

```

</p>
<a href="#tab_menus" class="go_tab_menu">
  </a>
</div>
<div id="Menu_03" class="content">
  <h3><a id="QuickMenu_03" href="#QuickMenu_03">Menu 03
  </a></h3>
  <p>
    Menu 03...<br />
    Menu 03...<br />
    ...
    Menu 03...<br />
    Menu 03...<br />
  </p>
  <a href="#tab_menus" class="go_tab_menu">
    </a>
</div>
</body>

```

5.4.4 noscript 이용 방법

<그림 II - 24>는 noscript 기능을 이용하여 CSS만으로도 동작하도록 프로그래밍 한 드롭다운 메뉴의 예이다. 즉 Tab 키를 이용하여 ‘검색 사이트 보기’ 링크에 초점이 주어지면, ‘google’, ‘yahoo’, ‘naver’ 등의 하위 링크 목록이 나타나는데 이때 Tab 키를 눌러 하위 링크 목록으로 이동할 수 있다.



그림 II - 24. noscript 기능을 이용한 드롭다운메뉴

Shift + Tab 키를 누르면 Tab 키의 경우와 반대 순서로 이동한다. 화면 낭독 프로그램은 초점의 이동 순서에 따라 그 내용을 읽어 주어야 한다.

참고

- 센스/드림/JAWS : 모두 정상적으로 동작한다.

JavaScript 코드에서 밑줄로 표시된 부분은 '검색 사이트들 보기'에 초점이 주어졌을 경우 하위 링크 목록을 나타나게 하는 것이고, 초점이 없어졌을 경우에는 하위 링크 목록을 보이지 않게 하는 것이다. JavaScript가 실행되지 않을 때 화면에 하위 링크 목록이 모두 나타나도록 noscript에서 설정한다. 이 부분도 밑줄로 표시하였다.

1) CSS 코드 부분

O (Good)

```
<link type="text/css" href="reset.css" rel="stylesheet"/>
<style type="text/css" title="">
#selectbox {
```

```

padding: 5px 10px;
border: 1px solid #aaa;
width: 130px;
}
#selectbox a {
font-size: 12px;
color: #333;
text-decoration: none;
}
#sites {
display: none;
}
#sites {
list-style: none;
}
#sites a {
color: black;
text-decoration: none;
display: block;
width: 110px;
height: 10px;
padding: 10px;
border-bottom: 1px dashed #aaa;
font-size: 12px;
font-family: "dotum";
}
#sites .last a {
border-bottom: 0;
padding-bottom: 5px;
}
</style>
</head>

```

2) 스크립트 코드 부분

O (Good)

```
<script type="text/javascript">
<!--
var isOver = false;
var bindSelectbox = function() {
    var button = document.getElementById("button");
    button.onmouseover = button.onfocus = function() {
        isOver = true;
        document.getElementById("sites").style.display = "block";
    }
    button.onmouseout = function() {
        isOver = false;
        document.getElementById("sites").style.display = "none";
    }

    var menus
    = document.getElementById("sites").getElementsByTagName("a");
    for (var i=0; i<menus.length; i++) {
        var menu = menus[i];
        menu.onmouseover = menu.onfocus = function() {
            isOver = true;
            document.getElementById("sites").style.display
                = "block";
        }
        menu.onmouseout = menu.onblur = function() {
            isOver = false;
            setTimeout(function() {
                if (!isOver) {
                    document.getElementById("sites")
                        .style.display = "none";
                }
            }, 100);
        }
    }
}
window.onload = function() {
    bindSelectbox();
}
//-->
```

```
</script>

<noscript>
  #sites {
    display: block;
  }
</noscript>
```

3) HTML 코드 부분

O (Good)

```
<body>

<div id="selectbox">
  <strong><a href="#sites" id="button">검색 사이트들 보기▼
    </a></strong>
  <ul id="sites">
    <li><a href="http://google.com">http://google.com</a>
    </li>
    <li><a href="http://yahoo.com">http://yahoo.com</a></li>
    <li class="last">
      <a href="http://naver.com">http://naver.com</a></li>
  </ul>
</div>
<strong></strong>

</body>
```


III. 접근성 있는 Flex 제작기법

1. Flex 개요

Adobe Flex는 리치 인터넷 애플리케이션(RIA: Rich Internet Application) 콘텐츠를 개발할 수 있는 도구의 하나이다. RIA란 하나의 화면에서 처리되는 비즈니스 조직으로써 웹 브라우저를 기반으로 동작하지만, 데스크톱/Desktop) 프로그램과 같이 다양한 기능을 구현할 수 있는 웹 클라이언트 기술이다.

기존의 웹 환경은 클라이언트의 요구에 따라 서버의 응대 과정이 동기식으로 처리된다. 클라이언트가 서버에 어떤 웹 페이지를 요구하면 서버가 데이터를 가공하여 웹 페이지를 완성한 후에 클라이언트에게 제공한

다. 따라서 클라이언트가 웹 페이지의 일부 데이터만 요구하는 경우에도 웹 페이지를 구성하는데 필요한 모든 데이터를 서버로부터 가져와야 하므로 서버의 불필요한 자원 낭비가 발생한다.

RIA 환경에서는 웹 페이지의 불필요한 다운로드를 최소화 할 수 있다. 현재 화면에 보이는 웹 페이지에서 갱신이 필요한 데이터만을 서버에서 다운받아 화면에 표시할 수 있으므로 웹 페이지 전체를 갱신하는 불필요함이 사라지게 되고, 웹 페이지를 동적으로 만들 수 있게 되었다. 이러한 특징으로 인하여 RIA 애플리케이션이 전 산업 분야로 확산되고 있다.

Adobe Flex는 RIA 콘텐츠를 제작할 수 있는 도구 중의 하나로 강력한 화면 표시 기능, 통합적인 사용자 인터페이스, 강력한 서버 및 서비스 지원, JavaScript 애플리케이션과의 호환성을 제공하는 장점이 있다.

Flex로 개발된 웹 애플리케이션 콘텐츠를 재생하기 위해서는 Flash Player가 필요하다. Flash Player는 MS Windows, Mac OS X, Linux 등 대부분의 운영체제하에서 동작이 가능하며, Internet Explorer, FireFox, Opera 등 대표적인 웹 브라우저의 플러그인(plug-in)으로 동작한다. 근래에는 휴대폰 브라우저용 Flash player가 제공되어 Flex 애플리케이션 콘텐츠(이하 'Flex 콘텐츠'라고 한다)를 실행할 수도 있다.

Flex 콘텐츠는 웹 환경에서 데스크톱 애플리케이션 프로그램과 같은 사용자 인터페이스를 지원하며, 웹 페이지를 화려하고 생동감 있게 꾸며 줄 수 있다. 그러나 마우스를 사용하지 못하거나 화면 낭독 프로그램을 사용해야 하는 사용자들에게는 Flex 콘텐츠가 새로운 어려움을 주게 되었다.

우리나라의 경우에 2008년 하반기까지 화면 낭독 프로그램은 Flex 콘텐츠에 접근할 수 없었다. 그러나 한국 Adobe가 기술지원을 함으로써 국내

화면 낭독 프로그램들도 Flex 콘텐츠에 접근할 수 있는 길이 마련되었다.

III장에서는 Adobe Flex를 이용하여 접근 가능한 Flex 콘텐츠를 제작하는 기법에 대해 기술할 것이다. 누구나 이 문서에서 제시한 방법에 따라 Flex 콘텐츠를 제작한다면 개발된 Flex 콘텐츠는 적절한 보조기기를 사용하는 장애인들에게 필요한 최소한의 접근성을 보장할 것이다. 그러나 이 문서는 접근성을 제공하는데 필요한 충분한 정보를 제공하지는 못한다. 그 이유는 접근성 있는 Flex 콘텐츠를 제작하기 위해서는 접근성에 대한 충분한 이해와 함께 Flex를 이용한 웹 콘텐츠 제작기법에 대해서도 잘 알아야 하기 때문이다. 예를 들어 Flex 콘텐츠에 포함된 이미지에 대한 대체 텍스트가 해당 이미지를 충실하게 설명하지 못한다면 아무리 뛰어난 접근성 기술을 적용하더라도 대체 텍스트의 미흡함을 해소해줄 수 없다. 또한 이미지에 대한 설명이 완벽하다고 해도 이것을 Flex 콘텐츠에 포함시키는 방법을 모른다면 접근성 있는 Flex 콘텐츠를 만들 수 없다.

2. 웹 접근성과 Flex 콘텐츠

2.1 Flex 버전

이 문서에서 참고로 하는 Flex 콘텐츠 제작 도구는 Adobe Flex 3(이하 'Flex'라고 함)를 대상으로 한다. 만일 Adobe Flex 3가 아닌 다른 버전을 언급해야 할 경우에는 Flex가 아닌 온전한 명칭을 사용할 것이다.

Flex 콘텐츠를 실행하기 위해서는 Adobe Flash Player가 필요하다. 특히 Adobe Flash Player 9 또는 그 이후 버전(이하 'Flash Player'이라 한다)을 이용하면 시각장애인도 화면 낭독 프로그램을 이용하여 Flex 콘텐츠에 접근할 수 있다.

우리나라에서 구입이 가능한 화면 낭독 프로그램은 엑스비전테크놀로지사의 센스리더 프로페셔널 에디션(Sense Reader Professional Edition)(이하 ‘센스리더’라 한다), 실로암시각장애인복지관에서 개발한 드림보이스 7.0 (이하 ‘드림보이스’라고 한다) 및 미국 Freedom Scientific사의 Jaws for Windows 10.0이 있다. 이들 세 가지 화면 낭독 프로그램은 2009년도 버전부터 Flash Player를 지원하기 시작하였다. 세 가지 화면 낭독 프로그램에 관한 자세한 사항은 각 회사의 웹 사이트를 참고하라.

2.2 접근성 있는 Flex 콘텐츠 제작

Flex 콘텐츠가 접근성을 지원하기 위해서는 우선 해당 애플리케이션 콘텐츠의 접근성을 활성화시켜야 한다. Flex 콘텐츠의 접근성 기능은 초기 값이 비활성 상태이다. 따라서 Flex 콘텐츠를 접근 가능하게 만들려면 Flex 서버 전체의 접근성 기능을 활성화 시키던가, 아니면 애플리케이션 별로 접근성 기능을 활성화 시켜야 한다. 접근성 기능을 활성화 시키는 방법은 다음과 같다.

2.2.1 Flex Builder 3을 이용한 접근성 지원방법

Adobe Flex Builder 3를 사용하여 Flex 애플리케이션 콘텐츠를 개발하는 경우에는 <그림 III - 1>과 같이 프로젝트 프로퍼티 대화상자의 Flex Compiler 섹션에서 ‘Generate accessible SWF file(접근성 있는 SWF 파일 생성하기)’ 옵션을 선택하여 컴파일하면 접근성을 지원하는 Flex 콘텐츠가 생성된다.

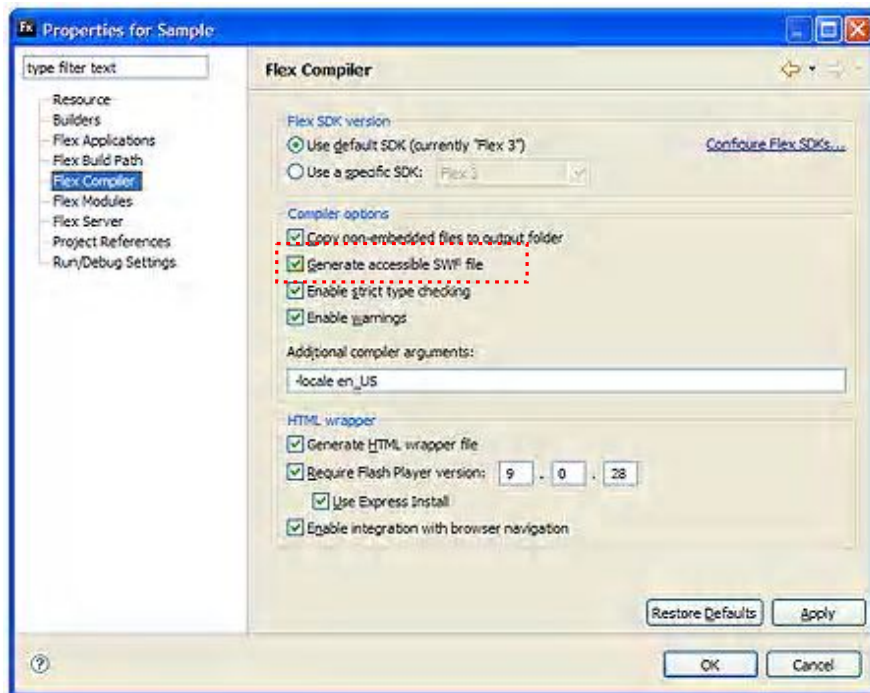


그림 III - 1. Flex Builder 3을 이용한 접근성 지원방법

2.2.2 Flex 콘텐츠 컴파일시 접근성 지원 방법

Flex 콘텐츠를 컴파일할 때마다 접근성을 지원하기 위해서는 아래와 같이 flex-config.xml 파일의 <accessible> 속성을 true('참')로 설정한다.

```
<mxml-compiler>
...
  <accessible>true</accessible>
...
</mxml-compiler>
```

2.2.3 커맨드-라인 컴파일러를 이용한 접근성 지원

커맨드-라인 컴파일러인 'mxmmlc'을 이용하여 파일을 컴파일하는 경우에 컴파일러로 하여금 접근성이 지원되는 SWF 파일을 생성하도록 알려주는 설정 변수(configuration variable)를 사용한다. 예를 들면 파일 c:/dev/myapps/appl.mxml을 컴파일하기 위한 커맨드-라인 컴파일러 옵션은 다음과 같다.

```
<mxmmlc -compiler.accessible c:/dev/myapps/appl.mxml>
```

또는

```
<mxmmlc -accessible c:/dev/myapps/appl.mxml>
```

위의 예제와 같이 커맨드 라인에 -compiler.accessible 또는 -accessible을 추가하여 컴파일 하면 접근 가능한 Flex 콘텐츠를 생성할 수 있다.

아래의 옵션을 선택하는 것은 프로젝트 디렉토리에 위치한 .actionScriptProperties파일 내의 컴파일러 노드가 접근성을 지원하는 SWF 파일을 생성하도록 지시하는 것과 같다.

```
<compiler additionalCompilerArguments="-locale en_US"
generateAccessible="true"> ...
```

2.2.4 개별 Flex 콘텐츠의 접근성 설정

만일 Flex 콘텐츠의 개발 시에 접근성을 지원하도록 제작되지 않았을 때에는, 웹 브라우저가 실행될 때 다음과 같이 query 파라미터와 accessible=true 값을 추가하면 접근성을 지원할 수 있다.

`http://www.mycompany.com/index.mxml?accessible=true`

이상의 네 가지 방법(2.2.1 - 2.2.4) 중 어떤 방법을 선택할 것인가는 개발자에게 달려있다. 어떤 방법을 선택하더라도 접근성을 지원하는 방법과 지원하지 않는 방법이 존재한다. 따라서 어떤 웹 사이트가 Flex 콘텐츠를 제공하는 경우에 클라이언트에서 보조 기술이 실행중인지를 검사하여, 보조 기술이 실행 중인 경우에는 접근성을 제공하는 Flex 콘텐츠 버전을 제공하도록 하고, 보조 기술이 실행되고 있지 않을 경우에는 접근성을 지원하지 않는 Flex 콘텐츠 버전을 제공하도록 하는 것도 가능하다.

3. Flex 콘텐츠 접근성 기술지침

3.1 (대체 텍스트 제공) '텍스트가 아닌 콘텐츠'에는 대체 텍스트를 제공하여야 한다.

국가표준 <항목 1.1>에 따르면, '텍스트가 아닌 콘텐츠'(이미지, 멀티미디어, 애니메이션 등)는 적절한 대체 텍스트를 제공하여야 화면 낭독 프로그램과 같은 보조 기술을 이용하는 장애인에게 그 내용을 전달할 수 있다. 이는 중요한 내용을 포함하고 있는 배경 이미지의 경우에도 마찬가지이다. 그러나 모든 '텍스트가 아닌 콘텐츠'에 대체 텍스트를 제공해야만 하는 것은 아니다.

3.1.1 요구조건

- (1) '텍스트가 아닌 콘텐츠'에는 적절한 대체 텍스트를 제공하여야 한다.
- (2) '텍스트가 아닌 콘텐츠'를 동적으로 교체하는 경우에는 대체 텍스트도 교체하여야 한다.

3.1.2 적용방법

화면 낭독 프로그램은 그래픽 이미지와 스테이지 상의 움직이는 요소의 의미를 구별할 수 없다. 따라서 그래픽 이미지나 애니메이션 등에 대한 대체 텍스트를 제공해야 한다. 필요하다면 콘텐츠 전체에 대한 대체 텍스트를 제공할 수도 있고, 콘텐츠를 구성하고 있는 개체 별로 대체 텍스트를 제공할 수도 있다.

가) toolTip의 사용

Flex 콘텐츠는 보통 이미지나 로더(loader) 컴포넌트를 사용하여 이미지를 보여준다. 이들 컴포넌트는 toolTip 속성을 이용하여 대체 텍스트를 화면 낭독 프로그램으로 전달할 수 있다. 따라서 Flex 콘텐츠의 대체 텍스트는 이미지 프로퍼티나 로더 컴포넌트 프로퍼티의 toolTip 속성을 이용한다. toolTip 속성을 제공하면 마우스를 이미지 위에 올려놓을 때에 toolTip 속성 값이 아래 <그림 III - 2>와 같이 화면에 표시된다.



그림 III - 2. tooltip 속성 사용 예제

<그림 III - 2>는 이미지에 tooltip 속성을 '아이스크림'으로 설정한 경우이며 이미지 위에 마우스를 올려놓았을 때의 화면을 캡처한 것이다.

이 예제를 위하여 작성한 코드는 아래와 같다.

O (Good)

```
<mx:Image width="60" height="56" source="assets/icecreampint.jpg"
      tooltip="아이스크림"
/>
```

나) Description 과 Name 필드

일반인에게는 필요하지 않지만 이미지에 대한 추가적인 설명을 보조 기술로 제공하기 위해서는 이미지의 description 프로퍼티를 사용할 수 있다. 아래의 코드는 위에 보인 그림에서 아이스크림에 대한 자세한 설명을 description 프로퍼티를 사용하여 제공하는 예이다. 코드에서 밑줄로

표시한 부분이 description 프로퍼티 부분이다.

O (Good)

```
<mx:Image width="60" height="56"
  source="assets/icecreampint.jpg"
  toolTip="아이스크림
  creationComplete="event.target.accessibilityProperties =
    new AccessibilityProperties();
    event.target.accessibilityProperties.description
    = '저희 아이스크림은 홈메이드를 위하여
    최고의 서비스를 제공합니다.'
/>
```

위의 예제와 같이 description 속성을 이용하여 이미지에 긴 설명문을 제공할 수 있다. 그러나 이미지에 대한 설명을 위해 항상 description 필드를 추가할 필요는 없다. 화면 낭독 프로그램으로 하여금 이미지에 대한 대체 텍스트를 쓸데없이 장황하게 읽어주는 것은 도리어 Flex 콘텐츠의 사용에 불편을 초래할 수도 있기 때문이다.

간단한 대체 텍스트를 제공하기 위해서는 ActionScript가 지원하는 accessibilityProperties.name을 이용하여 프로퍼티의 name 필드에 대체 텍스트를 제공한다.

description 필드와 name 필드의 기능은 동일하다. 화면 낭독 프로그램은 두 필드의 내용을 name 필드, description 필드의 순서로 읽어준다. 따라서 두 필드를 같은 내용으로 채우면 두 번 읽어주므로 주의할 필요가 있다. 일반적으로 description 필드는 한글 25자 이상인 경우에 사용하며, 그 이하인 경우에는 name 필드를 사용한다.

