```
}
]]>
</fx:Script>
...
<mx:Accordion id="a1" tabIndex="50" width="100%">
  <mx:Form tabIndex="51" label="Accordion Pane 1" width="100%">
    <mx:FormItem id="f15" label="Text Input 1:">
<s:TextInput tabIndex="54"/>
    </mx:FormItem>
  </mx:Form>
  <mx:Form tabIndex="52" label="Accordion Pane 2" width="100%">
    <mx:FormItem id="f16" label="Text Input 2:">
<s:TextInput tabIndex="55"/>
    </mx:FormItem>
  </mx:Form>
  <mx:Form tabIndex="53" label="Accordion Pane 3" width="100%">
    <mx:FormItem id="f17" label="Text Input 3:">
<s:TextInput tabIndex="56"/>
    </mx:FormItem>
  </mx:Form>
</mx:Accordion>
```

For TabNavigator components, use the `getTabAt()` method instead of `getHeaderAt()` to obtain the child component tabs. To correctly set the `tabIndex` on the title text of an MX Panel component, the component must be extended because the titleBar child component is protected and cannot be directly accessed from outside of the MX Panel component.

The MX Text and Label controls are also compound components; they both contain a textField. Unfortunately, the reading order cannot be set correctly for MX Text and Label controls when used with the Flex 4 SDK. This issue does not occur in the Flex 3 SDK, nor does it occur when using the Spark equivalent components with the Flex 4 SDK. Adobe recommends using Spark components instead of MX components whenever an equivalent exists. The reading order of Spark RichText and Label components is controlled by simply setting the `tabIndex` property.

## Skipping repetitive components

Repetitive components that appear on every screen in a Flex application can make the application difficult to use by individuals with disabilities. Users without disabilities quickly learn to ignore such repetitive components—a global navigation feature, for example—until they need them. In contrast, users of screen reader software must listen to and navigate past them at the beginning of each new screen, and keyboard-only users must tab through them on each screen to reach the new page content. It is a good practice to provide a method for users to avoid these items until needed. There are two commonly used methods for achieving this without changing the visual appearance of the Flex application:

- Change the reading order
- Provide a link, button, or shortcut key to skip past the repetitive content

### Changing the reading order

When repetitive navigation links appear above or to the left of content, change the reading order so that screen readers read the content before the navigation links. Screen reader users will encounter the main page content before the repetitive links and buttons. Keyboard-only users can access the content without tabbing through the links or buttons on each screen. If there are many other keyboard-accessible elements on the screen,