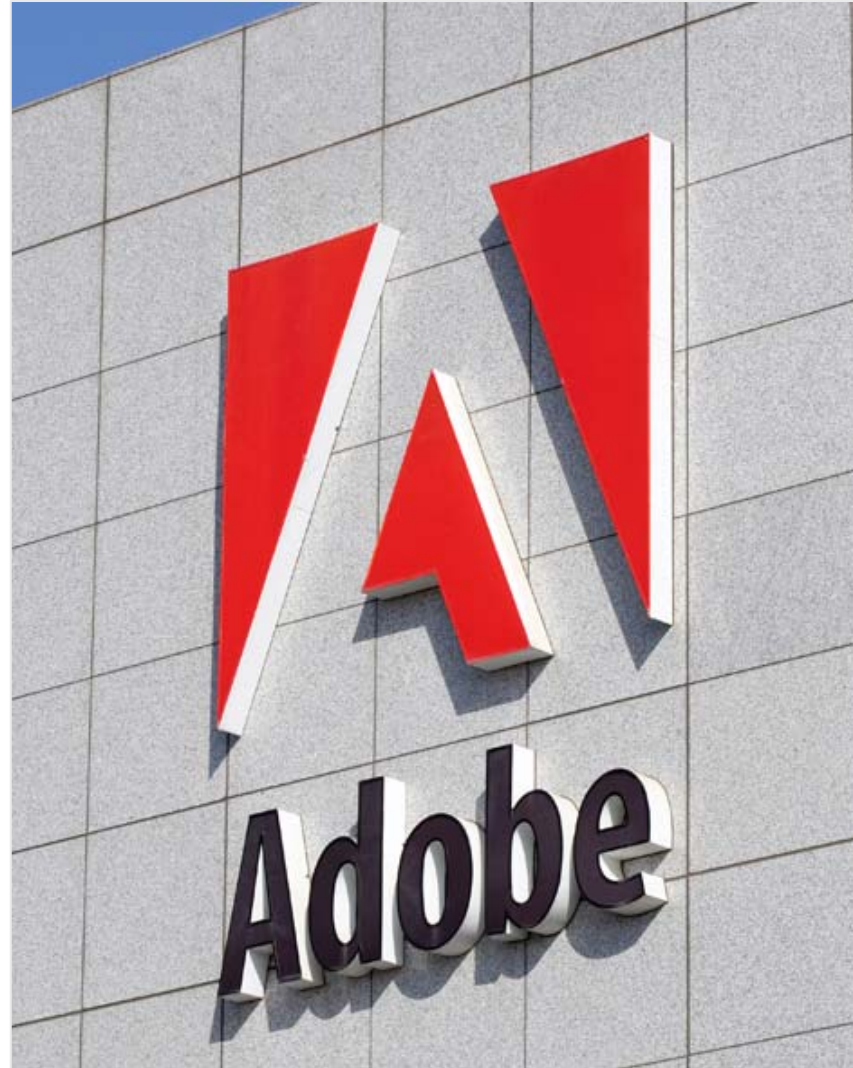


# Accessible Rich Internet Applications: Achieving accessibility with Adobe Flex

Sameer Bhatt