다) 대체 텍스트 교체

Flex 콘텐츠는 동적으로 이미지를 교체할 수 있다. 이때 원래의 이미지와 교체되는 이미지의 의미나 내용이 다른 경우에는, 교체되는 이미지의 대체 텍스트도 교체하여야 한다. 교체 방법으로는 tooltip, description, name 등을 이용할 수 있다.

라) 올바른 대체 텍스트

‘텍스트가 아닌 콘텐츠’에 제공되는 대체 텍스트는 화면에 표시되는 이미지를 대신해서 보조 기술(예를 들면 화면 낭독 프로그램)을 통하여 제공되는 정보이므로 그 내용이 적절하여야 한다.

바람직한 방법은 이미지를 화면에 제공하는 이유를 생각하여 대체 텍스트의 내용을 결정하는 것이 이미지의 모습을 그대로 설명하는 것보다 낫다. 예를 들어 투명한 플라스틱 병에 생수가 가득 들어있는 이미지에 대한 대체 텍스트를 ‘투명한 액체가 담긴 플라스틱 병’이라고 하기보다는 ‘생수병’이라고 하는 것이 더 적합하다.

‘텍스트가 아닌 콘텐츠’ 요소에 대체 텍스트를 제공하는 것은 어려운 일이 아니다. 가장 중요한 점은 ‘언제, 어떤 내용의 대체 텍스트를 제공할 것인가’를 고려하여야 한다는 점이다. 또한 대체 텍스트로 인하여 화면 낭독 프로그램 사용자에게 어떠한 영향을 주게 되는가를 생각하는 일이다.

3.2 (자막 제공 방법) 오디오가 있는 영상물은 오디오와 동기 되는 대체 텍스트를 제공하여야 한다.

국가표준 <항목 1.2>에 따르면, 비디오 등과 같이 오디오와 영상정보가 동시에 제공되는 경우, 음향정보와 화면 해설을 자막(caption)과 같이 시각적으로 제공하여야 한다.

3.2.1 요구조건

- (1) 오디오는 대체 매체(예를 들면 자막)를 제공하여야 한다.
- (2) 자막 파일은 W3C가 규정한 Timed Text XML(DFXP) 형식이나 FLV 큐 포인트 형식이어야 한다.

3.2.2 적용방법

가) 자막의 컴포넌트의 사용

Flex는 고품질의 Flash 비디오 콘텐츠를 전달하는데 사용될 수 있다. 콘텐츠와 함께 제공되는 오디오 정보는 동기화된 대체 텍스트를 자막(caption)의 형태로 송출하여야 한다. Flash 콘텐츠에 자막정보를 제공할 수 있는 확실한 방법은 런타임(runtime)에 자막을 송출하는 것이다.

일부 자막 소프트웨어(Hi-Caption Studio 및 MAGpie 등)는 자막 데이터를 내장한 고유한 XML파일을 생성한다. Captionate는 XML 파일 또는 FLV 큐 포인트를 이용하여 자막을 추가할 수 있다.

자막 소프트웨어를 이용하여 생성한 자막 파일을 Flex에서 사용하기 위해서는 추가적인 노력이 필요하다. Flash 콘텐츠용 자막 생성과 관련한 방법은 해당 자막 소프트웨어 개발사가 제공하는 정보를 참고하라. 또한 Flash 콘텐츠의 자막에 관한 내용은 IV장을 참고하라.

3.3 (색상과 접근성) 콘텐츠의 색상은 중요한 정보 제공 수단으로 사용하지 않는다.

국가표준 <항목 1.3>에 따르면, 색상으로 표현된 정보는 색상을 배제하여도 원하는 내용을 전달할 수 있어야 한다. 따라서 웹 콘텐츠는 화면을 흑백으로 바꾸더라도 명암이나 패턴으로 콘텐츠가 구분하여 표시하고자 하는 내용을 인지할 수 있어야 한다.

3.3.1 요구조건

- (1) 콘텐츠를 제작할 때 다양한 색상을 사용할 수 있다. 그러나 색상을 선택할 때 색상을 이용하여 중요한 정보를 제공하지 않아야 한다.
- (2) 콘텐츠는 배경색과 전경색 간의 대비가 충분히 차이가 나도록 설계되어야 한다.
- (3) 텍스트나 버튼 등 화면 구성 요소의 크기는 적절해야 한다.

3.3.2 적용방법

가) 색상

Flex에서 색상을 선택할 때에는 색각이상자와 저시력자를 고려하여야 한다. 즉 콘텐츠를 개발할 때에 색상만으로 필요한 정보를 제공하지 않아야 한다. 예를 들어 파란 버튼과 빨간 버튼을 제공하고 “파란 버튼을 누르시오” 또는 “빨간 버튼을 누르시오” 라고 사용법이 제시된 콘텐츠는 흑백화면 사용자 또는 색각이상자가 구분할 수 없기 때문이다.

색상과 함께 추가적인 정보를 제공하면 이 문제가 해결될 수 있다. 예를 들어 “앞으로 이동하려면 오른쪽의 빨간 버튼을 클릭하십시오” 또는 “뒤로 이동하려면 왼쪽의 파란 버튼을 클릭하십시오” 와 같이 버튼의 위치 정보를 함께 제공하면 색각이상자도 접근이 가능한 콘텐츠로 변한다.

나) 대비

색상과 관련된 두 번째 문제는 가독성을 향상시키는 방법이다. 즉 전경색과 배경색의 대비를 충분히 차이가 나게 하여 텍스트가 잘 보이도록 한다. 콘텐츠의 가독성을 평가하는 한 가지 방법은 흑백 화면에서 콘텐츠를 충분히 판독할 수 있는가를 확인하는 것이다. 색상의 대비가 낮으면 화면에 보이는 요소를 식별하기가 어렵다.

다) 화면 구성요소의 크기

콘텐츠를 만들 때 화면에 표시되는 화면 구성 요소(예를 들어 텍스트, 버튼 등)의 크기를 브라우저에서 조절할 수 있어야 한다. 만일 크기를 조절할 수 없다면 텍스트의 글씨 크기는 12호 이상을 사용한다.

3.4 (키보드의 이용) 마우스로 할 수 있는 모든 컨트롤은 키보드로도 제어가 가능해야 한다.

국가표준 <항목 2.4>에 따르면, 웹 콘텐츠는 키보드 또는 장애를 극복하도록 도와주는 여러 가지 입력 장치를 사용하는 경우에도 웹 콘텐츠가 제공하는 모든 기능을 사용할 수 있어야 한다. 예를 들어 마우스를 사용할 수 없는 장애인들도 마우스를 사용할 수 있는 사용자와 같이 키보드만으로 웹 콘텐츠가 제공하는 모든 기능을 동일하게 수행할 수 있어야 한다. 이 검사항목은 웹 브라우저의 이용 뿐 아니라 웹 콘텐츠가 제공하는 기능을 이용하는 경우에도 해당된다.

3.4.1 요구조건

- (1) 애플리케이션 콘텐츠는 키보드만으로 이용이 가능하여야 한다.
- (2) 버튼에는 단축키를 제공하여 접근성을 높인다.

3.4.2 적용방법

가) 키보드 지원

Flex가 제공하는 접근성 지원 컴포넌트로 개발한 애플리케이션은 마우스 이벤트를 키보드로 접근 가능하도록 변환한다. Flex는 접근성을 지원하는 애플리케이션 콘텐츠를 개발할 수 있도록 28개의 접근성 관련 컨트롤을 제공한다. 이들 접근성을 지원하는 컴포넌트에 대한 사항은 <III - 3.7 접근성 지원 컴포넌트의 사용> 및 <III - 5. 접근성 지원 Flex 컴포넌트>를 참고하라. 이 요구조건을 검사하기 위해서는 키보드와 화면 낭독

프로그램을 동시에 이용하거나 때로는 화면 낭독 프로그램을 사용하지 않고 키보드만 사용하여 조작할 수 있는가를 점검하면 된다.

나) 드래그 앤 드롭

드래그 앤 드롭(drag-and-drop) 이벤트는 마우스를 사용하는 것을 전제로 제공되는 기능이다. 장애인 중에서도 특히 전맹, 일부 저시력자, 지적장애인들은 마우스의 사용이 어렵다. 따라서 드래그 앤 드롭 기능도 키보드만으로 사용할 수 있어야 한다. 예를 들면 온라인 쇼핑몰에서 쇼핑 카트에 상품을 드래그(drag)하는 과정을 Enter 키만을 이용하여도 동일한 기능을 수행할 수 있어야 한다.

한 가지 방법은 링크 컴포넌트를 이용하는 방법이다. 즉 링크 컴포넌트에 상품 이미지를 제공하면 화면에는 상품 이미지가 표시되지만, 해당 상품 이미지의 대체 텍스트를 읽어주는 동안 Enter 키를 치면 해당 상품이 쇼핑 카트에 추가되도록 하는 것이다. 이것이 가능한 이유는 링크 컴포넌트가 키보드 접근을 지원하며, 이미지를 아이콘과 같이 표시할 수 있기 때문이다.

다) 복잡한 애플리케이션 콘텐츠의 단축키 설정

다수의 컨트롤이 필요한 복잡한 애플리케이션 콘텐츠는 단축키를 이용하여 사용하도록 하는 것이 좋다. 일부 사용자에게는 하나의 키를 조작하는 것도 많은 노력을 필요로 한다. 따라서 여러 차례 이벤트를 발생시키기 위하여 누르는 키의 수를 줄일 수 있는 단축키를 제공하는 것이 바람직하다.

예를 들어 도움말 화면으로의 빠른 전환을 제공하기 위해 단축키로 ‘?’

키를 사용할 수 있다. 이것은 사용자들이 '?' 키를 누를 때마다 도움말 화면으로 이동시킨다. 이러한 기능을 수행하기 위해서는 ActionScript가 제공하는 listener 개체를 사용한다. 즉 단축키를 생성하기 위해서는 아래의 프로그램과 같이 listener 이벤트를 정의하고, listener 이벤트에 대응하는 스크립트를 작성하여야 한다.

O (Good)

```
KeyListener = new Object();
KeyListener.onKeyDown = function() {
    if (Key.getAscii() == 63) { // 키보드 입력이 '?'키이면...
        // call same code that button's click handler calls
    }
}

Key.addListener(KeyListener);
```

3.5 (읽는 순서의 설정) 콘텐츠의 읽는 순서는 논리적이어야 한다.

국가표준 <항목 3.1>에 따르면, 웹 콘텐츠는 화면에 표시되는 구성요소의 나열 순서와 보조 기술을 통하여 전달되는 정보의 순서가 일치하여야 한다. 만일 그 내용이나 순서가 일치하지 않거나 논리적으로 상이할 경우에는 웹 콘텐츠를 화면을 통하여 인지하는 사람과 보조 기술을 통하여 인지하는 사람 간에 차이가 발생한다.

또한 웹사이트의 사용성을 높이기 위하여 웹사이트가 제공하는 모든 웹 페이지의 구조, 형태, 사용법 등이 통일성을 가져야 한다. 필요시에는 웹 페이지의 시작 부분에 사용법이나 구도에 관한 정보를 제공한다.

3.5.1 요구조건

- (1) Flex 콘텐츠의 화면구성 요소를 읽어주는 순서는 논리적이어야 한다.
- (2) 화면구성 요소의 배치와 이동 순서는 모든 웹 페이지 간에 일관성이 있어야 한다.
- (3) 웹 페이지의 레이아웃, 구조, 사용법이 복잡한 Flex 콘텐츠는 사용법이나 구도에 관한 정보를 제공하여야 한다.

3.5.2 적용방법

화면에 표시되는 콘텐츠의 배치와 이동 순서는 화면 왼쪽상단에서 화면 오른쪽 아래 방향이라는 전통적인 기준을 따라야 한다. 그러나 SWF 파일에서 보조 기술로 제공하는 콘텐츠 정보의 순서는 예상과 같이 전통적인 기준을 따르지 않는다. 단순한 Flex 콘텐츠의 경우에는 읽는 순서가 큰 문제가 되지 않을 수 있지만, 복잡한 Flex 콘텐츠의 경우에 콘텐츠의 읽는 순서를 통제하지 못하면 콘텐츠의 구조나 내용을 이해하기 어렵게 된다.

가) 단순한 사용자 인터페이스

Flex 콘텐츠의 읽는 순서를 제어하는 가장 쉬운 방법은 사용자 인터페이스를 비교적 단순하게 구성하는 것이다. 예를 들어 아코디언(accordion)이나 탭 내비게이터(tab navigator)를 사용한 콘텐츠는 사용자들이 이동하는 순서가 명확하므로 화면 낭독 프로그램이 읽어주는 순서도

명확하다.

트리 컨트롤(Tree Control), 데이터 그리드(Data Grid), 텍스트 영역(Text Area)으로 구성된 Flex 콘텐츠의 경우에 화면 낭독 프로그램은 트리 컨트롤, 데이터 그리드, 텍스트 영역의 순서로 콘텐츠를 읽어주어야 할 것이다. 실제로 화면 낭독 프로그램이 타이틀(title), 트리 컨트롤, 데이터 그리드, 텍스트 상자(text box), 그리고 링크(link)의 순서로 읽어준다면 화면에 표시된 순서와 읽는 순서가 동일하기 때문에 추가로 보완해야 할 사항은 없다. 이처럼 개발자들은 화면 낭독 프로그램을 이용하여 화면에 표시된 순서와 실제로 화면 낭독 프로그램이 읽어 주는 순서가 동일한지를 확인해야 한다.

이외에도 접근성을 지원하는 28개 컴포넌트가 화면 낭독 프로그램을 지원하는 기능은 <III - 5. 접근성지원 Flex 컴포넌트>를 참고하라.

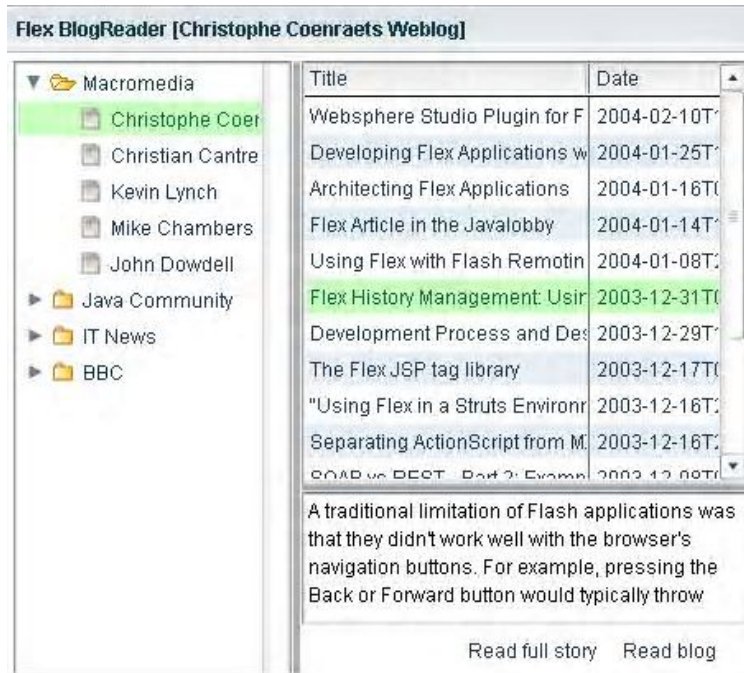


그림 III - 3. 트리 컨트롤(Tree Control), 데이터 그리드(Data Grid), 텍스트 영역(Text Area)으로 구성된 Flex 콘텐츠 예제

나) 복잡한 응용 콘텐츠

복잡한 레이아웃(layout)을 가진 Flex 콘텐츠는 화면 낭독 프로그램 사용자가 쉽게 이해하기 어렵다. 아래의 Flex 콘텐츠는 화면에 9개의 상품을 한 줄에 3개씩 3줄로 나열한 예제이다. 각 상품은 이미지와 레이블(label)로 구성되어 있다. 그런데 이미지와 레이블이 서로 다른 컴포넌트이므로 레이블의 읽는 순서와 이미지의 나열 순서가 일치하지 않는다. 따라서 Flex 콘텐츠는 읽는 순서가 화면에 표시된 순서와 일치하도록 만들어야 한다.



그림 III - 4. Tab 키에 의한 이동순서 예제

화면 낭독 프로그램의 읽는 순서를 제어하기 위해서는 MXML 파일의 tabIndex 속성 값을 설정해야 한다. SWF 파일에서 콘텐츠의 읽는 순서와 Tab 키에 의한 이동 순서는 동일하다. 즉 화면 낭독 프로그램은 tabIndex 목록에 열거된 순서대로 콘텐츠를 읽어주기 때문에, 텍스트 필드(text field) 뿐 아니라 초점이 주어지지 않는 모든 인터페이스 컨트롤을 포함한 모든 개체에 tabIndex 값을 제공한다.

아래의 코드는 위에 보인 <그림 III - 4>에서 첫 번째 줄의 콘텐츠에 tabIndex 값을 지정하는 프로그램이다. tabIndex 값을 지정하는 부분은 밑줄로 표시하였다. 마찬가지로 방법으로 나머지 2줄의 콘텐츠에도 tabIndex 값을 지정할 수 있다. 여기서 중요한 점은 응용 콘텐츠에 포함된 모든 개체의 tabIndex 값을 지정해야 하는 점이다. 만일 어떤 한 개체의 tabIndex 값을 지정하지 않아도 읽는 순서는 원래 상태로 돌아감을 유의하여야 한다.

O (Good)

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
  layout="absolute">
  <mx:Style>
    Application {
      backgroundColor: #FFFFFF;
      fontFamily: "verdana", "돋움체";
      fontSize: 12;
    }
    Tooltip{
      fontSize: 12;
      fontFamily: "verdana", "돋움";
    }
  </mx:Style>
  <mx:VBox width="100%" height="100%" paddingTop="10"
    paddingLeft="10" horizontalAlign="left" verticalAlign="top"
    horizontalScrollPolicy="off">

    <mx:Panel title="쇼핑몰 정보소개" tabIndex="1">
      <mx:HBox>
        <mx:VBox width="120" horizontalAlign="center">
          <mx:Image source="i1.jpg" toolTip="캔버스화"
            tabIndex="2" width="80" height="70"/>
          <mx:Label text="30000원" tabIndex="3">
        </mx:VBox>
        <mx:VBox width="120" horizontalAlign="center">
          <mx:Image source="i2.jpg" toolTip="부츠"
            tabIndex="4" width="80" height="70"/>
          <mx:Label text="40000원" tabIndex="5">
        </mx:VBox>
        <mx:VBox width="120" horizontalAlign="center">
          <mx:Image source="i3.jpg" toolTip="남성화"
            tabIndex="6" width="80" height="70"/>
          <mx:Label text="60000원" tabIndex="7">
        </mx:VBox>
      </mx:HBox>
    </mx:Panel>
  </mx:VBox>
</mx:Application>
```

```

        </mx:VBox>
    </mx:HBox>
    .
    .
    .
</mx:Panel>
</mx:VBox>
</mx:Application>

```

다) 복잡한 웹 페이지의 사용법 제공방법

Flex 콘텐츠는 기존의 HTML 기반의 웹 콘텐츠와는 동작 방법에서 매우 큰 차이가 있다. 우선 HTML 기반의 웹 콘텐츠는 웹 페이지 단위로 갱신이 이루어 졌으나 Flex 콘텐츠에서는 필요한 부분만 정보가 갱신된다. 따라서 처음 접한 사용자들, 특히 화면 낭독 프로그램 사용자들은 사용방법이 기존의 웹 콘텐츠와 매우 달라 당황하게 된다. 이러한 문제점을 해결하기 위하여 Flex 콘텐츠의 시작부분에 웹 사이트의 구성, 사용하는 컨트롤, 작업을 종료하는 방법 등에 대한 정보를 제공할 필요가 있다.

(1) description 프로퍼티를 이용한 사용법 제공

사용법을 제공하는 가장 간단한 방법은 Flash 콘텐츠에서와 같이 description 프로퍼티를 사용하여 대체 텍스트를 제공하는 것이다. description 프로퍼티를 이용하여 제공한 사용법에 관한 설명은 응용 콘텐츠가 처음으로 로딩될 때마다 읽어준다. 따라서 사용법에 대한 설명은 상황하게 작성하지 않는다.

아래의 프로그램은 어떤 블로그 리더 스크립트를 Flex 콘텐츠용으로 작성한 것이다. 우선 description을 지정하는 함수(function)를 정의한다. 이

예제의 경우에 함수명은 accessibleInit()이다.

O (Good)

```
function accessibleInit() {  
    var desc = createObject("TextInput","desc",0);  
    desc.x = 0;  
    desc.y = 0;  
    desc.width = 0;  
    desc.height = 0;  
    desc.accessibilityProperties = new AccessibilityProperties();  
    desc.accessibilityProperties.description= '사용법:  
    Flex 블로그의 사용자가 이 콘텐츠를 사용하려면  
    애플리케이션을 실행시킨 후에 폼 모드를 활성화 시키시오.'  
}
```

이어서 <mx:Application> 요소에서 accessibleInit() 함수를 호출한다.

O (Good)

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"  
    mx:Application xmlns:mx=""  
        creationComplete="accessibleInit();  
    pageTitle="Flex 블로그리더"  
>
```

(2) 새 창을 이용한 방법

사용법이 매우 복잡한 경우에는 사용법에 관한 설명을 별도의 화면으로 구성하는 것이 바람직하다. 즉 버튼이나 링크를 클릭하면 새 창을 열고 콘텐츠의 사용법을 제공하는 것이다. 이 때 사용법을 소개하는 새 창 열기로 이동하는 버튼이나 링크는 콘텐츠를 다운로드한 후에 화면 낭독 프로그램이 첫 번째로 읽어주도록 하는 것이 좋다. 일반적으로 이러한

목적으로 쓰이는 링크나 버튼은 화면에는 보이지 않는 ‘숨긴 링크’ 또는 ‘숨긴 버튼’으로 대개 콘텐츠의 첫 부분에 위치시킨다. 이들 숨긴 요소들은 화면에는 나타나지 않으나 화면 낭독 프로그램에 의하여 읽어주기 때문에 시각장애인에게 매우 편리하다.

