Microsoft Dynamics - Accessibility in forms, products, and controls

- 11/08/2017

This topic describes best practices for enabling accessibility in your form, product, or control. An accessibility checklist is also included.

Accessibility is about inclusion, that is, a person with a disability can perform the same task as a person without that disability. Making an accessible control or form should be as fundamental as making it secure, high-performing, or easy-to-understand.

Keyboard

The bedrock of accessibility is keyboard-only access. When you can use the keyboard to perform all the actions of a form, then it can be utilized by a non-sighted person or a person with restricted or limited use of their hands. This means that all controls can be reached via the tab sequence, direct action (such as Ctrl+S for Save), or through some other shortcut key that enables the user to move to a control, such as Navigation Pane, App Bar, Message Center, or Message Bar. A simple test is to simply disconnect your mouse and complete all core and secondary scenarios using only the keyboard.

Color

The use of color is encouraged and is a common way to express state or status of a record or other piece of information. However, color cannot be the only way that state or status is communicated. An accompanying symbol, help text, or additional column should include a textual description of the state or status. A simple test is to identify all use of color in your system and ensure that color isn't being used to express a state or status. A common example is to use the color red to indicate "needs attention," or the color green to mean an "OK" status.

Images

When showing an image there should be a label that describes the image. If the image expresses state or status of a record, then accompanying help text or an additional column should include a textual description of the state or status. If the image is symbolic, like a logo, then it doesn't require a textual description. If you have an image on a form or grid to convey a status, such as "in progress", ensure that the image has a tooltip that can then be read to someone who is utilizing a screen reader.

```
public display container statusImageDataMethod()
{
ImageReference statusImage;
if (this.Status == NoYes::Yes)
{
```

```
statusImage = ImageReference::constructForSymbol(ImageReferenceSymbol::Accept,
"Accept");
// a product ready example would use a label in place of an embedded string
 }
 else
  {
 statusImage = ImageReference::constructForSymbol(ImageReferenceSymbol::Cancel,
"Cancel");
// a product ready example would use a label in place of an embedded string
 }
 return statusImage.pack();
}
```

Extensible controls

Extensible controls are simply an extension of the controls that are provided by the framework, but require the same usability and accessibility. In addition, widgets or other visually rich controls, such as segmented entry controls or charts, should provide an Accessible Rich Internet Applications (ARIA) tag that is hidden from the sighted user, but provides an introduction to the blind user through a screen reader that describes the use of the control. Every action in an extensible control must be possible with a keyboard.

Layout and dynamic management of forms

A visually impaired or blind user cannot be surprised by significant or unexpected re-laying out of a form based on an action such as selecting a value or entering input.

Accessibility checklist

Measure	**Description**	**Success criteria**	**Validated (x)**
Keyboard-only access to all functions	Drill downs, lookups, hyperlinks, data input actions, and shortcut keys.	All action may be completed without the use of a mouse.	
Visible focus indicator		It’s always apparent which control has focus.	
Focus order	Cannot “jump to” a non-logical location.	Tabbing doesn’t jump to a non-logical portion or unexpected portion of the form.	
Focus trapping	Cannot be “trapped” in a control, focus must be able to	Tabbing into a control should allow tabbing out	

	move out of control using the keyboard.	or a keyboard equivalent that lets them escape.	
Images of text	Cannot use an image to display text.	For example, you cannot have a button label that is an image of text. Logos are exempt.	
Using color	Color alone cannot be used to convey status or state.	All status indicators on a form or grid must have unique shape or texture for each color. See the "Color" section in this document for more information.	
Text contrast	Minimum 4.5:1 ratio.	Extensible controls should use framework color theming to ensure compliance.	
User input instructions	Widgets or other multi-step controls must have a usability overview label or help text (can be embedded in ARIA tag).	When using a screen reader, the control must be introduced and its label or purpose described.	
Change of context	Changing the setting of, or entering data into any UI component must not automatically cause an unexpected change of context that might disorient the user without first notifying the user that the change of context will occur.	For example, selecting a value in a drop-down box or clicking a check box shouldn’t change the layout of the form unexpectedly or in a non-standard way.	
Consistent UI	Widgets or other elements must work consistently across the product. There cannot be two similar looking or modeled elements that behave differently.	For example, selecting a check box shouldn’t open a window.	
Fine motor control	Cannot require use of the mouse. Must support “sticky keys.”	User cannot be forced to click a moving target (pull right) or any other action that requires a mouse. The	

		same action must also be possible with the keyboard.	
Use of video	Video (with audio) must have accompanied text.	When showing pre-recorded content, the deaf person must have supporting text to consume the content.	
Use of images		All images must be accompanied by text that describes the specific content. For example, a “tooltip” or secondary column in a grid that describes and matches the state or status.	
Communication	Use of VoIP or other telecommunication device or software, such as Skype, must be accessible and usable from the keyboard.	Any use of VoIP or telecommunication is a legal obligation.	
Navigation		Navigation controls must communicate that navigation will occur if executed.	