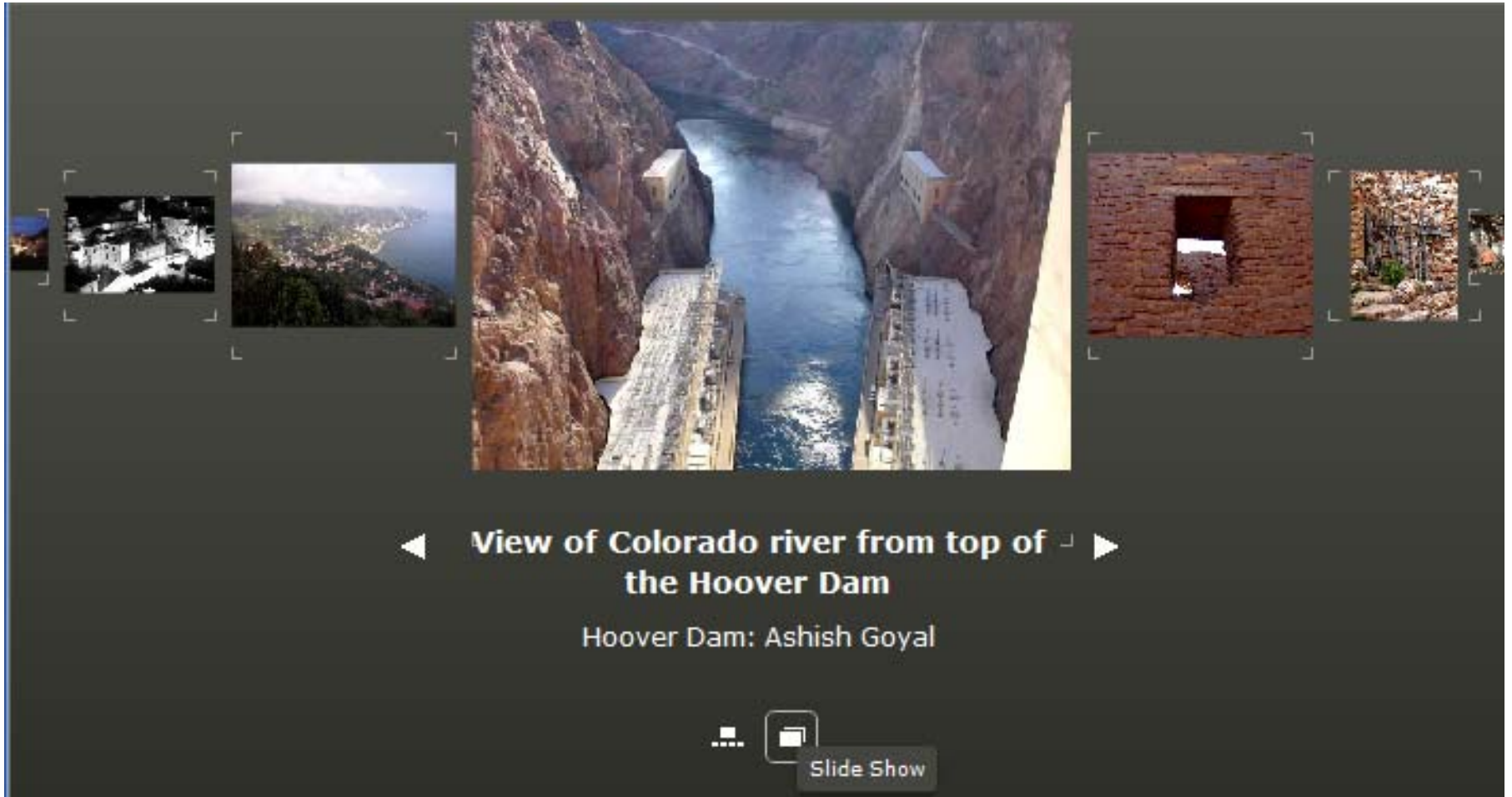
Adobe Systems

[sameer@adobe.com](mailto:sameer@adobe.com)



