Developers must also ensure that users who rely exclusively on the keyboard for navigation do not get trapped in the Flex content. Provide a way to move the keyboard focus out of or past the Flex content. Generally this is achieved by ensuring that the `SeamlessTabbing` parameter of the `OBJECT` element embedding the Flex content is set to `true` (or not set at all, as it will default to true). To set this value using a `PARAM` element as a child of the embedding `OBJECT` element, use the following syntax:

```
<PARAM NAME="SeamlessTabbing" VALUE="true">
```

If an application provides its own focus management, the developer must ensure that the custom focus manager properly handles keyboard focus in and out of the Flex content.

Note: In some versions of Firefox, there is a known issue with the Flash Player plug-in that makes it difficult to get focus within SWF content. This may prevent keyboard users from being able to tab into the Flash Player plug-in. If the target audience will be using Firefox, developers should consider creating a shortcut keystroke that would force focus into the Flex content when activated. This can be achieved using JavaScript and the `.focus()` method or by following technique FLASH17 of the W3C WCAG 2.0 techniques for Flash (*http://www.w3.org/TR/WCAG-TECHS/FLASH17.html*).

Accessible Flex components

Flex 4 includes a variety of components and containers in the Spark and MX component sets that support accessibility. Each component has been thoroughly reviewed and tested for accessibility and interoperability with assistive technology. In developing the default set of components, one goal was to accelerate development of accessible applications. For the complete list of accessible Spark and MX components provided in Flex 4, see "Accessible Components and Containers" in the Adobe Flex 4 documentation (*http://goo.gl/xiTNu*). (A list of the relevant Spark components and containers is also provided later in this document.)

Accessibility for custom components

It is possible to create custom UI components that meet the different technical and functional accessibility requirements for access by users with disabilities. Detailed instructions for creating such components are beyond the scope of this document. In situations that require a custom accessible component, developers should plan full support for MSAA from the beginning of the development process to ensure the behavior of custom components is consistent with default Flex components and complies with operating system requirements.

Among other requirements, MSAA requires that accessible components must export name, state, and role information. Developers should ensure that this information is exported by their custom objects in conformance to MSAA as implemented in Flash Player. After MSAA support has been added to a control, it is important to validate that the component works with assistive technology. Assistive technologies, most notably screen readers, rarely implement the entire MSAA specification, particularly when operating with a plug-in or ActiveX control such as Flash Player. Some degree of coordination with screen reader vendors or screen reader-specific scripts may be required if a custom-built component deviates from the approach used in the default Flex component set or a new control type is introduced.

Creating an accessible custom component requires detailed knowledge of MSAA and the various properties that must be exposed for the component itself and its child elements. This approach is limited by what MSAA information can flow through Flash Player and is defined in the MSAA specification itself. Not all types of objects have a direct representation in MSAA. For example, MSAA does not provide a specification for how a calendar component should expose accessibility identity information, nor does it allow for explicit association between table header and data cells. In addition, assistive technologies such as screen readers expect certain MSAA structures and properties based on how SWF content has historically been rendered. As a result, some of these technologies may not function as expected with custom implementations.

In practice, this means that while developers can choose to develop custom, accessible components, they should be prepared for significant development costs due to the complex nature of MSAA and the need to make components work with a variety of assistive technologies. **For this reason, developers are strongly encouraged to use the default Flex components that include built-in accessibility support whenever possible**.