또한 사용법 소개용 새 창 열기 링크나 버튼은 화면의 하단에 배치하여 일반인들에게도 사용법에 관한 정보를 제공할 수 있다.

3.6 (플러그인 정보 제공) 콘텐츠에 포함된 플러그인은 사용자가 설치할 수 있어야 한다.

국가표준 <항목 4.1>에 따르면, 웹 페이지에 포함된 콘텐츠를 구성하는데 필요한 모든 스크립트, 플러그인 등에 대한 정보와 링크는 제공되어야 한다. 또한 사용자가 설치할 수도 있어야 한다.

3.6.1 요구조건

- (1) 콘텐츠를 사용하기 위해서 필요한 플러그인은 사용자가 설치하여 실행할 수 있어야 한다.
- (2) 플러그인을 사용자가 설치할 수 없다면 플러그인의 설치방법 등을 제공하여야 한다.

3.6.2 적용방법

Flex 콘텐츠는 실행을 위해서 다양한 플러그인을 필요로 하는데 이것은 자동적으로 다운로드 받아 설치할 수 있어야 한다. 만일 자동적인 설

치가 불가능 하거나 설치하는 과정에서 사용자의 개입이 필요할 경우에는 해당 플러그인을 제공하는 웹 사이트 정보, 플러그인의 설치방법, 사용법 등에 관한 정보를 제공하여야 한다.

3.7 (접근성 지원 컴포넌트의 사용) Flex 콘텐츠 개발시에 사용하는 컴포넌트는 Flash CS4가 제공하는 접근성 지원 컴포넌트를 우선적으로 사용한다.

국가표준 <항목 4.1>에 따르면, Flex 콘텐츠가 제공하는 중요한 정보는 보조 기술로 제공될 수 있어야 한다. 또한 국가표준 <항목 2.4>에서 요구하는 바와 같이 키보드를 이용하여 Flex 콘텐츠를 사용할 수 있어야 한다. 따라서 Flex 콘텐츠를 제작하는 경우에 이러한 요구조건을 만족하기 위해서는 Flex가 제공하는 접근성 지원 컴포넌트를 사용하여야 한다.

3.7.1 요구조건

- (1) Flex 콘텐츠를 개발 시에는 Flex가 제공하는 접근성 지원 컴포넌트들을 우선적으로 사용한다.
- (2) 사용자가 컴포넌트를 자체적으로 개발하여 사용할 경우에 이들 컴포넌트는 접근성을 지원하도록 개발되어야 한다.

3.7.2 적용방법

가) 접근성 지원 컴포넌트

접근 가능한 Flex는 콘텐츠를 개발할 수 있도록 접근성을 지원하는

28개의 컨트롤을 제공한다. 이들 28개의 컴포넌트를 사용하면 대부분의 접근성과 관련한 실제적인 문제들(예를 들면 대체 텍스트의 제공, 레이블의 제어, 키보드 접근성 지원 등)을 자동적으로 해결할 수 있다. 28개의 컴포넌트에 대한 설명은 이 문서의 <III - 5. 접근성지원 Flex 컴포넌트>를 참고하라.

나) 사용자 개발 컴포넌트

컴포넌트가 접근성을 지원하기 위해서는 MSAA(Microsoft Active Accessibility) 규격에 의거하여 컴포넌트의 역할과 상태에 관한 정보를 알려줄 수 있어야 한다. 만일 개발자들이 컴포넌트를 직접 개발하여 사용하는 경우에 개발할 Flex 컴포넌트들이 기존의 컴포넌트와 동일한 특성을 가지도록 개발 초기부터 MSAA에 대한 검토와 구현계획을 세워야 한다.

화면 낭독 프로그램은 MSAA의 요구조건을 전부 지원하지는 않는다. 대개 12가지 기본적인 컨트롤을 지원하며, 이들 컨트롤은 HTML 표준에서도 지원하고 있다. 사용자가 기존의 컴포넌트를 변형하던지 새로운 컨트롤을 추가한 컴포넌트를 만들어 사용하고자 할 경우에는 화면 낭독 프로그램 개발자와의 협력도 강구하여야 한다. 우리나라의 화면 낭독 프로그램 제작사에 관한 정보는 <I. 리치 인터넷 애플리케이션의 접근성>을 참고하라.

Flex에서 제공하는 컴포넌트는 애플리케이션 콘텐츠의 빠른 개발을 위하여 제공하는 것이다. 사용자가 접근성을 지원하는 컴포넌트를 개발하여 사용하는 것은 MSAA나 화면 낭독 프로그램과의 호환성을 고려한다면 평범한 일은 아니다. 따라서 가능한 한 Flex가 제공하는 컴포넌트를 사용할 것을 권장한다.

4. Flex 콘텐츠의 접근성 평가방법

이 문서에서는 접근성 있는 콘텐츠를 개발하는데 있어서 매우 제한적인 지침을 제공할 뿐이다. 따라서 Flex 개발자는 접근성을 평가하기 위한 다양한 방법을 적용해 보아야 한다. 아래의 체크리스트는 Flex 부분과 관련한 접근성 검사목록을 발췌하여 정리한 것이다.

또한 Flex 애플리케이션 콘텐츠를 포함하고 있는 웹 콘텐츠의 평가방법에 대해서는 <I - 4. RIA 콘텐츠 접근성 평가방법>을 참고하라.

| 번호 | 항목 | 검사항목 | 국가표준 |
|----|--------|--|------|
| 1 | 대체 텍스트 | (1) '텍스트가 아닌 콘텐츠에 대해서는 적절한 대체 텍스트를 제공하고 있는가? (2) 동적으로 교체된 이미지는 교체되는 이미지에 알맞게 대체 텍스트도 교체하고 있는가? | 1.1 |
| 2 | 자막 | (3) 멀티미디어 콘텐츠는 동기화된 자막을 제공하고 있는가? | 1.2 |
| 3 | 색상 | (4) 색상 이외에도 명암이나 패턴으로 콘텐츠 구분이 가능한가? (5) 흰 바탕에 밝은 회색글자처럼 판독이 어려운 색 조합을 피하고, 흰 바탕에 검정 글자처럼 대비 차이가 큰 색 조합을 사용하고 있는가? (6) 저시력자나 다양한 환경의 사용자도 텍스트를 쉽게 읽을 수 있도록 적절한 크기로 텍스트를 제공하고 있는가? | 1.3 |

| | | | |
|---|-------|--|-----|
| 4 | 키보드 | <p>(7) 키보드만으로 콘텐츠가 제공하는 모든 기능에 대한 제어가 가능한가?</p> <p>(8) 초점이 주어지는 것만으로 상황이 바뀌지 않는가?</p> <p>(9) 하위 메뉴를 제공하는 주 메뉴의 경우, 키보드만으로도 하위 메뉴를 이용할 수 있는가?</p> | 2.4 |
| 5 | 읽는 순서 | <p>(10) 키보드로 폼 컨트롤, 링크, 객체 사이를 이동할 때, 논리적 이동 순서가 적절한가?</p> <p>(11) 복잡한 콘텐츠에 대해서는 사용법, 구도, 레이아웃에 관한 정보를 제공하는가?</p> | 3.2 |
| 6 | 플러그인 | <p>(12) 콘텐츠를 사용하는데 필요한 플러그인은 사용자가 설치하여 사용할 수 있는가? ※ 설치할 수 없다면 설치 방법을 제공하여야 함</p> <p>(13) 만일 플러그인의 설치가 불가능하다면 설치 방법을 제시하고 있는가?</p> | 4.1 |

4.1 대체 텍스트 제공

4.1.1 체크리스트

- (1) ‘텍스트가 아닌 콘텐츠’에 대해서는 아래와 같이 적절한 대체 텍스트를 제공하고 있는가?
 - (2) 동적으로 교체된 이미지는 아래와 같이 그 이미지에 알맞게 대체 텍스트도 교체하고 있는가?
- ‘텍스트가 아닌 콘텐츠’에 대해서는 대체 텍스트를 제공한다. 이미지, 멀티미디어 콘텐츠, 그래픽 아이콘, 그래픽 글자 등은 비록 텍스트의 형태를 하고 있더라도 코드로 사용될 수 없다. 즉 이들은 보조 기술에 제공되더라도 그 내용을 알 수 없다. 따라서 보조 기술이 코드로 인식할 수 없는 모든 객체를 ‘텍스트가 아닌 콘텐츠’라고 정의할 수 있다. ‘텍스트가 아닌 콘텐츠’는 보조 기술이 인지할 수 있는 대체 텍스트를 추가로 제공하여야 한다.
 - 만일 ActionScript로 교체하는 ‘텍스트가 아닌 콘텐츠’가 있다면 교체되는 ‘텍스트가 아닌 콘텐츠’에 대한 대체 텍스트도 교체해주어야 화면에 표시된 콘텐츠와 대체 텍스트가 일치하게 된다.
 - ~링크, ~버튼, ~로고, ~바로가기, ~메뉴와 같이 중복될 수 있는 내용을 대체 텍스트로 제공하지 않는다. 예를 들어 어떤 그래픽 버튼을 W3C 웹 사이트에 링크시켜 놓고, 대체 텍스트를 ‘W3C 바로가기 버튼’으로 설정하였다고 가정하면, 이 그래픽 버튼을 누를 때마다 화면 낭독 프로그램은 “버튼, W3C 바로가기 버튼”이라고 읽어

준다. 여기서 ‘버튼’을 두 번 읽지 않도록 하려면, 대체 텍스트를 ‘W3C 바로가기’나 ‘W3C 웹 사이트’로 설정하면 된다. 이것은 메뉴나 이미지 링크, 로고의 경우에도 해당된다.

- 대체 텍스트는 콘텐츠의 내용을 파악할 수 있는 핵심적인 설명을 제공해야 한다. 대체 텍스트는 콘텐츠를 직관적으로 나타낼 수 있는 내용으로 구성하는 것이 바람직하다. <III- 3.1.2 라) 올바른 대체 텍스트>에서 예를 든 것처럼, 투명한 플라스틱 병에 생수가 가득 들어있는 이미지에 대한 대체 텍스트로는 ‘투명한 액체가 들어있는 플라스틱 병’이라고 하기보다는 ‘생수병’이라고 하는 것이 바람직하다.
- 대체 텍스트가 필요하지 않은 경우에는 대체 텍스트를 제공하지 않는다. 때로는 대체 텍스트를 추가함으로 인하여 혼란을 야기시키는 경우가 있다. 예를 들어 글머리표는 “글머리표”라고 읽어주는 것이 도리어 혼란을 야기할 수 있다. 이런 경우에는 대체 텍스트를 제공하지 않는다.
- 교체되는 이미지의 대체 텍스트가 교체 전과 동일하면 대체 텍스트를 교체할 필요가 없다. 교체하는 이미지의 대체 텍스트가 교체하기 전과 동일할 경우에는 대체 텍스트를 교체할 필요가 없다. 스크립트를 이용하여 콘텐츠를 동적으로 교체하는 것은 불필요한 자원의 낭비를 초래하므로 가능한 한 대체 텍스트를 교체하지 않는다. 그러나 이미지의 내용이 교체 전과 비교하여 확연한 차이를 보일 경우에는 대체 텍스트도 교체하여야 한다.

4.1.2 관련 국가 표준 : 항목 1.1

4.1.3 Flex 프로그래밍 지침 : 항목 3.1

4.2 자막 제공 방법

4.2.1 체크리스트

(3) 멀티미디어 콘텐츠는 동기화된 자막을 제공하고 있는가?

- Flash를 이용한 오디오와 비디오 콘텐츠에는 FLV 또는 H.264 규격의 비디오에 간단히 자막을 추가할 수 있다. Flash에 의하여 만들어진 동영상 파일에는 아래 그림과 같이 자막이 포함되어야 한다.



그림 III - 5. 자막파일을 제공한 예제

- 필요에 따라서 자막 표시 위치의 변경, 자막 폰트의 변경, 전체 화면보기시의 폰트 크기 변경이 가능하다면 접근성은 더욱 향상된다. 아래 그림은 자막을 가독성 있는 폰트로 교체한 화면 모습이다.



그림 III - 6. 가독성이 좋은 폰트로 변경한 모습

- 동기화된 자막을 제공하는 것이 원칙이지만, 콘텐츠의 내용을 충분히 이해할 수 있는 원고를 제공하는 것도 허용된다.

4.2.2 관련 국가 표준 : 항목 1.2

4.2.3 Flex 프로그래밍 지침 : 항목 3.2

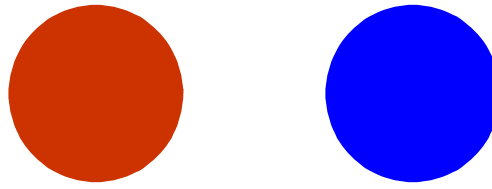
4.3 색상과 접근성

4.3.1 체크리스트

(4) 색상 이외에도 명암이나 패턴으로 콘텐츠 구분이 가능한가?

- 웹 콘텐츠는 색상을 사용할 경우 색상만으로 정보를 제공하지 않도록 한다. 예를 들어 아래 <그림 III - 7(a)>와 같이 파란 버튼과 빨간 버튼을 제공하고 “파란 버튼을 누르시오” 또는 “빨간 버튼

을 누르시오”라고 사용법이 제시된 콘텐츠의 경우, <그림 III - 7(b)>와 같이 흑백화면에서는 색상 구분이 불가능하기 때문이다.



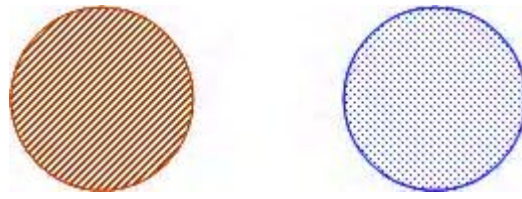
(a) 칼라모니터에 나타난 모양



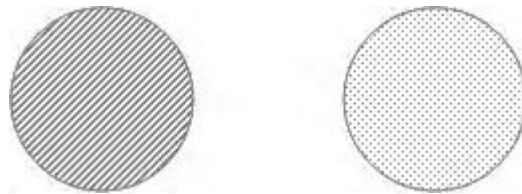
(b) 흑백모니터에 나타난 모양

그림 III - 7. 색상으로만 표시된 버튼

- 색상과 함께 추가적인 정보를 제공하면 이 문제가 해결될 수 있다. 예를 들어 “앞으로 이동하려면 왼쪽의 빨간 버튼을 클릭하시오” 또는 “뒤로 이동하려면 오른쪽의 파란색 버튼을 클릭하시오” 처럼 위치 정보를 제공하면 색각이상자도 접근이 가능하게 된다.
- 더욱 바람직한 방법은 <그림 III - 7(a)>를 아래 <그림 III - 8(a)>와 같이 색상과 함께 선과 점으로 도형을 표시하는 것이다. <그림 III - 8(b)>는 흑백 모니터에서도 충분히 구별이 가능하다.



(a) 칼라모니터에 나타난 모양



(b) 흑백모니터에 나타난 모양

그림 III - 8. 색상으로만 표시된 버튼

(5) 흰 바탕에 밝은 회색 글자처럼 판독이 어려운 색조합을 피하고, 흰 바탕에 검정 글자처럼 대비 차이가 큰 색조합을 사용하고 있는가?

- 웹 콘텐츠의 가독성을 향상시키기 위해서는 전경색과 배경색의 대비를 충분히 차이 나게 하여야 한다. 콘텐츠의 가독성을 평가하는 한 가지 방법은 흑백 화면에서 콘텐츠를 충분히 판독할 수 있는가를 확인하는 것이다. 색상의 대비 차이가 작으면 화면에 보이는 요소를 읽기 어렵다.

(6) 저시력자나 다양한 환경의 사용자도 텍스트를 쉽게 읽을 수 있도록 적절한 크기로 텍스트를 제공하고 있는가?

- 콘텐츠를 만들 때 화면에 표시되는 화면 구성 요소(텍스트, 버튼 등)의 크기를 브라우저에서 조절할 수 있어야 한다. 만일 크기를

조절할 수 없다면 텍스트의 글씨 크기는 12호 이상을 사용하여야 한다.

4.3.2 관련 국가 표준 : 항목 1.3

4.3.3 Flex 프로그래밍 지침 : 항목 3.3

4.4 키보드의 이용

4.4.1 체크리스트

(7) 키보드만으로 콘텐츠가 제공하는 모든 기능에 대한 제어가 가능한가?

- 웹 콘텐츠는 키보드만으로 사용이 가능하여야 한다. 또한 화면 낭독 프로그램과 같은 보조 기술과의 단축키 충돌이 없어야 한다.
- 특히 Tab 키와 Shift + Tab 키에 의한 초점 이동이 원활하여야 한다. 이 과정에서 가능한 한 불필요한 요소로 초점이 이동하지 않도록 함으로써 사용성을 높일 수 있다.

(8) 초점이 주어지는 것만으로 상황이 바뀌지 않는가?

- 화면 낭독 프로그램이 사용되는 경우에 Tab 키와 Shift + Tab 키에 의한 초점 이동시에 초점의 사라짐 등이 발생하지 않아야 한다.

- 초점이 주어지는 것만으로 상황이 바뀌어서는 안 된다. 반대로 활성화 시에는 상황이 바뀌어야 한다.
- 화면 낭독 프로그램을 사용할 때와 사용하지 않을 때 키보드 내비게이션에 차이가 없어야 한다.

(9) 하위 메뉴를 제공하는 주 메뉴의 경우, 키보드만으로도 하위 메뉴를 이용할 수 있는가?