## What is a Rich Internet Application?

# Accessible RIA



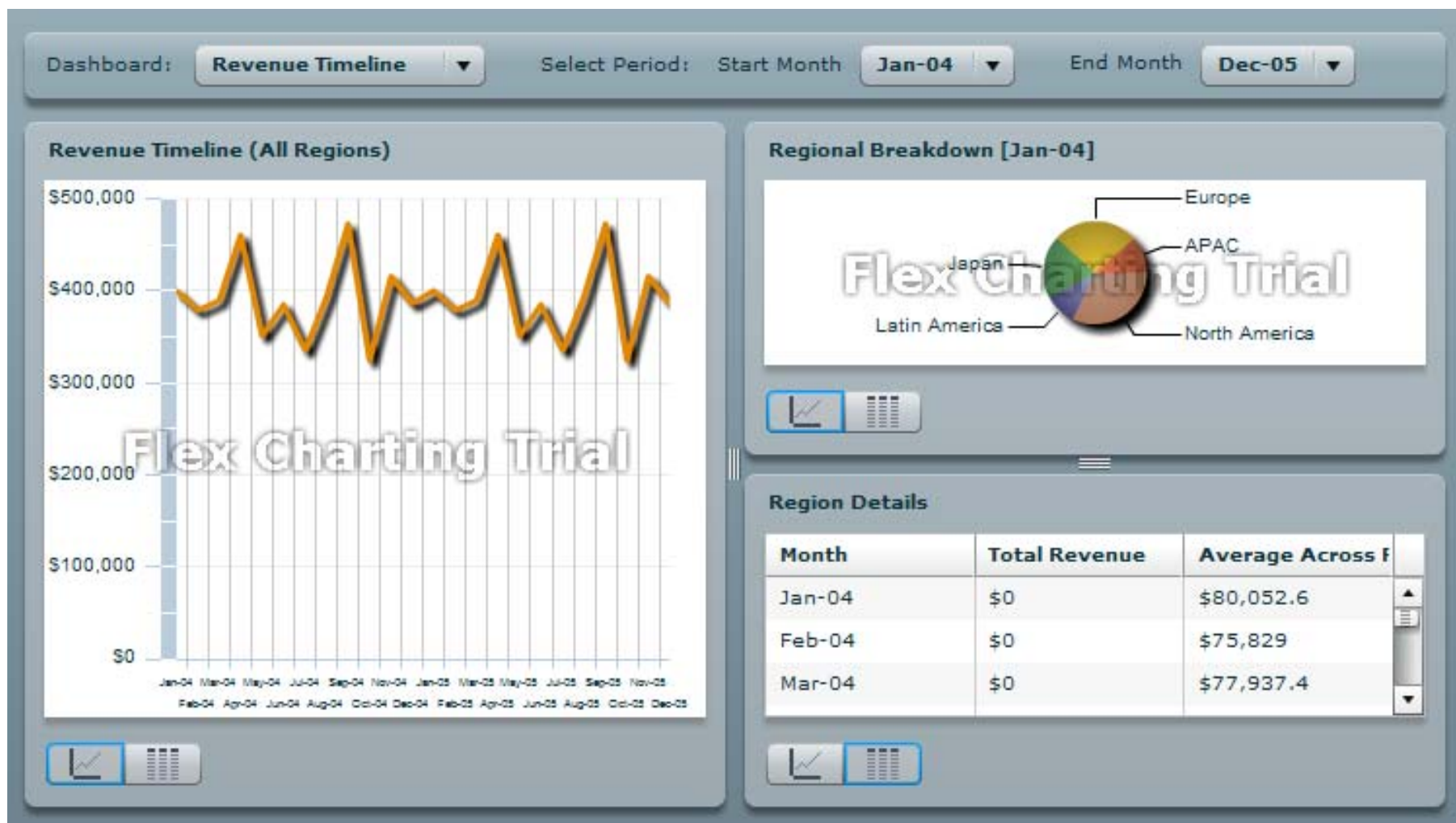
◀ **View of Colorado river from top of the Hoover Dam** ▶

Hoover Dam: Ashish Goyal

Slide Show

The slide show interface features a central large image of the Hoover Dam from an aerial perspective. To the left of the central image are three smaller thumbnail images: a small town, a coastal town, and a landscape with a bay. To the right of the central image are two smaller thumbnail images: a brick wall with a window and a stone wall. Navigation arrows are located on the left and right sides of the central image. Below the central image is a title and author name, and a 'Slide Show' button is at the bottom center.

