of 4[list item - Visible]
                Keely 4 of 4[list item - Visible]
              Submit[push button - Visible]
        [editable text - Visible]
```

Children:	0
Def Action:	Double Click
Description:	
Help:	
Help Topic:	
Keyboard:	
Location:	(784, 455, 1013, 479)
Name:	Avery 1 of 4
Parent:	Choose one or more users: , class: "M
Role Text:	list item
State:	selectable, focusable
Value:	
Id:	100
Window:	0x003d04cc, class: "MozillaV
Object Type:	Simple Element
Test Summary:	

The FlashPlayer’s accessible object model

FlashPlayer exposes by default

- Text
- Input text fields
- Buttons
- Simple MovieClips
- Scripted MovieClips (Flex and Flash UI Components)

flash.accessibility.AccessibilityImplementation

The AccessibilityImplementation class is the base class in Flash Player that allows for the implementation of accessibility in components.

It provides an IAccessible interface for a Flash component

- Methods
  - `get_accName()`
  - `get_accRole()`
  - `getChildIDArray()`
  - `get_accDefaultAction()`
  - `get_accFocus()`
  - `get_accSelection()`
  - `get_accState()`
  - `get_accValue()`
  - `accLocation()`
  - `accDoDefaultAction()`
  - `accSelect()`

Constants in MSAA

Object Roles

```
ROLE_SYSTEM_ALERT = 0x08;
ROLE_SYSTEM_ANIMATION = 0x36;
ROLE_SYSTEM_APPLICATION = 0x0e;
ROLE_SYSTEM_BORDER = 0x13;
ROLE_SYSTEM_BUTTONDROPDOWN = 0x38;
ROLE_SYSTEM_BUTTONDROPDOWNGRID = 0x3a;
ROLE_SYSTEM_BUTTONMENU = 0x39;
ROLE_SYSTEM_CARET = 0x07;
ROLE_SYSTEM_CELL = 0x1d;
ROLE_SYSTEM_CHARACTER = 0x20;
ROLE_SYSTEM_CHART = 0x11;
ROLE_SYSTEM_CHECKBUTTON = 0x2c;
ROLE_SYSTEM_CLIENT = 0x0a;
ROLE_SYSTEM_CLOCK = 0x3d;
ROLE_SYSTEM_COLUMN = 0x1b;
ROLE_SYSTEM_COLUMNHEADER = 0x19;
ROLE_SYSTEM_COMBOBOX = 0x2e;
ROLE_SYSTEM_CURSOR = 0x06;
ROLE_SYSTEM_DIAGRAM = 0x35;
ROLE_SYSTEM_DIAL = 0x31;
ROLE_SYSTEM_DIALOG = 0x12;
ROLE_SYSTEM_DOCUMENT = 0x0f;
ROLE_SYSTEM_DROPLIST = 0x2f;
ROLE_SYSTEM_EQUATION = 0x37;
ROLE_SYSTEM_GRAPHIC = 0x28;
ROLE_SYSTEM_GRIP = 0x04;
ROLE_SYSTEM_GROUPING = 0x14;
ROLE_SYSTEM_HELPBALLOON = 0x1f;
ROLE_SYSTEM_HOTKEYFIELD = 0x32;
ROLE_SYSTEM_INDICATOR = 0x27;
ROLE_SYSTEM_LINK = 0x1e;
ROLE_SYSTEM_LIST = 0x21;
ROLE_SYSTEM_LISTITEM = 0x22;
ROLE_SYSTEM_MENUBAR = 0x02;
ROLE_SYSTEM_MENUITEM = 0x0c;
ROLE_SYSTEM_MENUPOPUP = 0x0b;
ROLE_SYSTEM_OUTLINE = 0x23;
ROLE_SYSTEM_OUTLINEITEM = 0x24;
ROLE_SYSTEM_PAGETAB = 0x25;
ROLE_SYSTEM_PAGETABLIST = 0x3c;
ROLE_SYSTEM_PANE = 0x10;
ROLE_SYSTEM_PROGRESSBAR = 0x30;
ROLE_SYSTEM_PROPERTYPAGE = 0x26;
ROLE_SYSTEM_PUSHBUTTON = 0x2b;
ROLE_SYSTEM_RADIOBUTTON = 0x2d;
ROLE_SYSTEM_ROW = 0x1c;
ROLE_SYSTEM_ROWHEADER = 0x1a;
ROLE_SYSTEM_SCROLLBAR = 0x03;
ROLE_SYSTEM_SEPARATOR = 0x15;
ROLE_SYSTEM_SLIDER = 0x33;
ROLE_SYSTEM_SOUND = 0x05;
ROLE_SYSTEM_SPINBUTTON = 0x34;
 ROLE_SYSTEM_SPLITBUTTON = 0x3e;
ROLE_SYSTEM_STATICTEXT = 0x29;
ROLE_SYSTEM_STATUSBAR = 0x17;
ROLE_SYSTEM_TABLE = 0x18;
ROLE_SYSTEM_TEXT = 0x2a;
ROLE_SYSTEM_TITLEBAR = 0x01;
ROLE_SYSTEM_TOOLBAR = 0x16;
ROLE_SYSTEM_TOOLTIP = 0x0d;
ROLE_SYSTEM_WHITESPACE = 0x3b;
ROLE_SYSTEM_WINDOW = 0x09;
```

Constants in MSAA

Object States

