review, or enter information, developers should correct this focus issue. The following code illustrates one way of resolving this issue:

```
var focusObject: IFocusManagerComponent;
// attach events for watching for when focus is lost and gained
addEventListener(Event.DEACTIVATE, losingFocus);
addEventListener(Event.ACTIVATE, gainingFocus);
...
// called when focus is leaving Flash Player
function losingFocus(e:Event):void
{
  focusObject = focusManager.getFocus();
}
// called when focus is returning to Flash Player
function gainingFocus(e:Event):void
{
  if (focusObject)
    focusManager.setFocus(focusObject);
  //else
    // set focus to most appropriate element
}
```

Focus for custom components

Proper focus management, including visual, keyboard, and programmatic focus, may not be provided by default for custom components. If custom components are used, developers must take steps to implement and test for proper keyboard, visual, and programmatic focus. The procedures for providing focus to custom components are not within the scope of this document and developers should consult the Flex API for more information.

Sending events

When creating an accessible application it may be necessary to explicitly fire an accessibility event. While these events are automatically fired in the accessibility implementation classes for each component, sending events manually is sometimes required. For example, in some cases a component can gain focus before a needed screen redraw event is sent. In this situation, the screen reader may not correctly announce the focused item. To solve this, an event can be sent using the `sendEvent()` method of the `flash.accessibility.Accessibility` class. See "Creating a Custom Accessibility Implementation" (*http://goo.gl/crgjZ*) in the Flex documentation for more information on `sendEvent()`.

The `sendEvent()` method takes three required parameters: a component name, the child id (typically `0` if the change is on the component itself), and the hexadecimal number representing the MSAA event to send. (A list of other available events that can be sent are defined in the winuser.h file supplied with the Windows SDK.) Events are sent through Flash Player and thus not all events will actually be sent to assistive technologies. The `sendEvent()` method can be used to send events signaling focus changes, value changes, name changes, state changes, selection changes, window reordering, and alerts, among others. In the example provided below the `sendEvent()` method is used to indicate that the `dfStartDate` field has received focus and to ensure that this focus is properly noticed by assistive technology.

```
import flash.accessibility.Accessibility;
...
static const EVENT_OBJECT_FOCUS:uint = 0x8005;
Accessibility.sendEvent(dfStartDate,0,EVENT_OBJECT_FOCUS);
...
```

Using the `sendEvent()` method does change the programmatic accessibility information exported by visual interface elements–such as the current focused object, names, values, roles, and states–but rather fires an event alerting assistive technology that the change occurred. This in turn causes the assistive technology to update its representation of the application to match the current reality of the component's accessibility