# Accessible RIA



Wikipedia:  
“Web applications that have the  
features and functionality of  
traditional desktop applications”

## Microsoft Active Accessibility (MSAA)

### Overview

- MSAA is a standard method of sharing accessibility information
- By default many Microsoft controls contain MSAA contexts.
- Flash Player creates an MSAA server within the application that will send MSAA information to a client such as a screen reader or screen magnifier

### Accessible Flex Applications

- “Accessible” components are those with built-in MSAA support
- Even with accessibility enabled, there are accessibility concerns to address.

## Microsoft MSAA Resources

- Active Accessibility 2.0 Documentation
  - [http://msdn.microsoft.com/library/default.asp?url=/library/en-us/msaa/msaastart\\_9w2t.asp](http://msdn.microsoft.com/library/default.asp?url=/library/en-us/msaa/msaastart_9w2t.asp)
- Microsoft Active Accessibility 2.0 SDK Tools (Inspect32, AccExplore32, AccEvent32)
  - <http://www.microsoft.com/downloads/details.aspx?FamilyId=3755582A-A707-460A-BF21-1373316E13F0&displaylang=en>
- Microsoft Active Accessibility: Architecture
  - <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnacc/html/actvaccess.asp>

# Key concerns for novel user interface—

## How to support:

Keyboard Users

Screen Magnifier Users

Screen Reader Users

## Keyboard Interface: What's required?



*Expect* RIA controls to model existing keyboard behavior for basic controls.

Expect developers to provide instructions for controls that don't model other controls.

# Accessible RIA

- Flex 2.0 Keyboard Documentation:

[http://livedocs.adobe.com/flex/2/docs/wwhelp/wwhimpl/common/html/wwhelp.htm?context=LiveDocs\\_Parts&file=00001025.html](http://livedocs.adobe.com/flex/2/docs/wwhelp/wwhimpl/common/html/wwhelp.htm?context=LiveDocs_Parts&file=00001025.html)

- Controls:

<http://msdn.microsoft.com/library/en-us/dnwue/html/ch08c.asp>

- Guidelines for Keyboard User Interface Design:

<http://msdn2.microsoft.com/en-us/library/ms971323.aspx>

# Magnifier Users – What's required?

Keyboard access

Focus

Selection

# Screen Reader Users –

What's required?

Name

Role

State

(And Magnifier Requirements)

# Accessible RIA

**Name**

“What is this object called?”

**Role**

“What kind of object is this?”

**State**

“What are the current settings for this object?”

## Several familiar roles

- Graphic
- Checkbox
- Text
- Radio Button
- Grouping
- List
- Combo Box
- Button
- Text Area

## Less familiar roles

- Page tab
- Spin button
- Dialog
- Slider
- Menu
- More...

# Accessible RIA

Exploiting the Name for other purposes is sometimes needed.

- Assistive technology support for name is solid
- Augmenting the name for an object is easy
- Use standard controls whenever possible