```
STATE_SYSTEM_ALERT_HIGH = 0x10000000;
STATE_SYSTEM_ALERT_LOW = 0x04000000;
STATE_SYSTEM_ALERT_MEDIUM = 0x08000000;
STATE_SYSTEM_ANIMATED = 0x00004000;
STATE_SYSTEM_BUSY = 0x00000800;
STATE_SYSTEM_CHECKED = 0x00000010;
STATE_SYSTEM_COLLAPSED = 0x00000400;
STATE_SYSTEM_DEFAULT = 0x00000100;
STATE_SYSTEM_EXPANDED = 0x00000200;
STATE_SYSTEM_EXTSELECTABLE = 0x02000000;
STATE_SYSTEM_FLOATING = 0x00001000;
STATE_SYSTEM_FOCUSABLE = 0x00100000;
STATE_SYSTEM_FOCUSED = 0x00000004;
STATE_SYSTEM_HOTTRACKED = 0x00000080;
STATE_SYSTEM_INDETERMINATE = 0x00000020;
STATE_SYSTEM_INVISIBLE = 0x00008000;
STATE_SYSTEM_LINKED = 0x00400000;
STATE_SYSTEM_MARQUEED = 0x00002000;
STATE_SYSTEM_MIXED = 0x00000020;
STATE_SYSTEM_MOVEABLE = 0x00040000;
STATE_SYSTEM_MULTISELECTABLE = 0x01000000;
STATE_SYSTEM_NORMAL = 0x00000000;
STATE_SYSTEM_OFFSCREEN = 0x00010000;
STATE_SYSTEM_PRESSED = 0x00000008;
STATE_SYSTEM_PROTECTED = 0x20000000;
STATE_SYSTEM_READONLY = 0x00000040;
STATE_SYSTEM_SELECTABLE = 0x00200000;
STATE_SYSTEM_SELECTED = 0x00000002;
STATE_SYSTEM_SELFVOICING = 0x00080000;
STATE_SYSTEM_SIZEABLE = 0x00020000;
STATE_SYSTEM_TRAVERSED = 0x00800000;
STATE_SYSTEM_UNAVAILABLE = 0x00000001;
```

Constants in MSAA

Selection Flags

```
SELFLAG_TAKEFOCUS = 0x01;
SELFLAG_TAKESELECTION = 0x02;
SELFLAG_EXTENDSELECTION = 0x04;
SELFLAG_ADDSELECTION = 0x08;
SELFLAG_REMOVESELECTION = 0x10;
```

Constants in MSAA

Event Constants

