How to Develop Accessible Linux Applications

Sharon Snider

v1.0, 2002−01−28

Revision History

Revision v1.0	2002−01−28	Revised by: sds
Wrote and converted to DocBook SGML.		

This document provides Linux software developers with guidelines and test cases for developing accessible Linux applications.

Table of Contents

1. Introduction

This document provides developers with the information necessary to assess their applications for accessibility. Some of these tests should be performed using various types of adaptive technologies. As Karl Gahlke points out in his article on "Requirements For Adaptive Modules in Linux", "We have a legal obligation (The American Disabilities Act (ADA) in the United States) and a moral obligation to make Linux accessible to as many individuals as possible."

Please send any comments, or contributions via e−mail to Sharon Snider. This document will be updated regularly with new contributions and suggestions.

2. Developing Accessible Applications

Some of the most important reasons for developing accessible software are:

- Software should be accessible to as many users as possible.
- Accessibility to new products benefits everyone. Information technology has provided many benefits to society. However, individuals with disabilities can not participate fully when the technology does not meet the needs of users with disabilities.
- Compliance with worldwide regulations and standards such as Section 508 of the Rehabilitation Act, Americans with Disabilities Act and the World Wide Web Consortium's Web Accessibility Initiative.

2.1. Principles for Developing Accessible Applications

Developers need to consider the following needs of disabled users when developing an accessible application:

- Choice of input methods. Support should be available for various types of input, such as, keyboard, mouse and adaptive technologies. Pay close attention to keyboard navigation.
- Choice of output methods. Support should be available for various types of output, such as, visual display, audio, and print. The main focus is that text labels are provided for all user interface elements and objects, graphics, and icons.
- Consistency and flexibility with the user's system configuration. In addition, include customization options so the user can select color, font, and layout of the work area.

---

3. Guidelines for Developing Accessible Applications

The following sections contain guidelines and tests that developers can use to create more accessible applications. Use Pass, Fail, or Pending as a rating system for each item.

3.1. Keyboard Navigation

3.1.1. Guidelines

The following keyboard navigation settings and sequences can cause accessibility problems. You should confirm that:

- Efficient keyboard access is provided to application features.
- A logical keyboard navigation order has been implemented.
- The correct tab order is used for controls that are dependent on check boxes, radio buttons, or toggle state.
- Keyboard access does not override existing accessibility features.
- The application provides more than one method to perform keyboard tasks whenever possible.
- There are alternate key combinations whenever possible.
- There are no awkward reaches for frequently performed keyboard operations.
- The application does not use repetitive, simultaneous key presses.
- The application provides keyboards equivalents for all mouse functions.
- The application does not use any general navigation functions to trigger operations.
- All the keyboard invoked menus, windows, and tool tips appear near the object they relate to.

3.1.2. Tests

Run the following keyboard tests without using the mouse for all actions. Using only the keyboard commands, move the focus through all menus in the application. You should verify that:

- Context sensitive menus display correctly.
- Any functions listed on the tool bar can be performed using the keyboard.
- You can operate every control in the client area of the application and dialog boxes.
- Text and objects within the application can be selected.
- Any keyboard enhancements or shortcuts are working as designed.

3.2. Mouse Interaction

3.2.1. Guidelines

The following are mouse button actions and sequences that cause accessibility problems. You should confirm that:

- There is no input dependent on mouse button two or three.
- All mouse operations can be canceled.
- There is visual feedback throughout a drag and drop operation.

3.3. Graphical Elements and Objects

3.3.1. Guidelines

The following are graphical element attributes, object attributes, and naming conventions that are needed for accessibility. You should confirm that:

- There are no hard−coded graphical attributes, such as, lines, borders, or shadow thickness.
- There are descriptive names for all application program interface (API) objects.
- All multi−color graphical elements can be adjusted to monochrome only, whenever possible.
- All interactive graphical user interface (GUI) elements are easily identifiable.
- An option to hide non−essential graphics has been provided.

3.3.2. Tests

Test the application using a screen reader and verify that:

- Labels and text are being read correctly, including menus and tool bars.
- Object information is read correctly.

3.4. Fonts and Text

3.4.1. Guidelines

The following are font and text styles, attributes, and labels that cause accessibility problems. You should confirm that:

- All the font styles and sizes are not hard−coded.
- An option to turn off graphical backdrops has been provided.
- All label objects have names that make sense when taken out of context.
- There are no label names that have been used more than once in the same window.
- There is consistency with label positioning throughout the application.
- When using static text as a label for a control, the label immediately precedes the control in tab order.
- An alternative to what you see is what you get (WYSIWYG) is provided.

3.4.2. Tests

Run the following tests to confirm that font size and settings are maintained.

* Change the font in the application and confirm that the changes apply only to the application and not the desktop environment.
* Change colors within the application and confirm that the changes apply only to the application and not the desktop environment.
* Run a screen magnification program and test the font, color, and size of text when being viewed through a magnifier.