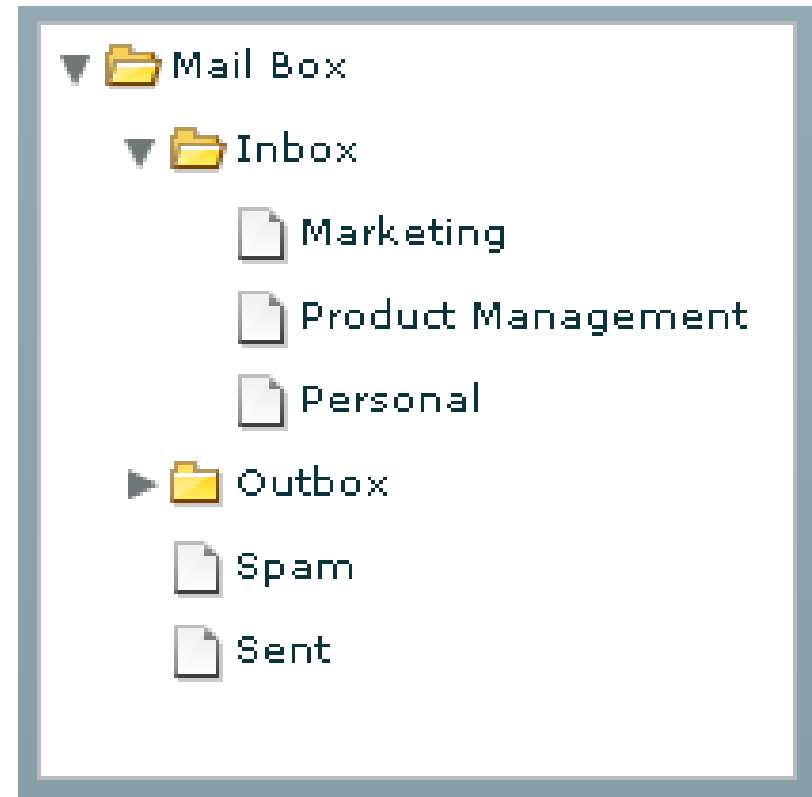
*In the absence of a sufficiently rich API, overloading an object's name to convey role information may be a necessary hack*

## The state of a control is equally important

- Is the control disabled?
- Is a list item selected?
- Is the checkbox checked?
- Is the control open?
- Is this tab the active tab?

Tree controls provide a good example of a control that conveys a lot of state information

- Folder or file selected?
- Folder open?
- Level in tree?
- Tree enabled or disabled?



# Accessible RIA

- WAI-ARIA has a module for state information
  - <http://www.w3.org/TR/aria-state/>
- UIAutomation from Microsoft
  - [http://msdn2.microsoft.com/en-us/library/ms697270\(VS.85\).aspx](http://msdn2.microsoft.com/en-us/library/ms697270(VS.85).aspx)



## 26 Accessible Flex Components

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

# Accessible RIA



## JAWS for Windows

- [http://www.freedomscientific.com/fs\\_products/software\\_jaws.asp](http://www.freedomscientific.com/fs_products/software_jaws.asp)
- JAWS 4.5, 6.1, 6.2, 7.0, 8.0, 9.0 provide solid support for Flash and Flex content
- Flex 2.0 requires JAWS 6.1
- Flex 3.0 requires JAWS 8.0



## Flash Components Scripts

- <http://www.adobe.com/macromedia/accessibility/features/flex/jaws.html>
- These scripts handle issues related to Flash components used in Adobe Flex applications

## Other Assistive Technologies

- Window-Eyes, IBM HomePage Reader, HAL/SuperNova, ZoomText, PC-Talker (Japanese), and JAWS-J also interoperate with Flash and Flex content.

# Best Practices for Development

```
*sample1.mxml X
Source Design
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
  <mx:Panel x="10" y="10" width="1031" height="800" lay
    <mx:Form>
      <mx:FormHeading label="Please register"/>
      <mx:FormItem label="Name">
        <mx:TextInput id="fname" width="200"/>
      </mx:FormItem>
      <mx:FormItem label="Gender">
        <mx:RadioButton label="Male"/>
        <mx:RadioButton label="Female"/>
      </mx:FormItem>
      <mx:FormItem label="E-mail address" width="302">
        <mx:TextInput id="email" width="200"/>
      </mx:FormItem>
      <mx:CheckBox label="Do not share my personal info"
    </mx:Form>
  </mx:Panel>
</mx:Application>
```



**Support Keyboard Access  
and Focus Management**

**Use Accessible  
Components**

**Convey Relationships**

**Use Color Wisely**

**Provide Captions**

**Test for Accessibility**

## Support Keyboard Access and Focus Management

- Common difficulty for assistive technologies
- A big difference between web apps and desktop apps