```
EVENT_OBJECT_CREATE = 0x8000;
EVENT_OBJECT_DESTROY = 0x8001;
EVENT_OBJECT_SHOW = 0x8002;
EVENT_OBJECT_HIDE = 0x8003;
EVENT_OBJECT_REORDER = 0x8004;
EVENT_OBJECT_FOCUS = 0x8005;
EVENT_OBJECT_SELECTION = 0x8006;
EVENT_OBJECT_SELECTIONADD = 0x8007;
EVENT_OBJECT_SELECTIONREMOVE = 0x8008;
EVENT_OBJECT_SELECTIONWITHIN = 0x8009;
EVENT_OBJECT_STATECHANGE = 0x800a;
EVENT_OBJECT_LOCATIONCHANGE = 0x800b;
EVENT_OBJECT_NAMECHANGE = 0x800c;
EVENT_OBJECT_DESCRIPTIONCHANGE = 0x800d;
EVENT_OBJECT_VALUECHANGE = 0x800e;
EVENT_OBJECT_PARENTCHANGE = 0x800f;
EVENT_OBJECT_HELPCHANGE = 0x8010;
EVENT_OBJECT_DEFACTIONCHANGE = 0x8011;
EVENT_OBJECT_ACCELERATORCHANGE = 0x8012;
```

mx.accessibility.AccImpl

Abstract class extends AccessibilityImplementation for Flex components

- Properties
  - `.eventsToHandle : Array`
    All subclasses must override this function by returning an array of strings of the events to listen for.
  - `.master : UIComponent`
    A reference to the UIComponent instance that this AccImpl instance is making accessible.
  - `.role : uint`
    Accessibility role constant of the component being made accessible.
    http://livedocs.adobe.com/flex/3/langref/accessibilityImplementationConstants.html#roles

mx.accessibility.AccImpl

Abstract class extends AccessibilityImplementation for Flex component

- Methods
  - **`enableAccessibility()`**
    This method is called by application startup code that is autogenerated by the MXML compiler.

    At runtime, when instances of that type of component are initialized, their **`accessibilityImplementation`** property will be set to an instance of this class.
  - **`eventHandler()`**
    Handles events dispatched from the **`master`** UIComponent.

Creating mx.accessibility.PopUpButtonAccImpl

Start with component class definition **`mx.controls.PopUpMenu`**

- Add **`[AccessibilityClass]`** metadata declaration to let the compiler know where to find the AccessibilityImplementation for the component.

```
[AccessibilityClass(implementation="mx.accessibility.PopUpButtonAccImpl")]
```

- Within the class definition, add the following placeholder for the "mix-in" function

```
mx_internal static var createAccessibilityImplementation:Function;
```

Creating mx.accessibility.PopUpButtonAccImpl

Continue with **`mx.controls.PopUpMenu`**

- Override the **`UIComponent.initializeAccessibility()`** method with the following to initialize the AccessibilityImplementation for a given component instance at runtime.

```
override protected function initializeAccessibility():void {
        if (PopUpButton.createAccessibilityImplementation != null)
            PopUpButton.createAccessibilityImplementation(this);
}
```

Creating mx.accessibility.PopUpButtonAccImpl

Copy over the similar **`mx.accessibility.ButtonAccImpl`** subclass to a new ActionScript class called **`mx.controls.PopUpMenuAccImpl`**

- Refactor `mx.accessibility.ButtonAccImpl` to `mx.controls.PopUpMenuAccImpl` in new class definition.

```
 …

public class PopUpButtonAccImpl extends AccImpl

{

…
```

Creating mx.accessibility.PopUpButtonAccImpl

Class Initialization

```
…

private static var accessibilityHooked:Boolean = hookAccessibility();

…

private static function hookAccessibility():Boolean

{

    PopUpButton.createAccessibilityImplementation =

        createAccessibilityImplementation;

        return true;

}

…
```

Creating mx.accessibility.PopUpButtonAccImpl

Class Methods

```
…
mx_internal static function
   createAccessibilityImplementation(component:UIComponent):void
{
          component.accessibilityImplementation =
          new PopUpButtonAccImpl(component);
}
…
public static function enableAccessibility():void
{
}
…
```

Creating mx.accessibility.PopUpButtonAccImpl

Class Constants

```
…
private static const STATE_SYSTEM_PRESSED:uint = 0x00000008;
private static const STATE_SYSTEM_HOTTRACKED:uint = 0x00000080;
private static const STATE_SYSTEM_HASPOPUP:uint = 0x40000000;
private static const EVENT_OBJECT_NAMECHANGE:uint = 0x800C;
private static const EVENT_OBJECT_STATECHANGE:uint = 0x800A;
private static const ROLE_SYSTEM_SPLITBUTTON:uint = 0x3e;
private static const ROLE_SYSTEM_BUTTONDROPDOWN:uint = 0x38;
…
```

More detail on SplitButton Control:

http://msdn.microsoft.com/en-us/library/bb404170.aspx#ActiveAccessibility2007OfficeFluentUI_TheSplitButtonControl

Creating mx.accessibility.PopUpButtonAccImpl

Constructor