- 주 메뉴에서 하위 메뉴로 이동하는 것은 화면 낭독 프로그램별로 그 방법이 상이하다. 그러나 스크립트로 프로그램을 작성할 때에 이러한 기능이 정상적으로 동작하는지를 확인하여야 한다. 때에 따라서는 Tab 키나 Shift + Tab 키로 이동할 때에 하위메뉴의 마지막에서 예상하지 못한 동작을 할 경우가 있다.

4.4.2 관련 국가 표준 : 항목 2.4

4.4.3 Flex 프로그래밍 지침 : 항목 3.4

4.5 읽는 순서의 설정

4.5.1 체크리스트

(10) 키보드로 폼 컨트롤, 링크, 객체 사이를 이동할 때, 논리적 이동 순서가 적절한가?

- 객체간의 이동은 Tab 키와 Shift + Tab 키를 이용하여 각각 순방향 및 역방향 이동이 가능하다. 이때 이동하는 순서는 화면에 보인 배치 순서와 같거나 읽어주는 순서가 논리적으로 적합하도록 구성하는 것이 바람직하다.
- 화면에 표시되는 콘텐츠의 배치와 이동 순서는 화면 왼쪽 상단에서 화면 오른쪽 아래 방향이라는 전통적인 기준을 따라야 한다. Flex 콘텐츠의 읽는 순서를 제어하는 가장 쉬운 방법은 사용자 인터페이스를 비교적 단순하게 구성하는 것이다. 예를 들어 아코디언(accordion)이나 탭 내비게이터(tab navigator)를 사용한 콘텐츠는 사용자들이 이동하는 순서가 명확하므로 화면 낭독 프로그램이 읽어주는 순서도 명확하다.
- 트리 컨트롤(Tree Control), 데이터 그리드(Data Grid), 텍스트 영역(Text Area)으로 구성된 Flex 콘텐츠의 경우에 화면 낭독 프로그램은 트리 컨트롤, 데이터 그리드, 텍스트 영역의 순서로 콘텐츠를 읽어주는지를 확인한다. 실제로 화면 낭독 프로그램이 타이틀(title), 트리 컨트롤, 데이터 그리드, 텍스트 상자(text box), 그리고 링크(link)의 순서로 읽어준다면 정상이다.
- 예를 들어 아래의 <그림 III - 9>에서 Tab 키에 의한 초점 이동순서는 ‘캔버스화’, ‘부츠’, ‘남성화’, ‘숙녀화’, ‘패딩’, ‘아동화’, ‘목도리’, ‘털모자’, ‘청바지’의 순서이어야 한다.



그림 III - 9. Tab 키에 의한 초점 이동순서

(11) 복잡한 콘텐츠에 대해서는 사용법, 구도, 레이아웃에 관한 정보를 제공하는가?

- Flex 콘텐츠는 데스크톱 애플리케이션 프로그램을 웹에서 실행될 수 있도록 만든 콘텐츠이므로 처음 사용하는 사용자, 특히 화면 낭독 프로그램 사용자는 웹 사이트의 구성, 사용하는 컨트롤, 작업을 종료하는 방법 등에 대해 잘 알지 못할 가능성이 있다. 따라서 개발자는 웹 콘텐츠의 구성과 사용법에 관한 정보를 충실히 제공하여야 한다.
- 사용법이 매우 복잡한 경우에는 별도의 화면을 구성하여 사용법을 알려준다. 즉 버튼이나 링크를 클릭하면 새 창을 열어 콘텐츠의 사용법을 제시하는 것이다. 이 때 사용법을 소개하는 새 창 열기로 이동하는 버튼이나 링크에 대한 설명은 콘텐츠를 다운로드

한 후에 화면 낭독 프로그램이 첫 번째로 읽어주도록 한다. 일반적으로 이러한 목적의 링크나 버튼은 화면에는 보이지 않는 ‘숨긴 링크’ 또는 ‘숨긴 버튼’으로 구현하고, 이를 콘텐츠의 첫 부분에 위치시킨다. 이들 숨긴 요소들은 화면에는 나타나지 않으나 화면 낭독 프로그램으로는 읽어주므로 일반인은 인지할 수 없지만 시각장애인에게는 매우 편리하다.

- 사용법 소개용 새 창 열기 링크나 버튼을 화면의 하단에 명시적으로 배치하여 일반인들도 사용법에 관한 정보를 제공받을 수 있도록 하는 것도 바람직하다.

4.5.2 관련 국가 표준 : 항목 3.2

4.5.3 Flex 프로그래밍 지침 : 항목 3.5

4.6 플러그인 정보 제공

4.6.1 체크리스트

(12) 콘텐츠를 사용하는데 필요한 플러그인은 사용자가 설치하여 사용할 수 있는가?

- Flex 콘텐츠 실행을 위해서는 다양한 플러그인을 필요로 한다. Flex 콘텐츠를 실행하는 과정에서 필요한 플러그인은 자동적으로 다운로드 받아 설치할 수 있어야 한다.

(13) 만일 플러그인의 설치가 불가능하다면 설치 방법을 제시하고 있는가?

- 플러그인의 자동적인 설치가 불가능 하거나 설치하는 과정에서 사용자의 개입이 필요할 경우에는 해당 플러그인을 제공하는 웹 사이트 정보, 플러그인의 설치방법, 사용법 등에 관한 정보를 제공한다.

4.6.2 관련 국가 표준 : 항목 4.1

4.6.3 Flex 프로그래밍 지침 : 항목 3.6

5. 접근성지원 Flex 컴포넌트

제 5절은 접근성을 지원하는 28개의 컴포넌트가 화면 낭독 프로그램과 결합하여 어떤 기능을 제공하는가를 요약, 발췌한 것이다. 여기서 유의할 사항은 이들 기능이 국내에서 사용할 수 있는 화면 낭독 프로그램에는 적용되지 않는 부분이 있다는 점이다.

5.1 Accordion container

- Accordion 컨테이너는 컨테이너 내부에 2개 이상의 패널(또는 탭)이 있으며, 키보드의 방향 키 조작에 따라 초점이 다음 패널로 이동하고, Space bar 또는 Enter 키를 누르면 해당 패널을 선택하게 됨. 초점은 해당 패널의 첫 번째 요소에 위치함.
- Page Up/Page Down 키는 컨테이너의 개별 패널 간을 이동함.

5.2 AdvancedDataGrid control

- AdvancedDataGrid 컨트롤은 도표를 표현하는 DataGrid와 트리구조를 표현하는 Tree 컨트롤의 접근성 기능을 지원함. 트리구조에 관한 접근성 기능은 <III- 5.27. Tree Control>을 참고할 것.

5.3 Alert control

- 서식(Form) 모드에서 Alert 컨트롤 안의 텍스트와 버튼의 레이블을 읽어줌.
- 서식 모드가 아닐 경우, Down 키를 누르면 Alert 컨트롤 안의 텍스트를 두 번 읽어줌.

5.4 Button control

- Space bar 키를 누르면 Button 컨트롤이 활성화 됨. 버튼을 동작시키지 않으려면 Space bar 키를 놓기 전에 Tab 키를 사용하여 초점을 버튼에서 다른 곳으로 이동해야 함.

5.5 CheckBox control

- Space bar 키를 누를 때에 check box 아이템이 활성화/비활성 상태로 바뀜.

5.6 ColorPicker control

- “ColorPicker 콤보 박스”라고 읽어주어야 함.
- Ctrl+Down 키를 누르면 컨트롤이 열리고, Ctrl+Up 키를 누르면 닫힘. ColorPicker 컨트롤이 열려있을 때, 4개의 방향키를 사용하여 color 간을 이동할 수 있음.
- ColorPicker 컨트롤이 열려있을 때, Enter 키를 누르면 Ctrl+Up 키를 누른 것과 같이 현재 선택된 color 값이 설정됨. 또한

ColorPicker 컨트롤이 열려있을 때 Escape 키를 누르면 드롭다운 된 것이 닫히고 color 값이 원래의 color 값으로 돌아가게 됨.

5.7 ComboBox control

- ComboBox는 사용자가 내장된 몇 가지의 텍스트 중에서 하나를 선택할 때 사용하는 컨트롤로 사용에 특별한 어려움이 없음.

5.8 DataGrid control

- DataGrid 컨트롤은 데이터 테이블을 표시하기 위해서 사용하는 컨트롤이며, 방향 키를 누르면 콘텐츠가 강조되고 필드내의 개별 캐릭터 간을 이동함.
- 화면 낭독 프로그램을 서식(Form) 모드에서 사용하면, Tab 키를 누를 때에 DataGrid 컨트롤 내의 편집 가능한 TextInput 필드 간을 이동할 수 있음.

5.9 DateChooser control

- Up, Down, Left, Right 방향키를 누르면 선택된 일자(date)를 변경할 수 있음. Home 키를 누르면 달의 첫 번째 일자(date)로 이동하고 End 키를 누르면 달의 마지막 일자(date)로 이동함.
- Page Up과 Page Down 키를 누르면 각각 이전 달과 다음 달로 이동함.

5.10 DateField control

- Ctrl+Down 키를 누르면 DateChooser 컨트롤을 열어 적절한 일자(date)를 선택할 수 있음. 화면 낭독 프로그램은 이 컨트롤을 선택하면 “드롭다운 캘린더”라고 읽어준 후에 오늘 날짜를 읽어주며, 이어서 “열기는 컨트롤 다운”라고 읽어줌.

5.11 Form container

- Form 컨테이너는 정보를 입력받을 때에 사용할 수 있는 컨트롤임.

5.12 Image control

- Image 컨트롤은 서식 모드가 비활성 상태에서 toolTip에 정의된 내용을 읽어줌. Form 모드에서는 초점을 주지 못하며, 키보드도 마찬가지로.

5.13 Label control

- Label 컨트롤은 다른 컨트롤과 연계되어 있을 때나 서식 모드가 비활성 상태일 때에만 읽어줌. 이 컨트롤은 Form 모드에서는 초점을 주지 못하며 키보드로도 이동할 수 없음.

5.14 LinkButton control

- LinkButton 컨트롤의 읽는 방법은 화면 낭독 프로그램에 따라 다름.

5.15 List control

- 화면 낭독 프로그램을 이용한 내비게이션 방법은 키보드 내비게이션 방법과 동일함.

5.16 Menu control

- 화면 낭독 프로그램을 이용한 내비게이션 방법은 키보드 내비게이션 방법과 동일함.

5.17 MenuBar control

- 화면 낭독 프로그램을 이용한 내비게이션 방법은 키보드 내비게이

선 방법과 동일함.

5.18 Panel container

- Title 속성을 이용하여 각 패널의 이름을 지정할 수 있음. 화면 낭독 프로그램은 Form 모드가 비활성 상태일 때에 패널 Title을 읽어줌.

5.19 RadioButton control/RadioButtonGroup control

- 한 그룹의 라디오 버튼을 선택하고 Enter 키를 누르면 해당 그룹이 선택됨. 방향키를 사용하여 그룹 내의 아이템 간을 이동할 수 있음. Down 키와 Right 키를 누르면 그룹 내의 다음 아이템으로 이동함. Up 키나 Left 키를 누르면 그룹 내의 이전 아이템으로 이동함.

5.20 Slider control

- 서식 모드에서 Slider 컨트롤은 방향 키의 조작에 따라 그 내용을 아래와 같이 읽어줌.
 - 좌-우 slider : Left 또는 Right 키를 이용하여 slider를 낮추거나 높일 수 있음. 또한 Page Up 키와 Page Down 키를 이용하여 slider의 상단 또는 하단으로 직접이동이 가능함.
 - 상-하 slider : Down 키와 Up 키를 이용하여 slider를 낮추거나 높일 수 있음. 또한 Home 키와 End 키를 이용하여 slider의 상단 또는 하단으로 직접이동이 가능함.

5.21 TabNavigator container

- 화면 낭독 프로그램이 TabNavigator를 만나면, 각 폴더를 “탭”으로 읽어줌. 또한 현재 폴더를 “활성”이라고 읽어줌. 폴더가 선택되

면 사용자는 Enter 키를 눌러서 해당 패널로 이동할 수 있음.

- 방향키를 누르면 초점이 다른 패널로 이동하고, 이 때 Space bar 키나 Enter 키를 누르면 해당 패널이 선택됨. Page Up 키나 Page Down 키를 사용하여 컨테이너 내의 패널 간을 이동할 수 있음.

5.22 Text control

- Text 컨트롤은 사용자가 초점을 줄 수 없다. 이 컨트롤은 Form 모드가 비활성 상태일 때에만 화면 낭독 프로그램이 읽어줌.

5.23 TextArea control

- TextArea 컨트롤은 여러 줄의 텍스트를 보여주는 컨트롤임.
- Home 키나 Page Down 키는 줄의 처음으로 이동함. End 키나 Page Up 키를 누르면 줄의 끝으로 이동함.

5.24 TextInput control

- TextInput 컨트롤은 사용자의 입력을 받아들이거나 텍스트를 표현하는 컨트롤임.
- Home 키나 Page Down 키는 줄의 처음으로 이동함. End 키나 Page Up 키를 누르면 줄의 끝으로 이동함.

5.25 TitleWindow container

- 화면 낭독 프로그램은 Form 모드가 비활성 상태일 때에만 TitleWindow 컨트롤을 읽어줌.

5.26 ToolTipManager

- 화면 낭독 프로그램이 실행중일 때, toolTip의 콘텐츠는 초점이 주어진 아이템을 읽은 후에 읽어줌. 접근이 불가능한 컴포넌트에 추

가된 tooltip은 읽어주지 않음.

5.27 Tree control

- Tree 컨트롤은 트리 메뉴를 보여주는 것으로, Up 키와 Down 키를 누르면 Tree 컨트롤 내의 아이템 간을 이동함.
- Right 키나 Space bar 키를 누르면 한 그룹을 열 수 있음. Left 키나 Space bar 키를 누르면 열려있던 그룹을 닫음.

6. Flex 애플리케이션 콘텐츠 구현 예

6.1 접근성을 고려한 입력서식 제작기법

6.1.1 로그인 창 예제

기본적으로 로그인 기능은 ID와 비밀번호를 TextInput 컴포넌트를 통해 입력 받아, 해당 계정의 유효여부를 판단하고, 그 결과에 따라 다음으로 진행하는 동작을 수행한다.

Flex에서 화면을 구성할 경우에 Form 컴포넌트를 이용하면, 보조 기술 사용자에게 필요한 정보를 용이하게 제공할 수 있다. 그러나 디자인 측면에서는 Box 컴포넌트나 Canvas 컴포넌트를 이용하는 것이 좋다.

Box 컴포넌트나 Canvas 컴포넌트를 이용하여 화면을 구성할 경우에 입력창 간에 이동하는 순서를 고려하여야 한다. 즉 입력창에 초점이 주어지면, 입력창의 Title이 보조 기술로 제공되므로 입력창의 배치와 순서가 사용자의 예상에 적합하여야 한다. 입력창의 순서는 tabIndex 속성에 의하

여 결정된다. Tab 키로 초점을 이동시키는 경우에 tabIndex 값이 작은 컴포넌트로 먼저 이동하므로 초점 이동 순서가 정확한 순서를 유지하도록 한다.

<그림 III - 10>의 프로그램은 아이디 입력창, 비밀번호 입력창 및 확인 버튼으로 구성된 로그인 창의 구현 예제이다.

The image shows a simple login form with a title bar that says '로그인'. Inside the window, there are two text input fields. The first is labeled '아이디' (ID) and the second is labeled '비밀번호' (Password). To the right of the password field is a button with the text '확인' (Confirm).

그림 III - 10. 로그인 창의 예제

참고

- 센스/JAWS : 센스리더와 Jaws for Windows는 Tab 키와 Shift + Tab 키에 의한 초점 이동시 콘텐츠를 정상적으로 읽어준다.
- 드림 : Tab 키와 Shift + Tab 키에 의한 초점 이동은 정상적이거나 대체 텍스트를 읽어주지 않으므로 이 부분의 개선이 필요하다.

아래 프로그램에서 보면, 아이디 입력창, 비밀번호 입력창 및 확인 버튼의 tabIndex 값이 각각 1, 2, 3으로 설정되었음을 알 수 있다. 또한 아이디와 비밀번호를 입력한 후에 Enter 키를 누르면 제출되도록 하기 위하여, 비밀번호를 입력하는 TextInput 컴포넌트의 키보드 이벤트를 속성을 enter="submitForm()"으로 설정한다.

O (Good)

```
<mx:Panel title="로그인" height="75%" width="75%" paddingTop="10"
paddingLeft="10" paddingRight="10" paddingBottom="10">
    <mx:Form width="100%" height="100%">
        <mx:HBox width="300" height="300">
            <mx:VBox width="200" height="101">
                <mx:FormItem label="아이디">
                    // Form을 통해 텍스트 입력창의 title을
                    // 바로 읽어들이 수 있도록 구성함
                    // 키보드 이동순서는 tabIndex를 이용함.
                    <mx:TextInput id="idData" width="135" tabIndex="1"/>tabIndex="2" displayAsPassword="true"
                        enter="submitForm()"/>
                </mx:FormItem>
            </mx:VBox>
            <mx:VBox verticalAlign="middle">
                <mx:Button label="확인" height="56" tabIndex="3"
                    click="submitForm()"/>
            </mx:VBox>
        </mx:HBox>
    </mx:Form>
</mx:Panel>
```

6.1.2 회원가입 예제

회원가입 시 각 입력항목에 대한 유효성 검증은 Validator 컴포넌트를 이용하여 수행할 수 있다. 만일 입력창에서 요구하는 값이 정해진 값이 아닐 경우에는 Validator 컴포넌트가 경고창을 이용하여 경고할 수 있다.

Validator 컴포넌트가 제공하는 경고창의 텍스트는 보조 기술로 제공된다.

Form 컴포넌트를 이용하는 경우에 해당 FormHeading 항목을 입력화면의 제목으로 사용하지 않는다. FormHeading을 사용하면 FormItem 컴포넌트 간의 이동 시에 보조 기술을 통하여 FormHeading 이 반복적으로 제공되므로 주의하여야 한다.

<그림 III - 11>에서 기본정보 탭에서 Tab 키를 누르면 초점이 입력창을 순차적으로 이동한다. '다음' 버튼을 스페이스 바로 누르면 다음 탭인 '연령/생년월일' 탭의 첫 항목으로 초점이 이동한다.

그림 III - 11. 회원가입 예제

참고

- 센스/JAWS : 센스리더와 Jaws for Windows는 Tab 키와 Shift + Tab 키에 의한 초점 이동시 콘텐츠를 정상적으로 읽어준다.
- 드림 : 드림보이스는 Tab 키와 Shift + Tab 키에 의한 초점 이동이 정상적이지만 대체 텍스트를 읽어주지 않으므로 이 부분의 개선이 필요하다.

아래의 프로그램은 Accordion 컴포넌트와 Form 컴포넌트를 이용하여 작성한 회원 가입 창 의 구현 예제이다.