Extensible controls – Insights for accessibility

An extensible control is simply an extension of the framework. The interaction of the extensible control should be considered no different than any other framework control. When the control is a visually-rich widget that offers mouse actions, the author needs to ensure that equivalent, keyboard access functionality is available. A widget may not run in an “accessible mode” that is different than the standard presentation. Instead the widget must offer similar functionality without the need to activate or toggle state of the control. All uses of color should be based on themed colors so that when the mode is “High Contrast” your color scheme matches the theme change. The World Wide Web Consortium (WC3) website provides guidance for "Supported States and Properties" (www.w3.org/TR/wai aria/states_and_properties#state_prop_def). This site is a helpful resource for ARIA tags and provides the definitions that you see below.

Control should

Introduce itself

Importantly, your control should not only identify itself by name, but (using a label or ARIA tag) give a brief introduction on how it works. Aria-describedby. Make sure your descriptive text is supplied via a framework label for localization.

aria-describedby - Identifies the element (or elements) that describes the object.

Example

```
<http://www.w3.org/TR/WCAG20-TECHS/ARIA1.html>
<button aria-label="Close" aria-describedby="descriptionClose"
onclick="myDialog.close()"></button>
<div id="descriptionClose">Closing this window will discard any information
entered and return you to the main page</div>
```

Indicate when it is busy It may not always be clear to the visually-impaired user why the control isn't responsive. Providing a "busy" message helps in these cases.

aria-busy (state) - Indicates whether an element, and its subtree, are currently being updated.

Example

```
<p aria-live="polite" aria-busy="true"></p>
```

Indicate that the contents have been validated and are invalid The async nature will result in a dynamic field state change. The message bar will introduce itself to the visually-impaired user, and the control itself should express an invalid state.

aria-invalid (state) - Indicates the entered value does not conform to the format expected by the application.

Indicate when a field is readonly

aria-readonly - Indicates that the element is not editable, but is otherwise operable. See related aria-disabled.

Indicate that the field requires input (mandatory) The sighted user understands that a field is mandatory through a visual symbol. The non-sighted user will need an identifying tag.

aria-required indicates that user input is required on the element before a form may be submitted.

Describe the state of a toggled value A toggle control has a toggled state. This tag will express that state.

aria-pressed (state) - Indicates the current "pressed" state of toggle buttons. See related aria-checked and aria-selected.

aria-valuenow - Defines the current value for a range widget. See related aria-valuetext.

aria-valuetext - Defines the human readable text alternative of aria-valuenow for a range widget.

Control could

Indicate an expanded state Complex interactions can be learned, but current state isn't always easy to determine without experimentation. When using an aria-expanded tag, the control describes its current state. An example is tabbing to a tab or FastTab section of a control.

aria-expanded (state) - Indicates whether the element, or another grouping element it controls, is currently expanded or collapsed.

Describe applicable context menu Microsoft Dynamics 365 for Finance and Operations provides a context menu. When the application author has provided functionality to the current control or context, you can announce that functionality.

aria-haspopup - Indicates that the element has a pop-up context menu or sub-level menu.

Other Misc

aria-live indicates that an element will be updated, and describes the types of updates the user agents, assistive technologies, and user can expect from the live region.

aria-multiline - Indicates whether a text box accepts multiple lines of input or only a single line.

aria-multiselectable - Indicates that the user may select more than one item from the current selectable descendants.

aria-orientation - Indicates whether the element and orientation is horizontal or vertical.

aria-setsize - Defines the number of items in the current set of listitems or treeitems. Not required if all elements in the set are present in the DOM. See related aria-posinset.

aria-sort- Indicates if items in a table or grid are sorted in ascending or descending order.

aria-valuemax - Defines the maximum allowed value for a range widget.

aria-valuemin - Defines the minimum allowed value for a range widget.