```
…
public function PopUpButtonAccImpl(master:UIComponent)
{
    super(master);

    role = ROLE_SYSTEM_SPLITBUTTON;
}
…
```

We set the role in the constructor to ROLE_SYSTEM_SPLITBUTTON.

Creating mx.accessibility.PopUpButtonAccImpl

Override **`eventsToHandle`** getter method inherited from **`mx.accessibility.AccImpl`**

```
…
override protected function eventHandler(event:Event):void {
    switch (event.type) {
        case "click":
            Accessibility.sendEvent(master, 0, EVENT_OBJECT_STATECHANGE);
            Accessibility.updateProperties();
            break;
        case "labelChanged":
            Accessibility.sendEvent(master, 0, EVENT_OBJECT_NAMECHANGE);
            Accessibility.updateProperties();
        break;
     }
}
…
```

Creating mx.accessibility.PopUpButtonAccImpl

Override **`eventHandler()`** method inherited from **`mx.accessibility.AccImpl`**

```
…
override protected function get eventsToHandle():Array
{
return super.eventsToHandle.concat([ "click", "labelChanged" ]);
}
…
```

Creating mx.accessibility.PopUpButtonAccImpl

Override **get_accRole** method inherited from **flash.accessibility.AccessibilityImplementation**

```
…

override public function get_accRole(childID:uint):uint

{

    if (childID == 0)

        return role;


     return ROLE_SYSTEM_BUTTONDROPDOWN;

}

…
```

Creating mx.accessibility.PopUpButtonAccImpl

Override **get_accValue** method inherited from **flash.accessibility.AccessibilityImplementation**

```
…
override public function get_accValue(childID:uint):String{
    var accValue:String;
    var popUpButton:PopUpButton = PopUpButton(master);
    if(childID == 0){
        var label:String = popUpButton.label;
        if(popUpButton.popUp && popUpButton.popUp is Menu && label != null && label != ""){
            // If the popUp exists and is a Menu and the label exists and is not an empty string...
            var popUpMenu:Menu = popUpButton.popUp as Menu;
            if(popUpMenu.itemToLabel(popUpMenu.selectedItem) == label){
            // If the label matches the popUp menu's selectedIndex, return the label as the accValue.
            accValue = popUpMenu.itemToLabel(popUpMenu.selectedItem);
            if(popUpButton.accessibilityProperties && popUpButton.accessibilityProperties.shortcut)
                // If a keyboard shortcut is defined in the PopUpButton's .accessibilityProperties object,
               //  append it to the returned accValue.
                accValue += " ("+popUpButton.accessibilityProperties.shortcut+")";
        }
    }
    return accValue;
}

…
```

Creating mx.accessibility.PopUpButtonAccImpl

Override **get_accState** method inherited from **flash.accessibility.AccessibilityImplementation**

```
…
override public function get_accState(childID:uint):uint{
    // the normal default state
    var accState:uint = 0;
    var popUpButton:PopUpButton = PopUpButton(master);
    if(childID == 1){ // the drop-down button
        if(popUpButton.popUp) accState = STATE_SYSTEM_HASPOPUP;
        if(popUpButton.mx_internal::isShowingPopUp == true) accState |= STATE_SYSTEM_PRESSED;
    } else {
        accState = getState(childID);
        if (popUpButton.selected) accState |= STATE_SYSTEM_PRESSED;
    }
    var mouseX:Number = master.mouseX, mouseY:Number = master.mouseY, bounds:Rectangle = master.getBounds(master);
    if((mouseX >= bounds.x && mouseX <= (bounds.x + bounds.width)
      && mouseY >= bounds.y && mouseY <= (bounds.y + bounds.height))
      || (popUpButton.focusManager.getFocus() == popUpButton)) accState |= STATE_SYSTEM_HOTTRACKED;

    return accState;
}
…
```

Creating mx.accessibility.PopUpButtonAccImpl

Override **getChildIDArray** method inherited from **flash.accessibility.AccessibilityImplementation**