O (Good)

```
<mx:Panel title="회원가입" height="75%" width="75%" paddingTop="10"
paddingLeft="10"paddingRight="10" paddingBottom="10">
<mx:Text width="100%" color="blue" text="회원가입에 필요한 정보를 입력
하여 주십시오"/>
<mx:Accordion width="100%" height="100%" id='accrd'
keyUp="dgKey(event)" >
// 첫 번 째 판넬
<mx:VBox label="기본정보">
// Form 컴포넌트를 이용한 항목 구현
<mx:Form width="100%" height="100%">
<mx:FormItem label="성명">
<mx:TextInput id="fname" width="200"/>
</mx:FormItem>
<mx:FormItem label="아이디">
<mx:TextInput id="idData" width="200" keyUp="ddd(event)"/>
</mx:FormItem>
<mx:FormItem label="비밀번호">
<mx:TextInput id="passwdData" displayAsPassword="true"
width="80"/>
```

```

</mx:FormItem>
<mx:FormItem label="비밀번호 확인">
    <mx:TextInput id="passwdData2" displayAsPassword="true"
        width="80"/>
</mx:FormItem>
</mx:Form>
<mx:HBox width="100%" horizontalAlign="right">
<!-- Accordion을 사용하는 경우에 키보드를 이용하여 다음화면으로
이동하도록 다음 패널의 첫 번째 입력창에 초점을 준다. -->

    <mx:Button label="다음" click="{accrd.selectedIndex=1;
        dob.setFocus();}" />

</mx:HBox>
</mx:VBox>
// 두 번째 패널
<mx:VBox label="연령 / 생년월일">
    <mx:Form width="100%" height="100%">
<mx:FormItem label="생년월일 (mm/dd/yyyy)">
    <mx:TextInput id="dob" width="200"/>
</mx:FormItem>
<mx:FormItem label="연령">
    <mx:TextInput id="age" width="200"/>
</mx:FormItem>
</mx:Form>
<mx:HBox width="100%" horizontalAlign="right">
<mx:Button label="이전" click="{accrd.selectedIndex=0;
    fname.setFocus();}" />
<mx:Button label="다음" click="{accrd.selectedIndex=2;
    email.setFocus();}" />

</mx:HBox>
</mx:VBox>
// 세 번째 패널
<mx:VBox label="연락처">
    <mx:Form width="100%" height="100%">

```

```

<mx:FormItem label="E-mail">
    <mx:TextInput id="email" width="200"/>
</mx:FormItem>
<mx:FormItem label="전화번호">
    <mx:TextInput id="phone" width="200"/>
</mx:FormItem>
</mx:Form>
<mx:HBox width="100%" horizontalAlign="right">

    <mx:Button label="이전" click="{accrd.selectedIndex=1;
                                   dob.setFocus();}"/>

</mx:HBox>
</mx:VBox>
</mx:Accordion>
</mx:Panel>

//Validator 컴포넌트를 이용한 텍스트는 경고창과 함께 보여줌.
<mx:StringValidator source="{fname}" property="text" minLength="4"
    maxLength="12"
    tooLongError="정상적인 이름이 아닙니다."/>
<mx:PhoneNumberValidator source="{phone}" property="text"
    invalidCharError="잘못된 연락처 정보입니다.."/>
<mx:DateValidator source="{dob}" property="text" wrongDayError="올바른
날짜형식이 아닙니다."/>
<mx:EmailValidator source="{email}" property="text" missingAtSignError=
"올바른 e-mail형식이 아닙니다."/>
<mx:NumberValidator source="{age}" property="text" minValue="18"
    maxValue="100" domain="int" invalidCharError="숫자만 사용할 수 있습니
다."/>

```

입력해야 하는 정보를 몇 개의 그룹으로 나누고 각 그룹을 Accordion
이나 TabNavigator 컴포넌트의 패널이나 탭으로 구성하는 경우에, 패널

간의 이동이나 탭의 전환이 용이하도록 단축키를 제공하거나 패널 전환 또는 탭 전환용 버튼을 추가한다. <그림 III - 12>에서 '다음' 버튼을 누르면 다음 패널('연령/생년월일')로 이동한다.

6.1.3 게시판 예제

게시판은 RichTextEditor 컨트롤을 이용하여 구성할 수 있다. RichTextEditor 컨트롤은 문장의 정렬, 색, 글자크기, 글씨체들을 조절할 수 있는 기능을 제공한다. RichTextEditor 컨트롤의 속성 중 showToolTip 속성을 true로 설정하면 각 기능키에 toolTip을 제공할 수 있어 접근성을 높일 수 있다.

<그림 III - 12>는 Tab 키에 의하여 입력 상자, 콤보박스, 버튼의 순서로 초점이 이동하는 것을 보여준다.

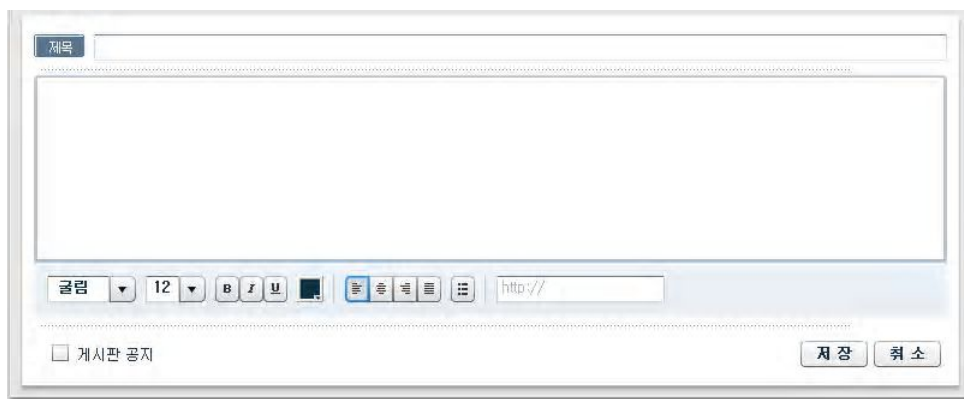


그림 III - 12. 게시판 예제

참고

- **센스** : 센스리더는 Tab 키에 의한 초점 이동 시 대체 텍스트를 정상적으로 읽어준다. 또한 Up/Down 키로 글자체, 글자크기 콤보박스 내 항목 간 이동이 가능하다.
Shift + Tab 키에 의한 초점 이동시에는 글자체, 글자크기 콤보박스 항목을 Up/Down 키로 이동가능 하다. 그러나 대체 텍스트를 정상적으로 읽어주지는 않는다. 이 부분의 개선이 필요하다.
- **드림** : 드림보이스는 Tab 키와 Shift + Tab 키에 의한 초점 이동이 정상적이지만 대체 텍스트를 읽어주지 않는다. 또한 Up/Down 키로 글자체, 글자크기 콤보박스 내 항목 간 이동이 불가능하다. 이 부분의 개선이 필요하다.
- **JAWS** : Jaws for Windows는 Tab 키와 Shift + Tab 키에 의한 초점 이동 시 정상이며 대체 텍스트를 잘 읽어준다.

RichTextEditor 컨트롤 이외의 기능은 toolTip과 Tab 키에 의한 이동순서를 고려하여 화면을 구성한다.

O (Good)

```
<mx:Script>
    <![CDATA[
        private function setRichTextEditor(id:RichTextEditor):void {
            id.fontFamilyArray = ['굴림', '돋음', '바탕', 'Arial'];
            id.textArea.imeMode = "KOREAN";
        }
    ]]>
</mx:Script>

<mx:VBox width="100%" height="100%">
    <mx:Panel title="게시판 " height="75%" width="800" paddingTop="10"
```

```
paddingLeft="10" paddingRight="10" paddingBottom="10">
    <mx:VBox id="center" width="100%" height="100%"
        verticalGap="0">

        <!-- 게시판 제목 -->
        <mx:VBox id="centerStory" width="100%" height="100%"
            verticalAlign="middle" horizontalGap="0" fontWeight="bold"
            verticalGap="0" paddingTop="5">
            <mx:HBox width="100%" height="100%">
            <mx:Image source="{imgTitleName}" toolTip="제목"/>
            <mx:TextInput id="storyContentsTA" width="100%"
                height="100%" paddingLeft="10"
                toolTip="제목입력창입니다."/>
            </mx:HBox>
            <mx:Spacer height="6"/>
        </mx:VBox>
        <!-- 글쓰기 -->
        <mx:VBox id="centerWrite" width="100%" height="100%"
            verticalAlign="middle" horizontalGap="0" verticalGap="0"
            paddingBottom="0" paddingTop="0" >
        <mx:Spacer height="1"/>
        <mx:RichTextEditor id="writeEditor" fontFamily="굴림"
            width="100%" height="100%" headerHeight="0"
            paddingBottom="0" paddingLeft="0" paddingRight="0"
            paddingTop="0" showToolTips="true" borderAlpha="1"
            borderColor="#B0C0D8" headerColors=["#ffffff,
            #ffffff]" footerColors=["#EDF6FA, #E5EEF0]"
            dropShadowEnabled="false" borderThicknessLeft="2"
            borderThicknessRight="2" borderThicknessTop="1"
            borderThicknessBottom="2"
            initialize="setRichTextEditor(writeEditor)"/>
        <mx:Spacer height="7"/>
        <mx:HBox width="100%" horizontalAlign="right"
            paddingBottom="5" paddingLeft="5" paddingRight="5"
            paddingTop="5" horizontalGap="5">
```

```

        <mx:Button label="저 장" click="" styleName="subButton"/>
        <mx:Button label="취 소" click="" styleName="subButton"/>
    </mx:HBox>
</mx:VBox>
</mx:VBox>
</mx:Panel>
</mx:VBox>

```

6.2 접근성을 고려한 차트 제작기법

6.2.1 차트 예제

Chart 컴포넌트는 각종 데이터를 막대 그래프(Bar graph), 파이 차트(Pie chart), 꺾은선 그래프(Line graph)를 묘사할 경우에 사용된다. Flex를 이용하여 차트를 표현하는 것은 매우 간단하다. 이 때 주의해야 할 점은 차트에서 색상을 배제하더라도(예를 들어 흑백 모니터로 화면을 확인할 경우) 충분히 인지할 수 있도록 표현되어야 한다는 것이다.

또한 Flex에서 제공하는 차트의 구체적인 내용은 보조 기술을 통하여 텍스트로 제공될 수 없다. 따라서 차트는 그 내용에 대한 대체 텍스트를 제공하여야 한다. <그림 III - 13>은 차트의 정보를 텍스트로 변환하여 출력하는 예제를 보여준다.



그림 III - 13. 차트 예제

참고

- 센스/JAWS : 센스리더와 Jaws for Windows는 Down 키를 사용하면 차트의 대체 텍스트를 한번에 읽어준다.
- 드림 : 차트에 반응하지 않는다. 이 부분의 개선이 필요하다.

O (Good)

```
<mx:Style>
  Application {
    backgroundColor: #FFFFFF;
    fontFamily: "verdana", "돋움체";
    fontSize: 12;
  }
  Tooltip{
    fontSize: 12;
    fontFamily: "verdana", "돋움";
  }
</mx:Style>
<mx:Script>
  <![CDATA[
    import mx.effects.Zoom;
    import mx.core.Container;
    import mx.collections.ArrayCollection;
    import mx.controls.Alert;
```



```

private var opiAC :ArrayCollection = new ArrayCollection([
    {OPI_NAME: '창의력', SUM_NUM: '7'},
    {OPI_NAME: '어휘력', SUM_NUM: '8'},
    {OPI_NAME: '의지력', SUM_NUM: '10'}]);

[Bindable]
private var accessStr : String = "";
/* chart 정보를 텍스트로 변환하는 부분*/
private function changStr(arrAC:ArrayCollection):void{
    var accStr :String = "";
    for( var i:int = 0; i<arrAC.length; ++i){
        accStr+= arrAC[i].OPI_NAME + ' ' +
        arrAC[i].SUM_NUM + ' 점. ';
    }
    accStr+= '입니다';
    accessStr = accStr;
}
</mx:Script>

<mx:VBox width="100%" height="100%"
creationComplete="changStr(opiAC);">
    <mx:Panel title="차트" height="75%" width="600" paddingTop="10"
paddingLeft="10" paddingRight="10" paddingBottom="10">
        <mx:BarChart id="bbb" color="0x567dbb" styleName="pageBar"
width="98%" height="100%" showDataTips="true" fontSize="11"
gutterLeft="70" fontFamily="굴림" dataProvider="{opiAC}" >
            <mx:verticalAxis>
                <mx:CategoryAxis categoryField="OPI_NAME"/>
            </mx:verticalAxis>
            <mx:horizontalAxis>
                <mx:LinearAxis maximum="24"/>
            </mx:horizontalAxis>
            <mx:series>
                <mx:BarSeries id="xxx" yField="OPI_NAME" xField="SUM_NUM"
displayName="항목" />
            </mx:series>
        </mx:BarChart>
    </mx:Panel>
</mx:VBox>

```

```

</mx:series>
<mx:backgroundElements>
  <mx:Array>
    <mx:GridLines direction="both">
      <mx:verticalStroke>
        <mx:Stroke weight="1" color="#CCCCCC"/>
      </mx:verticalStroke>
    </mx:GridLines>
  </mx:Array>
</mx:backgroundElements>
</mx:BarChart>

<!-- 차트가 화면 낭독 프로그램을 지원하지 않으므로, height 값을
0으로 설정한 상태에서 차트내용을 텍스트로 변환하여 제공함-->
<mx:Label text="{accessStr}" height="0" tooltip="{accessStr}"/>
</mx:Panel>
</mx:VBox>

```

6.3 사용자 UI를 고려한 제작기법

6.3.1 상하 스크롤 바 예제

스크롤은 다량의 정보 중에서 일부를 하나의 화면에 표시하는 방법이다. 아래 그림은 상하 스크롤 바의 예제로, 4줄의 정보만을 화면에 표시하고 있다. 스크롤 창에 보이지 않는 정보를 화면에 나타내려면 Up 키와 Down 키를 이용하거나, PageUp 키와 PageDown 키를 이용하여 스크롤시킨다. 주의할 점은 Flex 콘텐츠에서는 스크롤 바에 초점이 주어지지 않는다는 점이다.



그림 III - 14. 상하 스크롤 바 예제

O (Good)

```
<mx:VBox width="100%" height="100%">
  <mx:Panel title="Scroll" height="75%" width="600" paddingTop="10"
    paddingLeft="10" paddingRight="10" paddingBottom="10">
    <mx:VBox paddingLeft="5" verticalGap="0" width="400"
      height="80" borderStyle="solid" horizontalScrollPolicy="off" >
      <mx:Label text='txt1' />
      <mx:Label text='txt2' />
      <mx:Label text='txt3' />
      <mx:Label text='txt4' />
      <mx:Label text='txt5' />

      <!--실제로 화면에 보이지 않으나, 초점을 가질 수 있기 때문에
        하단의 정보도 조회할 수 있다.-->
      <mx:Button id="btn" width="0" height="0" tabEnabled="true"
        buttonMode="false" />

    </mx:VBox>
  </mx:Panel>
</mx:VBox>
```

6.3.2 롤링 배너 예제

배너에 포함된 모든 이미지는 tooltip, name 또는 description 필드를 이용하여 대체 텍스트를 제공하여야 한다. <그림 III - 15>는 4장의 이미

지로 구성된 롤링 배너이다. Tab 키와 Shift + Tab 키를 이용하여 '이전보기버튼(<)', '이후보기버튼(>)'으로 초점이 이동한다. 각 버튼의 클릭 동작은 Space Bar로 동작한다. 초점이 '이후보기버튼'에 있을 때 Space Bar 키를 누르면 다음 이미지로 전환된다.

화면 낭독 프로그램이 설치되었을 때에도 Tab 키, Shift + Tab 키, Space Bar 키의 기능은 동일하여야 한다.



그림 III - 15. 이미지 롤링 배너 예제

참고

- 센스/JAWS : 센스리더와 Jaws for Windows는 Tab 키와 Shift + Tab 키에 의한 초점 이동이 정상적이며 콘텐츠를 정상적으로 읽어준다. 버튼은 Space Bar와 Enter 키로 동작한다.
- 드림 : Tab 키와 Shift + Tab 키에 의한 초점 이동은 정상적이지만 대체 텍스트를 읽어주지 않는다. 이 부분의 개선이 필요하다. 버튼을 클릭하는 동작은 Space Bar를 누르면 된다.

O (Good)

```
<mx:Style>
    Application {
        backgroundColor: #FFFFFF;
        fontFamily: "verdana", "돋움체";
        fontSize: 12;
    }
    ToolTip{
        fontSize: 12;
        fontFamily: "verdana", "돋움";
    }
    .preBtn {
        upSkin: Embed('/images/pre_over.png');
        downSkin: Embed('/images/pre_down.png');
        overSkin: Embed('/images/pre_over.png');
    }
    .nextBtn {
        upSkin: Embed('/images/next_over.png');
        downSkin: Embed('/images/next_down.png');
        overSkin: Embed('/images/next_over.png');
    }
}
</mx:Style>

<mx:Script>
    <![CDATA[
        /* image 롤링배너를 위한 메서드.*/
        [Bindable]
        private var currentNum :Number = 0;
        [Bindable]
        private var labelStr:String = ""
        [Bindable]
        private var labelDesc:String = ""
    ]]>
</mx:Script>
```

```

/* 다음 배너를 조회하기 위한 function*/
private function nextHandler():void{
if(currentNum+1==imgAC.length){
currentNum=-1;
}

currentNum++;
if(img1.x==0){
img2.source=imgAC[currentNum].SOURCE;
img2.toolTip = imgAC[currentNum].DATA;
labelStr =imgAC[currentNum].DATA;
labelDesc =imgAC[currentNum].DESC;
}else{
img1.source=imgAC[currentNum].SOURCE;
img1.toolTip = imgAC[currentNum].DATA;
labelStr =imgAC[currentNum].DATA;
labelDesc =imgAC[currentNum].DESC;
}

moveRight();
}

/* 이전 배너를 조회하기 위한 function*/
private function preHandler():void{
if(currentNum-1==-1){
currentNum=imgAC.length;
}

currentNum--;
if(img1.x==0){
img2.source=imgAC[currentNum].SOURCE;
img2.toolTip = imgAC[currentNum].DATA;
labelStr =imgAC[currentNum].DATA;
labelDesc =imgAC[currentNum].DESC;
}

```

```

    }else{
        img1.source=imgAC[currentNum].SOURCE;
        img1.toolTip = imgAC[currentNum].DATA;
        labelStr =imgAC[currentNum].DATA;
        labelDesc =imgAC[currentNum].DESC;
    }

    moveLeft();
}

/* 다음 배너를 이동시키는 function*/
private function moveRight():void {

    if(img1.x == 0){
        move1.end();
        move2.end();
        img2.x = 285;
        move1.xTo = -285;
        move2.xTo = 0;
        move1.play();
        move2.play();

    }else{
        img1.x=285;
        move2.xTo = -285;
        move1.xTo = 0;
        move2.play();
        move1.play();
    }
}

/* 이전배너를 이동시키는 function*/
private function moveLeft():void {

```

```

        if(img1.x == 0){
            move1.end();
            move2.end();
            img2.x = -285;
            move1.xTo = 285;
            move2.xTo = 0;
            move1.play();
            move2.play();
        }else{
            img1.x=-285;
            move2.xTo = 285;
            move1.xTo = 0;
            move2.play();
            move1.play();
        }
    }
}

```

]]>

</mx:Script>

```

<mx:VBox width="100%" height="100%" creationComplete="nextHandler()">
    <mx:Panel title="IMAGE 배너" height="75%" width="650"
        creationComplete="nextHandler()" paddingTop="10"
        paddingLeft="10" paddingRight="10" paddingBottom="10" >
        <mx:HBox width="100%">
            <mx:Canvas width="400" height="300">
                <mx:Button x="10" y="30" styleName="preBtn"
                    click='preHandler();' toolTip="다음문항으로 이동합니다."
                    width="17" height="90"/>
                <mx:Canvas x="40" y="5" width="284" height="234"
                    borderStyle="none" horizontalScrollPolicy="off"
                    verticalScrollPolicy="off">
                    <mx:Image x='0' y='0' id='img1' source="images/001.jpg"
                        toolTip="동대산 전경입니다." width="284" height="235"/>
                </mx:Canvas>
            </mx:HBox>
        </mx:Panel>
    </mx:VBox>

```



```

<mx:Image x='285' y='0' id='img2' source="images/002.jpg"
  toolTip="두로봉 전경입니다." width="284" height="235"/>
</mx:Canvas>
<mx:Button x="350" y="30" styleName="nextBtn"
  click='nextHandler();' toolTip="다음문항으로 이동합니다."
  width="17" height="90"/>
</mx:Canvas>

<mx:VBox borderStyle="solid" height="240" width="180"
  horizontalAlign="center">
  <mx:Label id='lv1' text="{labelStr}" fontWeight="bold"/>
  <mx:Text htmlText="{labelDesc}" width="150" />
</mx:VBox>
</mx:HBox>
</mx:Panel>
</mx:VBox>

  <mx:ArrayCollection id='imgAC'>
    <mx:Object SOURCE="images/001.jpg" DATA="동대산 전경입니다."
      DESC='높이 1,434m의 동대산은 북쪽의 두로봉, 동쪽의 노인봉 등
        과 함께 백두대간의 줄기를 이룬다. ... ' />
    <mx:Object SOURCE="images/002.jpg" DATA="두로봉 전경입니다."
      DESC='두로봉은 강원도 평창군 진부면과 홍천군 내면 및 강릉시 연
        곡면 사이에 있는 산으로 높이 1,422m이다. ... ' />
    <mx:Object SOURCE="images/003.jpg" DATA="비로봉 전경입니다."
      DESC='적멸보궁은 부처님의 정골사리를 봉안한 곳이다. ...' />
    <mx:Object SOURCE="images/004.jpg" DATA="상왕봉 전경입니다."
      DESC='백두대간은 험찬 기세로 금강산, 설악산을 지나 대관령,
        태백산, 소백산으로 이어지는데 ... ' />
  </mx:ArrayCollection>
</mx:Application>

```

6.3.3 콘텐츠의 축소/확대 예제

Flex 콘텐츠는 이미지의 확대/축소 기능을 제공한다. 이 기능은 시각을 사용하는 경우에는 유용하지만 전맹인 시각장애인에게는 불필요한 기능이다.

