includes an event listener that moves the crop area when specific keys are pressed. The Control+Shift+arrow keys control the crop rectangle's location.

```
imageCropArea.addEventListener(KeyboardEvent.KEY _ DOWN, adjustCrop);
function adjustCrop(e:KeyboardEvent): void
{
  // check to ensure shift and control are pressed
  if (!e.ctrlKey || !e.shiftKey)
    return;
  switch (e.keyCode)
  {
    case 37: // left arrow
    {
       // move crop area left
    }
    case 38: // up arrow
    {
      // move crop area up
    }
    case 39: // right arrow
    {
       // move crop area right
    }
    case 40: // down arrow
    {
      // move crop area down
    }
}
```

Keyboard access to scrolling content and fields

In a Flex application, if a component in a scrollable area receives focus when it is not within the scrollable view, it is not automatically scrolled into view when the user tabs to it. It is best, therefore, not to use scrolling areas that contain actionable elements. If active elements must be used within scrolling areas, see "Creating Tab Order and Reading Order" (*http://goo.gl/nwLJv*) in the Flex 4 documentation for an example that shows how to scroll to a component when tabbing. Screen readers may not detect components that were previously out of view fast enough when they are brought into view. See "Sending events" on page 43 for details on how to use the `sendEvent()` method of the Accessibility class to notify the screen reader software that a new component has received focus.

When lengthy non-actionable text (such as license text) is placed in a field, the field must be scrollable via the keyboard so keyboard-only users can review it. By default, some components provide keyboard access via the up arrow, down arrow, page up, and page down keys. Developers should verify this functionality for the component, and, if it is not provided by default, implement another keyboard-accessible scrolling mechanism.

Elements that should not be keyboard accessible

Some elements should not be keyboard accessible. Including the following elements in the tab order only slows down access to content by users who rely solely on the keyboard for navigation:

- Form labels. Users need to be able to tab to form elements, not to their labels.
- Elements not visible on the screen, including elements that are offstage or hidden behind other elements. Note that this does not include actionable controls that are out of sight in a scrollable area.
- Elements that are inactive or disabled, unless it is important to convey that the button exists. In that case, the state of the button should be conveyed if the focus is set on that disabled object.
- Decorative objects that convey no meaningful information.