To avoid this type of forced focus change, developers should keep the focus on the radio button or combo box until the user explicitly opts to move on, for example by pressing the Tab key. A Submit or Next button may be used to invoke the action based on the selected radio button; for example:

```
// 4 Radio buttons are present
<s:RadioButtonGroup id="group1" />
<s:RadioButton label="Small" id="rbtnPizzaSizeSmall" tabIndex= "2"
     groupName="group1" />
<s:RadioButton label="Medium" id=”rbtnPizzaSizeMedium" tabIndex = "3"
     groupName="group1" />
<s:RadioButton label="Large"  id="rbtnPizzaSizeLarge" tabIndex = "4"
     groupName="group1" />
<s:RadioButton label="Extra Large” id=”rbtnPizzaSizeExtraLarge”
     tabIndex = "5" groupName="group1" />
<s:Button label="Submit" id="btnSubmit" tabIndex ="6" />
...
//ActionScript
btnSubmit.addEventListener(MouseEvent.CLICK, advancedToNextStep );
function advanceToNextStep(e: MouseEvent): void
{
  // advance to next step not shown
}
```

Visual focus indication

By default, Flex provides a blue rectangle to indicate the current focus for standard components. In addition, Flash Player provides a yellow rectangle for any native Flash objects interspersed with Flex components. Some developers may change the focus rectangle skin to hide the visual focus, perhaps because the color does not match the application theme. When the visual focus is not visible, however, users will have no idea where the next keyboard action will take place. This makes the application exceptionally difficult to use. With few exceptions, this technique will also render an application inaccessible. All accessible applications must provide a visual indication of focus. While Adobe recommends using the default Flex and Flash focus indications, developers are welcome to provide their own form of visual focus indication. Providing no visual focus within an application, however, is not an option for accessible applications. Similarly, providing a visual focus that is not easy to see may cause difficulties for users with visual impairments.

To modify the visual focus indication for a component, set its `focusSkin` property to the class name of the custom focus skin. The custom class can be defined via MXML or ActionScript. The class must implement `get` and `set` methods for the target property and must have an `updateDisplayList()` method. A combination of the Spark Rect primitive and its `fill` property can be used; for example:

```
// set the focusSkin property to MyFocusSkin class
<s:TextInput width="150" accessibilityName="First Name:" focusSkin="MyFocusSkin" />

// mxml file MyFocusSkin.mxml containing the custom focus skin class
<?xml version="1.0" encoding="utf-8"?>
<s:Group xmlns:fx="http://ns.adobe.com/mxml/2009"
                xmlns:s="library://ns.adobe.com/flex/spark"
                xmlns:mx="library://ns.adobe.com/flex/mx">
    <fx:Script>
        <![CDATA[
            import mx.events.FlexEvent;

            import spark.components.supportClasses.SkinnableComponent;

            private var _target:SkinnableComponent;

            public function get target():SkinnableComponent
```