따라서 아래 <그림 III - 16>과 같은 Flex 콘텐츠의 경우에 ‘컨텐츠 확대’ 및 ‘컨텐츠 축소’ 버튼은 Tab 키로 이동이 불가능해야 한다. 그러나 비장애인에겐 이들이 버튼과 같은 역할을 하여야 한다.



그림 III - 16. 콘텐츠의 축소/확대 예제

이 예제를 Flex로 구현할 때 Button 컨트롤을 이용하면 Tab 키에 의하여 초점이 주어지므로 버튼의 Label 속성을 화면 낭독 프로그램이 읽어주게 된다. <그림 III - 16>과 같이 ‘컨텐츠 확대’ 또는 ‘컨텐츠 축소’ 버튼으

로 초점이 이동하지 않게 하려면, image 컨트롤을 이용하여 버튼을 대신 하여야 한다. 또한 이미지 위에 마우스 포인터가 위치할 때에 마우스 포인터가 손바닥 모양으로 변화하도록 하면, 초점이 제공되지 않으나 마우스 사용은 가능하므로 시각장애인과 비장애인 모두가 편리하게 이용할 수 있다.

화면 낭독 프로그램을 실행하더라도 Tab 키나 Shift + Tab 키에 의하여 초점이 버튼으로 이동하지 않아야 한다.

참고

- 센스/JAWS : 센스리더와 Jaws for Windows는 Up/Down 키에 의한 초점 이동시 콘텐츠를 정상적으로 읽어준다.
- 드림 : 드림보이스는 Flex 콘텐츠에 반응하지 않는다.

O (Good)

```
<mx:Script>
  <![CDATA[
    /* 콘텐츠의 확대 또는 축소 기능*/
    private var originWidth:Number=1;
    private var originHeight:Number=1;

    public function contentZoom(flag:String):void{
      var zoom:Zoom = new Zoom();
      zoom.originY=0;

      if (flag == 'zoomIn'){
        originHeight += 0.1;
        originWidth += 0.1;
      } else if (flag == 'zoomOut') {
        originHeight -= 0.1;
        originWidth -= 0.1;
      }
    }
  ]]>
</mx:Script>
```

```

        if (originHeight < 1){ originHeight=1; }
        if (originWidth < 1) { originWidth=1; }
    }

    zoom.zoomHeightTo = originHeight;
    zoom.zoomWidthTo = originWidth;
    zoom.target = Application.application.kado;
    zoom.play();
}

// accessibility 속성을 제거한다.
// (화면 낭독 프로그램에서 읽혀지지 않게 처리)
    private function removeAccessibility(event:FlexEvent):void{
        (event.target.accessibilityProperties
            = new AccessibilityProperties()).silent=true
    }
    ]]>
</mx:Script>

<mx:VBox width="100%" height="100%">
    <mx:Panel title="KADO 위치안내소개">
        <mx:Label text="콘텐츠 축소 / 확대" fontWeight="bold"
            fontSize="14" color="#343434"/>
        <mx:HBox width="650" height="480" id="targetBox"
            verticalScrollPolicy="off" horizontalScrollPolicy="off"
            paddingTop="10" paddingLeft="10">
            <mx:Box width="638" height="463" paddingLeft="0"
                paddingRight="0" paddingBottom="0" paddingTop="0" id="kado"
                borderStyle="solid" horizontalScrollPolicy="off"
                verticalScrollPolicy="off">
                <mx:Image width="330" toolTip="한국정보문화진흥원 약도 - 대표
                    전화 02-3660-2500, 홈페이지담당전화 02-3660-2745 지하철 5호
                    선 발산역에서 강서 보건소 사이 백석초등학교 옆(등촌중학교 건너

```

```
    편) 한국정보문화진흥원" source="map.gif"/>
  </mx:Box>
</mx:HBox>

<mx:HBox width="650" verticalAlign="bottom" horizontalAlign="right">
  <mx:Button label="콘텐츠 확대" click="contentZoom('zoomIn')"
    tabEnabled="false" creationComplete="removeAccessibility(event)"/>
  <mx:Button label="콘텐츠 축소" click="contentZoom('zoomOut')"
    tabEnabled="false" creationComplete="removeAccessibility(event)"/>
</mx:HBox>
</mx:Panel>
</mx:VBox>
```


IV. 접근성 있는 Flash 제작기법

1. Flash 개요

Adobe Flash는 인터넷 애니메이션 저작도구의 하나이다. Adobe Flash로 작성한 웹 콘텐츠는 벡터 이미지를 스크립트로 제어하여 마우스의 움직임에 맞춘 애니메이션과 함께 음향효과를 제공할 수 있으며, 사용자와 인터넷간의 상호작용도 가능하다. 특히 벡터 이미지를 사용하기 때문에 브라우저 창의 크기를 바꾸더라도 화면의 질이 떨어지지 않으며, 화려한 디자인의 웹 콘텐츠를 구현할 수 있는 특징이 있다.

1996년 11월에 들어와 Macromedia사가 FutureWave Software를 인수하고 FutureSplash Animator의 명칭을 Macromedia Flash 1.0으로 바꾸어

출시하였다. 2005년 Adobe System이 인수하기 전까지 기능을 개선하여 버전 9.0에 이르게 되었다. 이 과정에서 Flash는 단순한 인터넷 애니메이션 저작도구를 탈피하여 사용자와 인터넷이 상호작용이 가능한 웹 애플리케이션을 개발할 수 있는 웹 애플리케이션 개발도로 진화하였다.

2005년 4월에 Macromedia사를 인수한 Adobe System은 Flash를 리치 미디어(rich media) 웹 콘텐츠와 이러닝(e-learning) 콘텐츠를 생성할 수 있는 도구로 계승 발전시켰다. 2007년 4월부터는 Flash를 Adobe Creative Suite 3계열에 편입하여 Adobe Flash라는 상품명으로 발매하였다. 2008년 11월 현재, Flash 최신버전은 Adobe Flash CS4 Professional이다.

Flash 콘텐츠를 재생하는 프로그램에서는 Flash Player가 사용된다. 이 프로그램은 MS Windows, Mac OS X, Linux 등 대부분의 운영체제하에서 동작이 가능하며, Internet Explorer, Firefox 등 대표적인 웹 브라우저의 플러그인(plug-in)으로 동작한다. 근래에는 휴대폰에서도 Flash player가 탑재되어 사용되기도 한다.

Flash 콘텐츠의 파일 포맷은 공개되어 있어 누구나 Flash 데이터를 가공, 생성할 수 있으며, 심지어 저작도구를 개발하여 배포할 수도 있다. 이러한 정책으로 인하여 Flash는 인터넷 콘텐츠에 가장 많이 사용되는 표준의 하나로 자리매김을 하고 있다.

일부에서는 HTML 기반의 웹 콘텐츠에 비하여 Flash 콘텐츠의 다운로드에 시간이 걸리기 때문에 Flash 사용을 좋아하지 않는 경우도 있지만, Flash 콘텐츠를 사용하지 않는 웹 사이트는 찾아보기 힘들 정도로 Flash 기술이 광범위 하게 사용되고 있다.

Flash가 이렇게 널리 알려지게 된 것은, Flash가 기존의 웹 콘텐츠 제작

방식과는 다른 획기적인 기능을 갖고 있기 때문이다. 우선 Flash는 벡터 기반의 드로잉 방식을 지원해 적은 용량으로 보다 많은 내용을 담을 수 있기 때문에 기존의 방식에 비하여 콘텐츠를 다운받는 시간이 크게 단축된 점과, 사용 방법이 간단해서 조작하기 쉽고, 작업 흐름이 간편해 빠른 시간 안에 원하는 결과물을 인터넷에서 구현할 수 있는 점이 대표적인 기능이다. 사용자의 입장에서 보면 간단히 Flash Player를 다운받아 설치하면 Flash 콘텐츠가 실행될 수 있다. 뿐만 아니라 화려한 애니메이션과 함께 고급스런 음향 효과를 함께 제공할 수 있다는 점도 Flash가 웹 애니메이션 제작도구로서 폭발적인 인기를 누리게 된 비결이다.

그러나 Flash 콘텐츠는 정적인 웹 페이지를 화려하고 생동감 있게 변모시키는 대신, 장애를 가진 사람들이나 멀티미디어 환경에 적응하지 못하는 계층에게는 새로운 어려움을 주게 되었다. 예를 들면 Flash Player로 재생하려는 Flash 콘텐츠가 비디오와 자막으로 구성되어 있다고 하더라도 자막을 시각장애인에게 제공할 수 없다면 Flash 콘텐츠는 시각장애인에게는 접근이 불가능하다.

우리나라의 경우 2008년 전반기까지는 Flash 콘텐츠에 접근할 수 있는 기술적 통로가 마련되어 있지 않아 장애인들이 접근하기 매우 어렵거나 불가능한 콘텐츠로 인식되어져 왔다. 그러나 이미 Flash는 MSAA (Microsoft Active Accessibility)를 지원하고 있으므로 접근성을 지원하는 Flash 콘텐츠는 MSAA를 지원하는 보조기기를 사용하는 장애인들의 접근이 가능하다. 다행히 국내의 일부 화면 낭독 프로그램 개발자들도 Adobe 사의 지원을 받아 Flash 콘텐츠에 접근할 수 있는 길이 마련되었다.

IV장에서는 Adobe Flash를 이용하여 Flash 콘텐츠를 제작함에 있어서 접근성을 제공하는 기법을 기술한다. 따라서 이 문서에서 제시한 방법에

따라 Flash 콘텐츠를 제작하면 적절한 보조기기를 사용하는 장애인들에게 필요한 최소한의 접근성을 보장할 수 있게 된다.

그러나 이 문서는 접근성 제공을 위한 충분한 정보를 제공하지는 못한다. 그 이유는 접근성 있는 Flash 콘텐츠를 제작하기 위해서는 접근성 있게 콘텐츠를 제작하는 기법을 알아야 할 뿐 아니라, 부가적으로 제공해야 하는 정보가 충실해야 하기 때문이다. 예를 들어 Flash 콘텐츠에 포함된 이미지에 대한 보조적인 설명이 해당 이미지를 충실하게 설명하지 못한다면, 기술적인 방법이 아무리 뛰어나다고 하더라도 설명의 불충분으로 인하여 발생하는 접근성의 저하문제는 어떤 방법으로도 해결될 수 없기 때문이다. 이 문서에서는 Flash 콘텐츠의 접근성을 제공하는데 필요한 기술적인 방법을 중심으로 기술한다.

2. 웹 접근성과 Flash 콘텐츠

이 문서에서 대상으로 삼는 Flash 콘텐츠 제작 도구는 Adobe Flash CS4 Professional(이하 'Flash'라고 한다)을 대상으로 한다. 만일 Adobe Flash CS4 Professional이 아닌 다른 버전을 언급해야 할 경우에는 Flash라는 명칭이 아니라 온전한 명칭을 사용할 것이다.

Flash 콘텐츠와 같은 리치 미디어를 사용하기 위해서는 Abode Flash Player가 필요하다. Abode Flash Player(이하 'Flash Player'이라 한다)는 Flash Player 6.0 버전부터 장애인에게 접근성이 제공되는 리치 미디어 콘텐츠를 사용할 수 있도록 하는 접근성 지원 기능을 지원하였다. Flash Player를 사용하면 시각장애인도 화면 낭독 프로그램을 이용하여 접근성이 있는 Flash 콘텐츠에 접근할 수 있다.

우리나라의 경우에 Flash Player와 함께 사용할 수 있는 충분한 기능을 지닌 화면 낭독 프로그램은 아직까지 개발되지 않았다. 그러나 일부 화면 낭독 프로그램이 부분적으로 Flash 콘텐츠에 대한 접근성을 제공하고 있으므로 점차 기능이 향상될 것으로 기대한다.

Flash Player는 미국 Microsoft가 제공하는 MSAA를 지원하므로 Flash 콘텐츠 개발자들이 접근성을 제공하기 위하여 조치를 취한 부분은 화면 낭독 프로그램으로 전달된다. 따라서 개발자들이 Flash 콘텐츠를 개발할 때에 접근성이 제공된다면, 화면 낭독 프로그램이 이를 판독하여 접근성 정보를 읽어주게 된다.

2.1 접근 가능한 Flash 콘텐츠 구현

접근 가능한 Flash 콘텐츠를 개발하기 위해서는 Flash 콘텐츠를 사용할 장애인에 대한 폭넓은 이해가 필요하다. 그러나 대부분의 개발자들은 이러한 경험이나 보조 기술에 대한 이해가 부족하기 때문에 접근성 있는 Flash 콘텐츠를 개발하는 것이 용이하지 않다. 이 문서에서는 적은 노력으로도 최소한의 접근성을 지원하는 Flash 콘텐츠를 만들 수 있는 방법을 소개하여 개발자들이 실무에 적용할 수 있도록 할 것이다.

접근 가능한 Flash 콘텐츠를 개발하기 위해서는 액세스 가능성 패널²⁴⁾ 또는 ActionScript를 이용하여야 한다. 액세스 가능성 패널은 아래 <그림 IV - 1>과 같이 Flash 무비를 구성하는 요소에 대한 접근성 지원 기능을 설정하고 대체 텍스트(text equivalent)를 제공하는데 이용된다. Flash 빌더에서 액세스 가능성 패널을 Flash 전면에 나타나게 하려면 Shift + F11 키를 누른다.

24) Flash CS4 영문버전에서는 Accessibility Panel, 즉 접근성 패널이라고 한다.

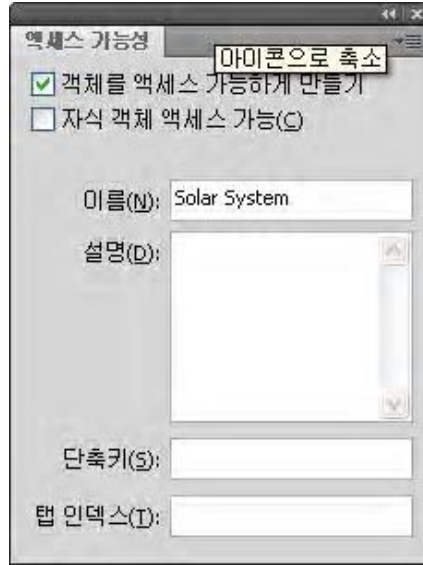


그림 IV - 1. '액세스 가능성' 패널

액세스 가능성 패널에는 객체를 액세스 가능하게 만들기²⁵⁾, 자식 객체 액세스 가능²⁶⁾, 자동 레이블 등 2~3개의 체크박스와, 대체 텍스트를 입력하는 필드인 이름 필드와 설명 필드로 구성된다.

액세스 가능성 패널의 객체를 액세스 가능하게 만들기 체크박스는 해당 객체를 접근성 있게 만들 것인가를 설정하는 체크박스이다. 따라서 해당 객체가 접근성을 지원해야 하는 경우에는 키보드에 의한 접근이 가능하며, 대체 텍스트를 제공하는 경우에는 이 옵션을 체크(설정)해야 한다.

자식 객체 액세스 가능 옵션은 선택된 객체의 자식 객체들에게도 별도로 접근성을 지원할 것인가를 설정하는 옵션이다. 만일 전체 무비에 대하여 하나의 객체로 인지하도록 하고, 하나의 대체 텍스트를 일률적으로 제

25) Flash CS4 영문버전에서는 'Make Object Accessible', 즉 '객체를 접근성 있게 만들기'의 의미이다.

26) Flash CS4 영문버전에서는 'Make child objects accessible', 즉 '자식 객체를 접근성 있게 만들기'의 의미이다.

공하고자 하는 경우에는 이 옵션을 언체크(uncheck)하여야 한다.

자동 레이블 옵션은 화면 낭독 프로그램이 객체에 제공된 레이블을 자동적으로 읽어줄 것인가를 결정한다. 예를 들어 버튼의 레이블이 '확인'이라고 하면 자동 레이블 옵션이 설정되어 있을 경우에 화면 낭독 프로그램은 "확인 버튼"이라고 읽어준다. 만일 자동 레이블 옵션이 설정되어 있지 않으면 "버튼"이라고 읽어준다.

접근성 패널을 이용하여 정보를 제공하는 절차는 ActionScript를 이용하여 수행할 수도 있다. ActionScript에 관한 자세한 설명은 Adobe 자료²⁷⁾를 참고하라.

2.2 Flash 콘텐츠의 임베딩

Flash를 HTML 콘텐츠에 포함시키려면 object 태그와 embed 태그를 이용한다. W3C에서는 object 태그를 사용하여 Flash 콘텐츠를 HTML에 임베드 시킬 것을 권고하고 있다. 그러나 대부분의 브라우저는 embed 태그를 이용하여 사운드나 동영상 등의 멀티미디어 요소를 HTML 문서에 포함하는 것을 허용하고 있다.

Flash 콘텐츠를 HTML 문서에 임베딩 시킬 경우에 param 태그의 속성을 window로 설정하지 않으면 Flash 콘텐츠에 보조 기술이 접근할 수 없다. 그 이유는 Flash 콘텐츠에 param 태그의 wmode 속성을 transparent 또는 opaque 로 설정하면 Flash 콘텐츠가 HTML 콘텐츠의 후면에 위치하는 것으로 간주하기 때문에 보조 기술 자체가 Flash 콘텐츠의 존재를 인식하지 못하기 때문이다. 따라서 보조 기술 사용자에게 중요한 정보를

27)

http://help.adobe.com/ko_KR/Flash/10.0_UsingFlash/WSd60f23110762d6b883b18f10cb1fe1af6-7c2ba.html

제공하는 Flash 콘텐츠는 반드시 window 모드로 설정하여야 한다. 이 때, wmode 속성을 지정하지 않으면 기본 값이 window 모드로 자동 설정된다.

브라우저에 따라 미세한 차이가 존재한다. 예를 들면 FireFox, Opera, Safari에서는 object 태그와 param 태그를 이용하는 경우에 Flash 콘텐츠를 표시하지 못하는 경우가 발생한다. 또한 Internet Explorer에서는 embed 태그와 param 태그를 함께 설정하지 않으면 JavaScript 함수 호출은 가능하나 함수의 리턴 값을 사용할 수 없는 경우가 발생한다.