```
…
override public function getChildIDArray():Array
{
    var childIDs:Array = [];
    for (var i:int = 0; i < 1; i++)
    {
        childIDs[i] = i + 1;
    }
    return childIDs;
}
…
```

Creating mx.accessibility.PopUpButtonAccImpl

Override **`get_accDefaultAction`** method inherited from **`flash.accessibility.AccessibilityImplementation`**

```
…
override public function get_accDefaultAction(childID:uint):String
{
    if(childID == 0){
        return "Press";
    }
    return "Open";
}
…
```

Creating mx.accessibility.PopUpButtonAccImpl

Override **`accDoDefaultAction`** method inherited from **`flash.accessibility.AccessibilityImplementation`**

```
…
override public function accDoDefaultAction(childID:uint):void {
    var popUpButton:mx.controls.PopUpButton = PopUpButton (master);
    if (childID==0) { // force a keyboard click
        var event:KeyboardEvent = new KeyboardEvent(KeyboardEvent.KEY_DOWN);
        event.keyCode = Keyboard.SPACE;
        master.dispatchEvent(event);

        event = new KeyboardEvent(KeyboardEvent.KEY_UP);
        event.keyCode = Keyboard.SPACE;
        master.dispatchEvent(event);
    } else if(childID == 1){ // toggle the popIp button
        if(popUpButton.mx_internal::isShowingPopUp == true){
            popUpButton.close();
        } else {
            popUpButton.open();
        }
    }
}
…
```

Creating mx.accessibility.PopUpButtonAccImpl

Override **`getName`** method inherited from **`mx.accessibility.AccImpl`**

```
…
override protected function getName(childID:uint):String {
    var popUpButton:mx.controls.PopUpButton = master as mx.controls.PopUpButton;
    var popUp:UIComponent = popUpButton.popUp as UIComponent;

    if(popUp){
        if(!popUp.accessibilityProperties) popUp.accessibilityProperties = new AccessibilityProperties();
        if(popUp.accessibilityProperties.name == "") popUp.accessibilityProperties.name = "Context";
    }
    if(childID == 1) return (popUpButton.mx_internal::isShowingPopUp == true) ? "Close" : "Open";

    var label:String = popUpButton.label;
    return label != null && label != "" ? label + " PopUpButton" : "PopUpButton";
}
…
```

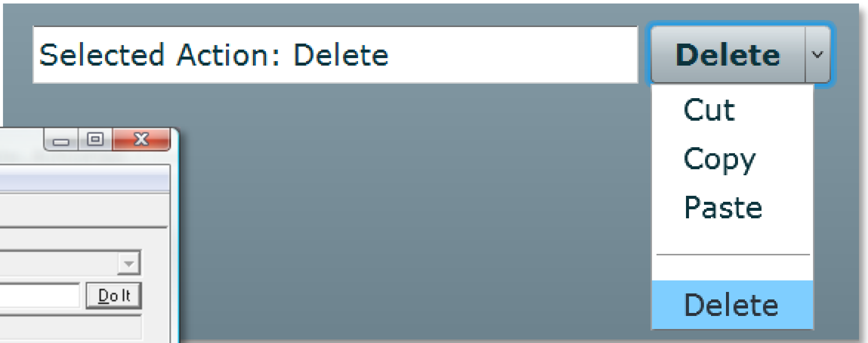
Creating mx.accessibility.PopUpButtonAccImpl

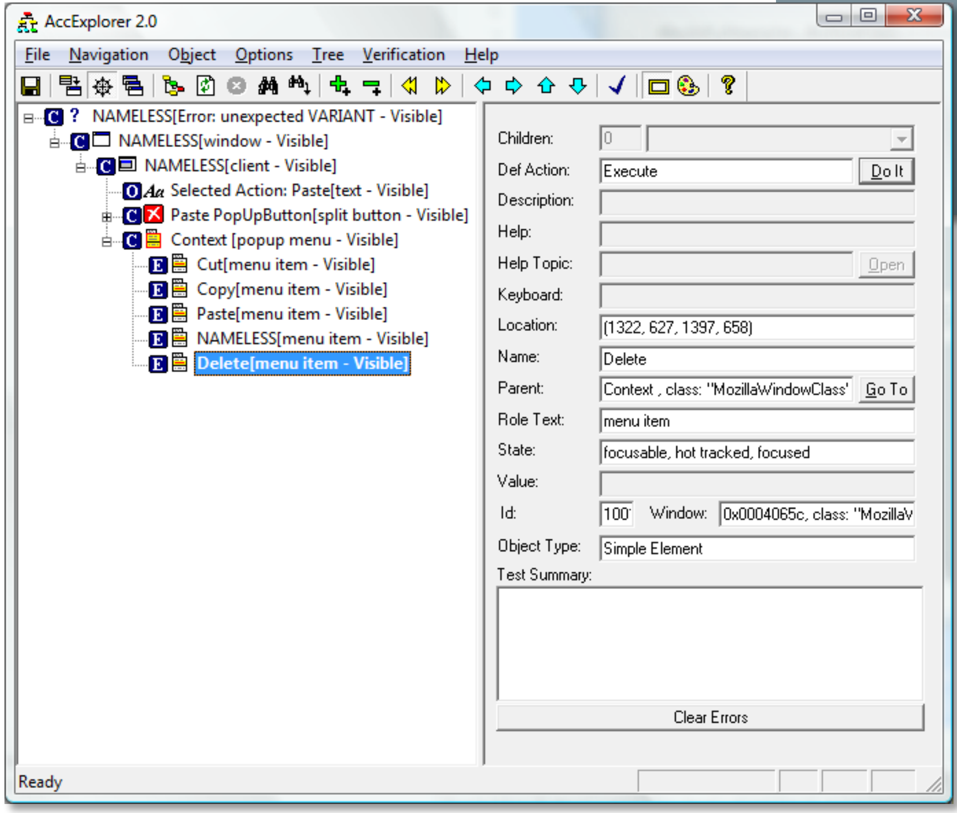
Override **`accLocation`** method inherited from **`flash.accessibility.AccessibilityImplementation`**

