```
            <mx:FormItem>
                <s:Button id="mySubmitBtn" label="Login"
                    click="submitLogin();"/>
            </mx:FormItem>
        </mx:Form>
    </s:Panel>
```

Adding event listeners to monitor for Enter and Space key press events will not work correctly for all screen readers and should be avoided when possible. Screen readers such as JAWS for Windows, Window-Eyes, and NVDA trap keys such as Enter and Space and will not send them through to the Flex application in commonly used screen-reading modes.

When not to provide keyboard access

In general, all operable controls in a Flex application must be keyboard accessible unless one or more of these is true:

- The control is disabled
- The functionality is duplicated by a second, keyboard-accessible component
- The functionality cannot be implemented without a mouse, drawing pad, or other pointing device. (This may include, for example, freehand or pressure sensitive drawing. It does not, however, include drag-and-drop operations, which can often be accomplished through keyboard access and are discussed below.)

Keyboard access can be removed from an item that is typically in the tab order by setting the `tabEnabled` property to `false`. This is done automatically when the `enabled` or `visible` property of the component is set to `false`. The `tabEnabled` property should be used when the component should not receive keyboard focus but must remain enabled and visible.

```
// ActionScript
btnDelete.tabEnabled = false;

// MXML
<s:Button id="btnDelete" tabEnabled="false" />
```

While `tabEnabled` is the preferred way to disable keyboard focus, developers may also use `focusEnabled` for the same effect.

```
//ActionScript
btnDelete.focusEnabled = false;

// MXML
<s:Button id="btnDelete" focusEnabled="false" />
```

When components are grouped together and the whole set of components should not be tab-enabled, the `tabChildren` property can be set to `false`. This property is useful when a title window or simulated modal dialog appears and components in the background need to be removed from the tab order.

```
//ActionScript
vgSignInForm.tabChildren = false;

// MXML
<s:Button id="vgSignInForm" tabChildren="false" />
```

Keyboard accessibility considerations

When ensuring the keyboard accessibility of an application, the following cases merit special attention:

Drag-and-drop functionality

Many users who have difficulty using a mouse will be unable to use applications that depend upon drag-and-drop functionality. Hence, developers must provide a keyboard-accessible way to perform drag-and-drop