Flash 콘텐츠를 HTML 문서에 임베딩 시킬 때 설정하는 param 태그와 embed 태그의 wmode 속성의 의미는 아래 <표 IV - 1>과 같다. Flash 콘텐츠의 임베딩 시에 설정하는 param 태그의 속성에 따른 웹 콘텐츠의 차이에 관한 자세한 사항은 Stephanie Sullivan의 블로그²⁸⁾를 참조하라.

표 IV - 1. embed 및 param 속성의 의미

| param 속성 | 배 치 | 투명 여부 | 접근성 제공여부 |
|-------------|--------------------------------|----------|-------------|
| window | Flash 파일이 항상 전면에 위치 | 불투명-기본 값 | ○ |
| transparent | 레이어(DIV) 등이 Flash 파일 전면에 위치 가능 | 투명 | × |
| opaque | 레이어(DIV) 등이 Flash 파일 전면에 위치 가능 | 불투명 | × |

28) Stephanie Sullivan의 블로그 :
<http://www.communitymx.com/content/article.cfm?cid=E5141>

3. Flash 접근성 지원 프로그래밍 지침

제 3절에서는 Flash 콘텐츠를 포함하는 웹 콘텐츠가 접근성을 제공하기 위하여 Flash 콘텐츠가 지켜야 할 코딩 방법을 제시한다. 제시하는 프로그래밍 지침의 나열 순서는 우리나라 국가 표준인 ‘인터넷 웹 콘텐츠 접근성 지침(Internet Web Contents Accessibility Guideline; KICS.OT-10.0003; 2005. 12. 21)’의 순서에 따르기로 한다. 이 지침에서는 인터넷 웹 콘텐츠 접근성 지침을 ‘국가표준’이라고 한다.

3.1 (대체 텍스트 제공) 텍스트가 아닌 Flash 콘텐츠 요소는 대체 텍스트를 제공하여야 한다.

국가표준 <항목 1.1>에 따르면, ‘텍스트가 아닌 콘텐츠’(이미지, 멀티미디어, 애니메이션 등)는 적절한 대체 텍스트를 제공하여야 화면 낭독 프로그램과 같은 보조 기술을 이용하는 장애인에게 그 내용을 전달할 수 있다. 이는 중요한 내용을 포함하고 있는 배경 이미지의 경우에도 마찬가지이다.

3.1.1 요구조건

(1) Flash 콘텐츠를 구성하는 요소는 다음과 같이 대체 텍스트를 제공한다.

- 그래픽 요소에는 대체 텍스트를 제공하여야 한다.
- 그래픽 아이콘(icon)에는 대체 텍스트를 제공하여야 한다.
- 페이지의 특정 영역을 강조하는 애니메이션의 움직임에 대해서는 그 내용에 대한 대체 텍스트를 제공하여야 한다.

- 텍스트 분리(Break Apart) 기능을 사용할 경우에는 대체 텍스트를 제공하여야 한다.

3.1.2 적용방법

Flash를 사용하는 개발자는 ActionScript 또는 액세스 가능성 패널을 이용하여 대체 텍스트를 제공할 수 있다. 또한 스테이지(stage)에 추가한 요소에 제공한 대체 텍스트는 Flash player와 화면 낭독 프로그램을 이용하여 읽어줄 수 있다. 따라서 대체 텍스트를 제공하는 것은 어려운 일이 아니다. 더 중요한 점은 ‘언제, 어떤 내용의 대체 텍스트를 제공할 것인가’를 결정하는 일이다.

가) 이미지와 화면 낭독 프로그램

아래 <그림 IV - 2>는 스테이지에서 ‘깨끗한 물’이라는 정적 텍스트(static text)와 플라스틱 이미지를 구성한 Flash 화면 모습이다. 완성된 콘텐츠를 Flash Player로 실행하면 “깨끗한 물”이라고 읽어준다.

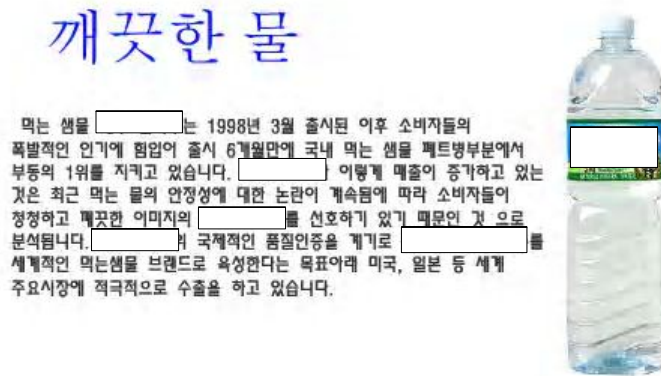


그림 IV - 2. 스테이지(stage)에 정적 텍스트와 이미지를 추가한 예제

위의 <그림 IV - 2>에서 스테이지의 오른쪽에 놓인 플라스틱 물병 이미지는 텍스트가 아니기 때문에 Flash Player로 읽어주지 못한다. 물병 이미지는 전체 화면에서 매우 상징적인 내용을 담고 있으므로, 시각장애인에게 이미지에 대한 정보를 제공할 수 있는 수단으로 화면 낭독 프로그램이 이미지를 읽어줄 수 있도록 대체 텍스트를 제공하여야 한다.

대체 텍스트를 제공하기 위해서는 액세스 가능성 패널을 사용한다. 우선 스테이지에서 이미지를 선택한 후, 객체를 라이브러리(Library)에 심볼(symbol)로 저장한다. Flash에서는 그래픽 심볼의 경우에 대체 텍스트를 제공할 수 없으므로 그래픽 심볼을 무비 심볼이나 버튼으로 저장해야 한다.

아래 <그림 IV - 3>은 액세스 가능성 패널의 모습으로 대체 텍스트를 보조 기술로 제공하도록 객체를 액세스 가능하게 만들기 체크박스가 선택된 모습이다.

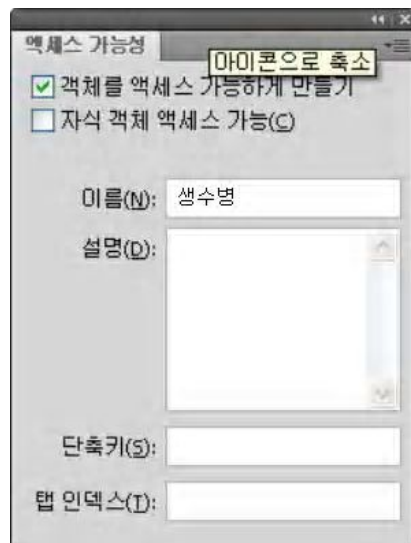


그림 IV - 3. 이름 필드에 대체 텍스트를 제공하는 예제

액세스 가능성 패널에서 이름 필드와 설명 필드는 각각 대체 텍스트를 제공하는 용도로 사용된다. 이름 필드는 비교적 간단한 대체 텍스트를 제공하는데 사용되며, 설명 필드는 대체 텍스트가 비교적 긴 경우에 사용한다. 이름 필드와 설명 필드는 각각 HTML의 'alt' 속성과 'longdesc' 속성과 동일한 것으로 보면 된다. 화면 낭독 프로그램은 두 필드의 내용을 구분하지 않고 이름 필드, 설명 필드의 순서로 읽어준다. 따라서 두 필드를 같은 내용으로 채우면 두 번 읽어주므로 주의할 필요가 있다.

짧은 대체 텍스트는 이름 필드를 사용한다. 설명 필드는 한글 25자 이상인 대체 텍스트에 사용하는 것이 좋다.

나) 무비와 대체 텍스트

Flash를 이용하여 콘텐츠를 생성하는 과정에서 텍스트, 입력 텍스트 필드, 버튼, 무비클립(movie clip) 뿐 아니라 무비 전체를 접근성 있게 만들 수 있다. 여기서 텍스트 요소는 접근성을 지원하기 위한 추가적인 처리가 필요하지 않으나, 나머지 요소들의 경우에는 액세스 가능성 패널을 이용하여 대체 텍스트를 제공하여야 한다.

아래 <그림 IV - 4>는 달이 지구를 공전하는 모습의 무비이다. 전체 무비에 대한 대체 텍스트를 제공하기 위해서는 무비에서 스테이지를 선택하고 이름 필드에 '지구를 공전하는 달'이라고 대체 텍스트를 입력한다. 그리고 아래 <그림 IV - 5>와 같이 액세스 가능성 패널에서 자식 객체 액세스 가능 체크박스를 언체크 한다.

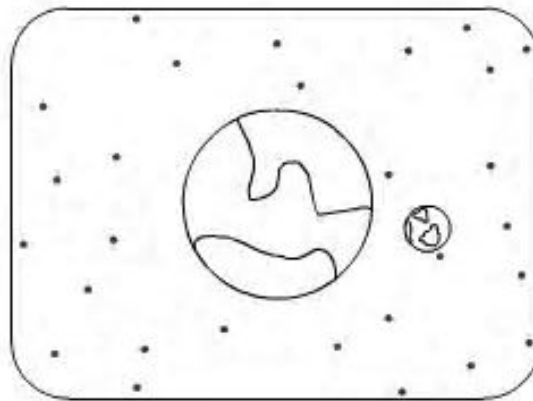


그림 IV - 4. 무비의 예제(한국 Adobe 제공)

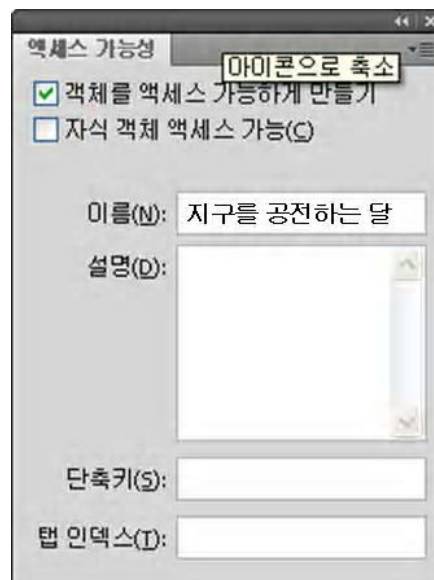


그림 IV - 5. 전체 무비에 하나의 대체 텍스트를 제공하는 방법

다) 올바른 대체 텍스트

‘텍스트가 아닌 콘텐츠’에 제공되는 대체 텍스트는 화면에 표시되는 이미지를 대신해서 보조 기술(예를 들어 화면 낭독 프로그램)을 통하여 제공되는 정보이므로 그 내용이 적절하여야 한다. ‘텍스트가 아닌 콘텐츠’ 요소에 대체 텍스트를 제공하는 것은 어려운 일이 아니다. 가장 중요한 점은 ‘언제, 어떤 내용의 대체 텍스트를 제공할 것인가’를 고려하여야 한다는 것이다. 또한 대체 텍스트로 인하여 화면 낭독 프로그램 사용자에게 어떠한 영향을 주게 될 것인가를 생각하는 일이다.

<그림 IV - 2>의 예에서는 투명한 플라스틱 병에 생수가 가득 들어있는 이미지에 대한 대체 텍스트를 ‘투명한 액체가 담긴 플라스틱 병’이라고 하기보다는 ‘생수병’이라고 하는 것이 더 적합하다. 마찬가지로 <그림 IV - 4>에 보인 무비에서도 대체 텍스트는 달과 지구에 대하여 개별적으로 묘사하기보다는 달과 지구의 관계를 나타내는 ‘지구를 공전하는 달’이라는 대체 텍스트가 적절하다(<그림 IV - 5> 참조).

3.2 (자막 제공 방법) 오디오가 있는 영상물은 오디오와 동기되는 대체 텍스트를 제공하여야 한다.

국가 표준 <항목 1.2>에 의하면, 비디오나 오디오 등의 멀티미디어 콘텐츠에는 화면해설을 자막(caption)과 같은 시각적인 수단으로 제공하여야 한다.

3.2.1 요구조건

- (1) 비디오와 오디오는 대체 매체로 자막을 제공하여야 한다.

- (2) 자막 파일은 W3C가 규정한 Timed Text XML(DFXP) 형식이나 FLV 큐 포인트 형식이어야 한다.

3.2.2 적용방법

가) 자막 컴포넌트의 사용

Flash는 오디오와 비디오 콘텐츠의 저작도구로 광범위하게 이용된다. 그 이유는 개발자들이 FLV 또는 H.264 규격의 비디오에 간단히 자막을 추가할 수 있도록 Flash 컴포넌트를 제공하기 때문이다. Flash에서 FLVPlaybackCaptioning 컴포넌트를 이용하면 FLVPlayback 컴포넌트에 자막을 추가할 수 있다. FLVPlaybackCaptioning 컴포넌트를 이용하여 FLVPlayback 컴포넌트에 추가할 수 있는 자막의 구조는 W3C의 DFXP(Timed Text XML) 파일이다. Adobe에서도 Flash 콘텐츠용 자막으로 DFXP 형식을 권장하고 있다. DFXP 형식의 자막을 생성하는 소프트웨어는 여러 종류가 있으며, 일부 회사에서는 인터넷을 통하여 자막 변환 서비스를 제공하기도 한다.

FLVPlaybackCaptioning 컴포넌트는 여러 개의 FLVPlayback 컴포넌트를 이용할 수 있다. 가장 간단한 방법은 다음과 같다. 우선, FLVPlayback 컴포넌트를 스테이지로 끌어다 놓고, FLVPlaybackCaptioning 컴포넌트를 동일한 스테이지에 끌어다 놓는다. 이어서 자막파일 URL을 지정한 후에 showCaptions를 '참'으로 설정하면 자막이 자동으로 연결된다.



그림 IV - 6. 자막파일을 지정하는 방법

위 <그림 IV - 6>은 FLVPlaybackCaptioning 컴포넌트의 설정 방법을 보여주고 있다. 좌측의 구성요소 관리자의 source 파라미터에 자막파일 이름을 입력한 상태이다. 그 밖의 파라미터를 설정하여 FLVPlayback 컴포넌트를 이용한 자막처리가 가능하도록 한다. Flash에 의하여 만들어진 SWF 파일은 아래 <그림 IV - 7>과 같이 비디오에 자막이 포함된 파일을 생성한다.



그림 IV - 7. 자막파일을 제공한 예제

화면상에서 자막이 표시되는 위치를 바꾸어야 할 필요가 있을 경우에는 FLVPlaybackCaptioning 컴포넌트에서 자막의 위치를 선택할 수 있다. 뿐

만 아니라 폰트, 전체화면보기를 선택하였을 경우에 영상과 폰트의 자동 확대 기능 등도 설정할 수 있다. 아래 <그림 IV - 8>은 자막 폰트를 가독성이 좋은 폰트로 변경한 화면 모습이다.



그림 IV - 8. 가독성이 좋은 폰트로 변경한 모습

FLVPlaybackCaptioning 컴포넌트는 unicode를 지원하므로 한글 뿐 아니라 심볼, 음악기호 등도 자막으로 제공할 수 있다.

나) FLV 큐 포인트의 사용

FLVPlaybackCaptioning 컴포넌트는 내장된 큐포인트(embedded cue points)에 포함된 자막을 화면에 보여준다. FLV 큐포인트(cue points)의 장점은 자막 데이터가 FLV 파일에 포함되어 있으므로 트래킹할 파일 수가 적어진다는 점이다. Captionate은 FLV 파일에 큐 포인트를 추가할 수 있는 자막 생성 프로그램이다.

3.3 (색상과 접근성) 무비에 사용되는 색상은 중요한 정보 제공 수단으로 사용하지 않는다.

국가표준 <항목 1.3>에 따르면, 색상으로 표현된 정보는 색상을 배제하여도 원하는 내용을 전달할 수 있어야 한다. 따라서 웹 콘텐츠는 화면을 흑백으로 바꾸더라도 명암이나 패턴을 통해 콘텐츠가 구분하여 표시하고자 하는 내용을 사용자가 인지할 수 있어야 한다.

3.3.1 요구조건

- (1) Flash 콘텐츠는 색상을 이용하여 정보를 제공하지 않아야 한다.
- (2) Flash 콘텐츠는 배경색과 전경색 간의 대비가 분명해야 한다.
- (3) Flash 콘텐츠의 텍스트, 이미지 텍스트, 버튼 텍스트 등은 가독성이 충분하여야 한다.

3.3.2 적용방법

가) 색상

Flash 콘텐츠에서 색상을 선택할 때에는 색각이상자와 저시력자를 고려하여 선택하여야 한다. 이것은 색상만으로 정보를 제공하지 않아야 함을 의미한다. 예를 들어 파란 버튼과 빨간 버튼을 제공하고 “파란 버튼을 누르시오” 또는 “빨간 버튼을 누르시오” 라고 사용법이 제시된 콘텐츠의 경우 흑백화면 사용자, 또는 색각이상자는 구분할 수 없다.

색상과 함께 추가적인 정보를 제공하면 이 문제가 해결될 수 있다. 예를 들어 “앞으로 이동하려면 오른쪽의 빨간 버튼을 클릭하십시오” 또는 “뒤로 이동하려면 왼쪽의 파란 버튼을 클릭하십시오”와 같이 위치 정보를 제공하면, 색각이상자도 접근이 가능하게 된다.

나) 배경색과 전경색

색상과 관련된 두 번째 문제는 가독성을 향상시키기 위해서는 전경색과 배경색 간의 대비가 충분히 커야 한다. 콘텐츠의 가독성을 평가하는 한 가지 방법은 콘텐츠를 흑백 화면에 표시하였을 경우에도 콘텐츠의 판독이 가능한가로 평가할 수 있다. 전경색이 배경색과 비교하여 대비가 낮으면 화면에 보이는 요소를 판독하기 어렵다.

다) 글자크기

Flash 콘텐츠에 사용하는 텍스트, 버튼, 링크 등의 크기는 충분한 가독성을 제공하도록 크기를 조절할 수 있어야 한다. 만일 글자크기를 조절할 수 없을 경우에는 텍스트 크기를 12호 이상으로 설정하여 가독성을 높인다.

3.4 (깜박거리는 객체 사용 제한) 콘텐츠는 스크린의 깜박거림을 피할 수 있도록 구성되어야 한다.

국가 표준 <항목 2.3>에 따르면, 잦은 깜박거림이 있는 콘텐츠는 광과민성 증세를 가진 장애인이 사용할 경우 발작을 일으키는 원인이 되므로 피해야 한다. Flash 무비는 그 특성상 화면의 급격한 전환이 가능한 콘텐츠를 제공할 수 있으므로 광과민성 환자에게 치명적인 위해가 가해지지

않도록 해야 한다.

3.4.1 요구조건

- (1) 웹 콘텐츠에는 애니메이션 등과 같이 깜빡거리는 주파수의 범위가 3 Hz에서 49 Hz 사이인 콘텐츠 요소들을 포함하지 않아야 한다.
- (2) 만일 위에서 명시한 요구조건을 만족할 수 없는 웹 콘텐츠는 깜빡거림이 있는 웹 페이지로 이동하기 전에 이 페이지에 깜빡거림이 있음을 사전에 사용자에게 경고해주어야 한다.

3.4.2 적용 방법

광과민성 발작 증세를 지닌 사람들은 빛이 깜빡거리는 것에 반응하여 발작을 일으킨다. 특히 3Hz에서 49Hz 사이의 깜빡거림은 발작을 가장 잘 일으키는 주파수 범위이다. 따라서 콘텐츠를 개발할 때에 깜박거림의 주파수가 요구조건을 만족하여야 한다.

가) 홈 페이지의 깜박거림 배제

웹사이트에 처음 접속했을 때에 화면에 나타나는 홈 페이지에는 최소한 깜빡거림이 없어야 한다. 만일 다음 페이지부터 깜빡거림이 있는 콘텐츠가 포함되어 있다면, 홈 페이지에 이러한 내용을 표시하여 사용자가 다음 페이지로 이동할 것인가를 선택할 수 있도록 해야 한다.