3.5. Color and High Contrast Settings

3.5.1. Guidelines

The following are color and high contrast guidelines for the application environment. You should confirm that:

* The application color is not hard−coded and can be changed.
* Color is used as an enhancement and not the only way to convey information.
* The application supports various high contrast settings (For example, black on white, or white on black).
* The application is not dependent on a particular high contrast setting.

3.5.2. Tests

Run the following tests and verify that:

* All information is available by printing a screen shot to a black and white printer.
* All information is conveyed correctly when settings are set to only black and white or high contrast.
* At least three high contrast schemes are available, and they function correctly.
* High contrast settings in the desktop environment are respected by the application (For example, the window bar and font colors that are set by the desktop environment do not change).

3.6. Magnification

3.6.1. Guidelines

The following magnification functions should be built into the application. You should confirm that:

* The application provides the ability to magnify the work area.
* The application has the option to scale the work area.
* The applications not adversely effected by changing the magnification settings.

3.7. Audio

3.7.1. Guidelines

The following are guidelines for audio output. Using a screen reader, confirm that:

- The user can hear all required audio output.
- Audio is not the only means that the information is conveyed.
- The user can configure frequency and volume on all audio alerts and sounds.

3.7.2. Tests

The application should have an option to show audio alerts and sounds visually. Test that the audio is working correctly with sound enabled. Verify that:

- The application is working as designed when the user performs an action that generates an audio alert.
- The application works correctly when increasing or decreasing the volume.
- Warning messages and alerts can be heard correctly in a noisy work environment.

3.8. Animation

3.8.1. Guidelines

The following are guidelines for all animation that is included in the application. You should confirm that:

- There are no blinking elements with a frequency greater then 2 Hertz (Hz) and lower then 55Hz.
- There are no large areas that flash or blink.

3.8.2. Tests

Run the following tests on applications that include animation. You should verify that:

- An option is available to stop or turn off animation.
- When the animation is turned off it is working correctly.

3.9. Focus

3.9.1. Guidelines

Focus is determined by the location of the cursor as the user moves through the application or display panels. The following are guidelines for focus within the application. You should confirm that:

- Focus starts at the most commonly used controls.
- The current input focus is clearly displayed at all times.
- The input focus is in the active display panel.

- The appropriate feedback is provided when the user attempts to navigate past the end of a group of related objects.
- The default audio alert is played when the user presses an inappropriate key.

3.10. Visual Focus Indicator

3.10.1. Guidelines

The visual focus indicator tells the user the position of the cursor and provides enough information, so the user understands what to do next. The following are guidelines for the visual focus indicator. You should confirm that:

- There is sufficient audio information for the visual focus indicator, so the user can figure out what to do next.
- Screen readers and Braille devices can confirm the current cursor position within the application and read the content of the visual focus indicator.

3.10.2. Tests

Test the following using a screen reader or Braille device. You should verify that:

- When moving among objects the visual focus indicator is easy to identify.
- Keyboard navigation through the application menus is clearly visible when the focus moves.
- The screen reader or Braille device is tracking the visual focus indicator as you navigate using a keyboard.
- When running a screen magnification program that the magnifier can track the visual focus indicator accurately as you navigate using the keyboard and mouse.

3.11. Timing

3.11.1. Guidelines

The following guidelines apply to timing options built−in to the application. You should confirm that:

- There are no hard−coded timeouts or other time−based features.
- There are no objects that display briefly and then hide information based on the movement of the mouse pointer.

3.11.2. Tests

Test the following for timing related to your application. You should verify that:

- The user is notified before a message times out and is given the option to indicate that more time is needed.

- An option is available to adjust the response time and confirm that it is working as designed.

3.12. Documentation

3.12.1. Guidelines

The following are guidelines for writing accessible documentation:

- All documentation is in an accessible format (For example, HTML, or text).
- Documentation is available on all accessibility features of the application.
- State if the application does not support the standard keyboard access that is used by the operating system.
- Identify if there are unique keyboard commands.
- Identify and explain all accessibility features.
- When documenting mouse actions, include the alternative keyboard action as well.

3.12.2. Tests

Run the following test to verify that the documentation is available and accessible.

- Open a help file while in the application using a screen reader or Braille device and confirm the information is accessible, clear, and precise.

4. Additional Resources:

The following Web sites provide checklists and testing information that is more specific to the various Linux development environments:

- American Foundation for the Blind provides information on creating accessible applications at http://www.afb.org/info−document_view.asp?documentid=198.
- GNOME Accessibility Project has written a guide specifically for application development in the GNOME 2.0 desktop. It includes information using their Accessibility Tool Kit (ATK). Additional information is available athttp://developer.gnome.org/projects/gap/guide/gad/index.html.
- IBM Accessibility Center provides links to a Java, Web, and Software accessibility checklist for application development. This site is located at http://www−3.ibm.com/able/guidelines.
- Sun Accessibility provides accessibility information on designing accessible Java applications. More information is available at www.sun.com/access/developers/software.guide.html.
- The Web Accessibility Initiative Web site includes guidelines, checklists, and techniques for developing accessible Web sites and applications. Additional information is located at http://www.w3.org/WAI/.