```
…
override public function accLocation(childID:uint):* {
    var location:* = master; // the master component
    if(childID == 1){
        // calculate the rectangle location of the child drop-down button
        var popUpButton:PopUpButton = master as PopUpButton;
        var popUpButtonRect:Rectangle = popUpButton.getRect(popUpButton);
        var arrowButtonsWidth:Number = popUpButton.mx_internal::getArrowButtonsWidth();
        location = new Rectangle(
                popUpButtonRect.x + popUpButton.width - arrowButtonsWidth,
                popUpButtonRect.y,
                arrowButtonsWidth,
                popUpButton.height );
    }
    return location;
}
…
```

Creating mx.accessibility.PopUpButtonAccImpl

Test with MSAA SDK tools and screen readers


Selected Action: Delete



```
? NAMELESS[Error: unexpected VARIANT - Visible]
  NAMELESS[window - Visible]
    NAMELESS[client - Visible]
      Selected Action: Paste[text - Visible]
      Paste PopUpButton[split button - Visible]
      Context [popup menu - Visible]
        Cut[menu item - Visible]
        Copy[menu item - Visible]
        Paste[menu item - Visible]
        NAMELESS[menu item - Visible]
        Delete[menu item - Visible]
```

Children:	0
Def Action:	Execute
Description:	
Help:	
Help Topic:	
Keyboard:	
Location:	(1322, 627, 1397, 658)
Name:	Delete
Parent:	Context , class: "MozillaWindowClass"
Role Text:	menu item
State:	focusable, hot tracked, focused
Value:	
Id:	100 Window: 0x0004065c, class: "MozillaV
Object Type:	Simple Element
Test Summary:	

There will be bugs.
But don’t get discouraged.

You’re at MAX.

You can do this!

Questions?

Usefulness

http://www.adobe.com/accessibility/

http://blogs.adobe.com/accessibility/

http://blogs.adobe.com/koestler/

http://livedocs.adobe.com/flex/3/langref/flash/accessibility/package-detail.html

http://majordan.net/adobe/msaa_documentation/html/accessible_7b.html

http://www.microsoft.com/downloads/details.aspx?familyid=3755582A-A707-460A-BF21-1373316E13F0&displaylang=en

http://www.eclipse.org/actf/downloads/tools/aDesigner/index.php

http://majordan.net/AccLinkExample/

CONNECT. DISCOVER. INSPIRE.