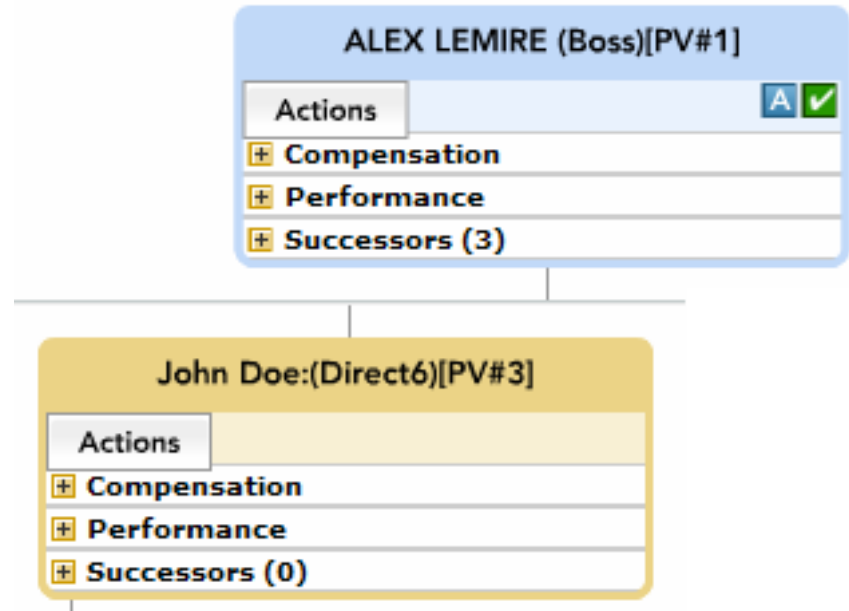


## Define Reading Order

- Today: DOM order or MSAA order
- Define the order by using tabIndex property

## Convey Relationships

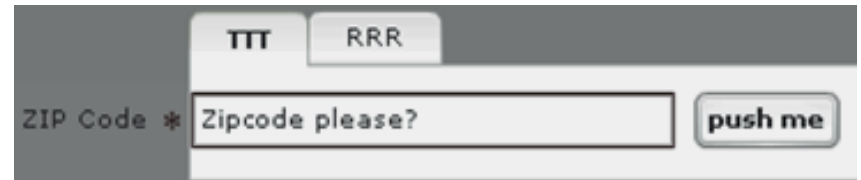
- Communicating what is going on in an application is largely a design issue.
- Assistance can be provided within the application, in the form of a “accessibility information” screen.
- A significant challenge!



## Use Color Wisely

Ensure colors are viewable by individuals who are color blind

- Avoid red – green color combinations
- Use redundant text indicators
- Ensure contrast of foreground and background combinations



## Provide Instructions

- Use a Help or Site Info screen to provide information about the purpose of the application and accessibility specific instructions
- Provide detail about how to use controls created using components
- Non-obvious functionality needs to be revealed for users

# Accessible RIA

- **Customer Success Stories**



Primary barriers for Hearing and Cognitively-impaired users are often content-related

- Deaf and hard of hearing users need captions
- Users with cognitive impairments have many different needs related to complexity and presentation format
- Coding solutions are used to implement content issues – for example, word-completion in a search control.

Ado	
adobe	1,110,000,000 results
adobe reader	486,000,000 results
adobe acrobat	278,000,000 results
adobe photoshop	238,000,000 results
adobe acrobat reader	297,000,000 results
adobe flash player	155,000,000 results
adoption	294,000,000 results
adobe flash	374,000,000 results
adorama	7,830,000 results
adobe illustrator	69,700,000 results

## Testing Strategies

- Test with the keyboard
- Test with MSAA Inspector tools
  - Microsoft Active Accessibility
- Test with assistive technologies
- Test with users

# Resources

- <http://www.adobe.com/accessibility/> (public site)
- <http://blogs.adobe.com/accessibility/> (external blog)

## Q&A

**Better by Adobe.™**