나) Flash 콘텐츠의 크기를 작게 만들

Flash 콘텐츠를 목적상 깜박거리게 만들어야 하는 경우에는 임베드

되는 Flash 콘텐츠의 크기를 가로와 세로 모두 200 픽셀 이하가 되도록 한다.

3.5 (키보드의 이용) 마우스로 할 수 있는 모든 컨트롤은 키보드로도 제어가 가능해야 한다.

국가표준 <항목 2.4>에 따르면, 웹 콘텐츠는 키보드 또는 장애를 극복하도록 도와주는 여러 가지 입력 장치를 사용하는 경우에도 웹 콘텐츠가 제공하는 모든 기능을 사용할 수 있어야 한다. 예를 들어 마우스를 사용할 수 없는 장애인들도 마우스를 사용할 수 있는 사용자와 같이 키보드만으로 웹 콘텐츠가 제공하는 모든 기능을 동일하게 수행할 수 있어야 한다. 이 검사항목은 브라우저의 기능 뿐 아니라 웹 콘텐츠가 제공하는 기능에도 해당된다.

3.5.1 요구조건

- (1) 무비에 버튼과 다른 컨트롤을 추가하는 경우에 키보드만으로 무비를 사용할 수 있어야 한다.
- (2) 키보드 접근성을 제공하기 위하여 프레임 내에서 스크립트를 사용하고 스크립트를 직접 객체에 연결하지 않는다.
- (3) 빈 무비클립을 버튼으로 사용하지 않는다. 빈 무비클립으로 구성된 '히트영역(hit area)'은 화면 낭독 프로그램으로 판독이 불가능하다.
- (4) 버튼에는 단축키를 제공하여 접근성을 높인다.

3.5.2 적용방법

가) Flash Player와 화면 낭독 프로그램의 사용

Flash Player는 마우스를 사용하는 이벤트들을 키보드로도 접근할 수 있다. 이 요구조건을 검사하기 위하여 키보드와 화면 낭독 프로그램을 동시에 이용하거나 때로는 화면 낭독 프로그램을 사용하지 않고 키보드만으로 조작할 수 있는가를 검사한다.

나) 투명한 히트 영역을 사용하지 않을 것

Flash 콘텐츠에는 보이지 않는 버튼클립(empty button clip)을 사용하지 않는다. 보이지 않는 버튼클립이란 어떤 모양을 가진 일정한 영역을 히트 상태가 될 수 있는 히트 영역(hit area)으로 정의한 것이다. 이 방법은 단일 라이브러리를 가지는 객체들을 텍스트 객체들 위에 위치시키고 스크립트만 바꾸면서 반복해서 재사용할 수 있는 장점이 있다.

투명한 히트 영역은 버튼클립이 'up' 상태에 있을 때에 화면 낭독 프로그램이 버튼에 관한 콘텐츠를 읽어주지 못하므로 사용자에게 버튼이 없다고 판단하게 하는 문제가 있다. 이 문제는 Flash 콘텐츠 자체로는 해결하기 어렵지만 화면 낭독 프로그램의 기능을 보완하면 해결이 가능하다. 화면 낭독 프로그램은 투명한(transparent) 무비클립이 'up' 상태이면 이를 버튼으로 간주하여 사용자에게 사용할 수 있도록 알려주기만 하면 된다. 화면 낭독 프로그램이 이 기능을 지원하는지를 확인하려면 해당 화면 낭독 프로그램 개발자에게 문의하라.

다) 복잡한 애플리케이션 콘텐츠의 단축키 설정

다수의 컨트롤이 필요한 복잡한 애플리케이션 콘텐츠는 단축키를 이용하여 사용하도록 하는 것이 좋다. 지체장애인들은 키보드를 사용하는데 어려움이 있으므로 누르는 키의 수를 줄일 수 있는 단축키를 제공하는 것이 바람직하다.

복잡한 애플리케이션 콘텐츠의 단축키를 생성하기 위해서는 listener 이벤트를 정의하고, listener 이벤트에 대응하는 스크립트를 작성하여야 한다. 아래의 ActionScript 2.0 스크립트는 웹 페이지를 오픈하는 경우를 보인 것이다.

```
on (click) {  
    gotoURL(http://www.kado.or.kr/index.html);  
}
```

이 스크립트는 버튼처럼 사용되는 무비클립의 인스턴스와 연결되어 있다. 이것을 프레임(예를 들어 무비의 첫 번째 프레임)에 연결하도록 ActionScript 3.0 버전으로 작성한 스크립트는 다음과 같다. 아래 스크립트는 ActionScript의 MouseEvent 메소드를 사용하고 있으며, 이 메소드는 키보드로도 운영이 가능하다.

O (Good)

```
function gotoKadoSite(event:MouseEvent):void  
{  
    var adobeURL:URLRequest=new URLRequest(  
        "http://www.kado.or.kr/");  
    navigateToURL(adobeURL);  
}
```

```
}  
home_mc.addEventListener(MouseEvent.CLICK, gotoKadoSite);
```

3.6 (접근성 지원 애니메이션) 화면의 잦은 갱신을 유발시키는 콘텐츠의 경우에는 보조 기술에 영향을 주지 않도록 구현하여야 한다.

국가표준 <항목 2.6>에 따르면, 화면의 잦은 갱신을 유발시키는 Flash 콘텐츠는 화면 낭독 프로그램으로 하여금 끊임없이 콘텐츠를 읽어주도록 하므로 화면 낭독 프로그램 사용자의 접근성을 떨어뜨린다. 따라서 Flash 콘텐츠는 화면의 갱신에도 불구하고 보조 기술에 영향을 주지 않아야 한다.

또한 국가표준 <항목 2.3>에 따르면, Flash 콘텐츠와 같은 애니메이션이나 동영상의 경우에는 초기상태가 정지된 상태이어야 한다.

3.6.1 요구조건

- (1) 무비의 움직임을 웹 페이지의 갱신으로 처리하지 않도록 해야 한다.
- (2) 무비는 초기 상태가 정지된 상태이어야 한다.

3.6.2 적용방법

가) 애니메이션과 화면 낭독 프로그램

웹 사이트가 제공하는 Flash 콘텐츠를 다운로드하면, 화면 낭독 프로그램은 무비의 상태(예를 들어 웹 콘텐츠의 다운로드 상태 등)를 알려주어야 한다. 콘텐츠를 다운 받은 후에는 웹 콘텐츠를 페이지의 처음부터 마지막까지 읽어준다.

Flash 콘텐츠의 특징은 콘텐츠가 시간에 따라 변화한다는 점이다. 그런데 Flash Player는 콘텐츠에 변화가 발생하면, 화면 낭독 프로그램에게 콘텐츠에 변화가 있음을 알려준다. 화면 낭독 프로그램은 이러한 통보를 받으면 자동적으로 웹 페이지의 첫 부분부터 읽기를 시작한다. 따라서 이러한 특성을 고려하지 않고 만들어진 Flash 콘텐츠는 심각한 문제를 야기하게 된다.

예를 들어 몇 개의 프레임을 반복해서 보여주도록 만든 어떤 광고용 배너를 Flash Player로 화면에 표시하는 경우에 이 배너는 화면의 변화를 계속적으로 유발시켜 화면 낭독 프로그램은 반복적으로 웹 페이지의 처음부터 읽어주려고 한다. 화면 낭독 프로그램 사용자들은 이러한 문제에 자주 봉착하게 되므로, 이를 방지하기 위하여 화면 낭독 프로그램은 Flash 이벤트를 중지시키는 단축키를 제공할 필요가 있다. 마찬가지로 화면 낭독 프로그램의 해제 단축키를 다시 누르면 Flash 이벤트가 원래 상태로 돌아가야 한다. 화면 낭독 프로그램이 이 기능을 제공하는 지 여부는 해당 화면 낭독 프로그램 개발사로 문의하라.

나) 전체 무비의 접근성 제공

애니메이션 무비는 애니메이션 전체에 하나의 대체 텍스트를 제공하는 것이 일반적인 방법이다. 그렇지 않으면 도리어 접근성을 떨어뜨리는

경우가 종종 발생한다.

전체 무비에 대한 대체 텍스트는 <IV- 3.1 나) 무비와 대체 텍스트>에 기술한 바와 같이 개별 요소들에 대한 사항보다는 요소들 간의 관계를 묘사하는 것이 바람직하다.

전체 무비에 대한 대체 텍스트를 제공하기 위해서는 액세스 가능성 패널에서 자식 객체 액세스 가능 체크박스를 언체크하면, 그룹에 포함된 모든 요소들을 하나로 간주하게 되므로 전체 무비에 대하여 하나의 대체 텍스트를 제공한다. 이렇게 하면 무비의 변화를 화면 낭독 프로그램으로 통보하지 않기 때문에 화면 낭독 프로그램이 콘텐츠를 처음부터 반복해서 읽어주는 현상을 해소할 수 있다. 앞에 보인 <그림 IV - 4>의 ‘지구를 공전하는 달’이 그 예이다.

다) 애니메이션의 정지

애니메이션의 접근성을 높이는 것은 화면을 정지시키는 것과 밀접한 관계가 있다. 일반적으로 무비는 화면전환이나 내려받기 중에는 움직임이 포함되는 것이 자연스러우나, 다운로드가 완료된 후에는 화면을 정지시키는 것이 중요하다. 학습장애자에게 있어서 화면의 움직임은 사용자의 관심을 빼앗아 화면상의 다른 요소들을 파악하지 못하도록 한다.

3.7 (비디오/오디오 컨트롤) 무비에 포함된 비디오 및 오디오는 보조 기술 사용자에게 컨트롤을 제공하여야 한다.

국가표준 <항목 2.4> 및 <항목 4.1>에 따르면, 마우스를 사용할 수 없는 장애인들도 마우스를 사용할 수 있는 사용자와 같이 키보드만으로 웹 콘텐츠가 제공하는 모든 기능을 동일하게 수행할 수 있어야 한다.

3.7.1 요구조건

- (1) 비디오 또는 오디오 재생 컨트롤은 시각장애인, 저시력자 및 키보드만 사용할 수 있는 사용자들이 이용할 수 있도록 구비되어야 한다.
- (2) Flash가 제공하는 접근성 있는 비디오 또는 오디오 재생장치 Skin을 이용하여 Flash 콘텐츠를 개발하여야 한다.
- (3) 무비는 화면 낭독 프로그램을 사용하여 콘텐츠를 청취하는데 방해를 주지 않도록 오디오를 제공하지 않아야 한다. 만일 무비의 특성상 오디오를 제공해야 하는 경우에는 사용자가 오디오의 재생/일시정지를 제어할 수 있어야 한다.
- (4) 만일 무비의 특성상 배경음악과 같은 오디오를 제공해야 하는 경우에는 사용자가 볼륨(Volume)을 조절하여 재생중인 오디오의 음량을 줄일 수 있어야 한다.
- (5) 콘텐츠에 포함된 모든 컨트롤은 사용될 때마다 그 결과를 보조 기술로 알려주어야 한다.

3.7.2 적용방법

Flash는 접근성 있는 비디오를 생성할 수 있도록 도와준다. 비디오와 함께 제공되는 자막은 청각장애인, 시각장애인, 저시력자 및 지체장애인에게 접근성을 제공한다. Flash가 제공하는 FLVPlayback 컴포넌트는 비디오 콘텐츠를 재생하는데 필요한 접근성 기능을 자동적으로 지원한다.

웹 콘텐츠를 다운받는 도중에 음악이나 오디오가 재생된다면 화면 낭독 프로그램 사용자는 매우 큰 어려움에 빠질 수 있다. 무비의 오디오는 화면 낭독 프로그램 사용자들이 무비 콘텐츠의 내용을 청취하는 과정을 방해한다. 따라서 사용자는 음악 또는 오디오의 재생여부를 컨트롤할 수 있어야 한다. 가장 간단한 방법이 사용자로 하여금 오디오를 재생시키거나 일시정지할 수 있도록 하는 것이다. 이는 사용자가 오디오를 다룰 수 있게 함으로써 어려움을 주지 않고 오디오에 대한 경험을 가질 수 있도록 하기 위한 것이다.

가) Flash 스킨을 이용한 구현

Flash CS4가 제공하는 모든 스킨은 키보드와 화면 낭독 프로그램을 지원하므로 개발자는 간단히 FLVPlayback 컴포넌트를 스테이지에 끌어다 놓기만 하면 생성된 비디오 콘텐츠는 접근성이 제공되는 재생관련 컨트롤을 이용할 수 있게 된다.

나) 컨트롤 단축키

비디오 컨트롤은 키보드 단축키로도 이용이 가능하다. Play/Pause(재생 및 일시정지), Stop(정지), Rewind(되감기), Mute(음소거), 자막(Closed

Caption)과 같은 버튼은 Tab 키를 이용하여 선택이 가능하며, 스페이스바로 활성화 시킬 수 있다.

볼륨(Volume)이나 재생위치(Playhead position) 등과 같은 슬라이드 컨트롤(Slide control)은 화살표 키(Arrow key)로 조절 가능하며, Home 과 End 키는 슬라이드 컨트롤의 처음과 마지막으로 직접 이동하는 역할을 한다. 숫자 키는 볼륨의 오디오 레벨을 직접 선택하는데 사용된다.

단축키를 이용하여 오디오 재생 컨트롤을 사용할 수 있다. 예를 들어 Play/Pause(재생/일시정지)용 단축키는 'p'를 할당하여, 토글되도록 할 수 있다. 즉, 'p'를 한번 누르면 재생(Play)되고, 한번 더 누르면 일시정지(Pause)로 전환되며, 다시 한번 'p'를 누르면 재생(Play)으로 바뀌는 것이다.

음소거(Mute) 단축키로는 'm'이나 숫자 '0' 을 사용한다. 음소거 단축키를 누르더라도 재생중인 오디오는 멈추지 않고, 소리를 완전히 죽여서 들리지 않도록 한다. 따라서 음소거를 하면 오디오는 들리지 않게 되나 화면 낭독 프로그램은 콘텐츠를 읽어주므로 시각장애인에게 매우 편리하게 된다.

볼륨(Volume)은 화면 낭독 프로그램이 동작하고 있는 동안 재생중인 오디오의 음량만을 줄이는 경우에 사용된다. 따라서 볼륨 컨트롤은 음악 스트리밍의 경우에 오디오가 사용자의 주의를 산만하게 하지 않도록 할 경우에 필요하다.

Flash Player는 Flash 콘텐츠를 재생하기에 앞서 보조 기술이 동작하고 있는지를 검사할 수 있다. 따라서 Flash 콘텐츠를 제작할 때에 보조 기술이 설치되어 동작 중일 경우에는 오디오 컨트롤을 음소거(Mute)로 설정하

거나 볼륨을 매우 작게 조절하도록 함으로써 화면 낭독 프로그램 사용자의 불편함을 줄일 수 있다.

다) 단축키 설정

화면 낭독 프로그램별로 비디오 재생 컨트롤에 할당된 단축키가 다를 수 있다. 그 이유는 화면 낭독 프로그램이 일부 키를 이미 다른 용도로 할당하고 있기 때문이다. 그렇다고 하더라도 컨트롤의 종류는 위에서 열거한 방법과 크게 차이가 나지 않을 것이다. 화면 낭독 프로그램의 비디오 및 오디오 재생 컨트롤 단축키에 관한 정보는 해당 화면 낭독 프로그램과 관련한 웹 사이트를 참고하라.

3.8 (읽는 순서의 설정) Flash 무비를 읽는 순서는 논리적이어야 한다.

국가표준 <항목 3.1>에 따르면, 웹 콘텐츠는 화면에 표시되는 구성요소의 나열 순서와 보조 기술을 통하여 전달되는 정보의 순서가 일치하여야 한다. 만일 이 순서가 일치하지 않거나 논리적으로 상이할 경우에는, 웹 콘텐츠를 화면을 통하여 인지하는 사람과 보조 기술을 통하여 인지하는 사람 간에 차이가 발생한다.

또한 국가표준 <항목 2.4>에 따르면, 키보드로 Flash 콘텐츠 구성요소 간을 이동할 때, 논리적 이동 순서가 화면에 나타난 순서와 일치하여야 한다.

3.8.1 요구조건

- (1) 무비의 화면구성 요소를 읽어주는 순서는 논리적이어야 한다.

3.8.2 적용방법

Flash로 생성된 무비를 화면 낭독 프로그램을 이용하였을 때에 읽는 순서는 접근성 설계에 있어서 매우 중요한 일이다. Flash 무비의 요소들을 읽는 순서는 화면 왼쪽 상단에서 화면 오른쪽 아래 방향이라는 전통적인 기준을 따르지 않는다. 따라서 순서가 뒤바뀐 내용을 화면 낭독 프로그램을 통하여 읽고 이를 이해하기는 매우 어렵다.

예를 들어 아래 <그림 IV - 9(a)>의 Flash 콘텐츠는 3줄로 버튼을 늘어놓아 키보드를 형상화 한 것이다. 언뜻 보기에는 알파벳 순서에 따라 초점이 버튼으로 이동할 것이라고 생각하기 쉬우나, 실제로는 <그림 IV - 9(b)>의 번호 순서대로 초점이 이동한다.



(a) 버튼의 구성



(b) 버튼의 초점 이동 순서

그림 IV - 9. 콘텐츠의 초점 이동 순서에 관한 예제(한국 Adobe 제공)

따라서 화면에 보이는 위치와 초점이 이동하는 순서를 일치시키기 위해서는 초점의 이동 순서를 결정하는 과정이 필요하다.

초점이 화면 구성요소 간을 이동하는 순서를 제어하는 방법은 아래와 같다.

- ActionScript를 이용하여 초점이 이동하는 순서를 정한다.
- 콘텐츠 오프스테이지를 이용하여 초점이 이동하는 순서를 정한다.

가) ActionScript의 이용방법

초점이 구성요소 간을 이동하는 순서를 정교하게 제어할 수 있는 방법으로 ActionScript의 .tabindex 프로퍼티를 사용하는 것이다.

ActionScript에서 초점이 이동하는 순서와 Tab 키에 의한 이동 순서는 동일하다. 그러나 ActionScript를 이용하여 무비를 이동하는 순서를 제어하고자 할 때에는 무비 내의 텍스트 필드 및 화면 표시를 위한 요소들을 포함한 모든 인스턴스(instance)들은 .tabindex 목록에 포함되어야 한다.

초점이 이동하는 순서를 제어하기 위하여 스테이지의 모든 인스턴스는 인스턴스 네임(instance name)을 가지고 있어야 한다. 인스턴스 네임을 가져야 하는 인스턴스는 텍스트, 무비클립, 버튼 심볼 뿐 아니라 무비가 실행되는 동안 사용되는 모든 컴포넌트를 포함한다.

정적 텍스트(static text)에는 인스턴트 네임을 부여할 수 없다. 따라서 정적 텍스트 인스턴스는 이동하는 순서를 초기 설정으로 돌아가게 한다.

ActionScript를 이용하여 이동하는 순서를 제어하기 위해서는 동적 텍스트(dynamic text) 필드를 이용해야 한다. 동적 텍스트를 사용하면 전체 파일 크기가 커지는 단점이 있다.

.tabindex 목록은 무비가 실행되는 동안 사용되는 모든 인스턴스를 포함해야 한다. 여기에는 화면에 나타나지 않는 요소, 오프스테이지(offstage)에 있는 요소, 다른 인스턴스에 가려진 모든 요소가 포함된다. 만일 어떤 요소가 화면 낭독 프로그램 사용자에게 불필요할 경우에는 이 요소의 visible 프로퍼티를 '거짓'으로 설정하거나 .silent 프로퍼티를 '참'으로 설정한다. 뿐만 아니라 무비가 시작할 때에는 보이지 않으나 나중에 나타나는 요소들도 .tabindex 목록에 포함되어야 한다.

일련의 자식 SWF 파일을 부모 무비에 로딩하는 경우, .tabindex 값을 자식 무비클립에 수록하여야 한다. 그러나 목록에서 각각의 자식 SWF 파일로 초점이 이동하는 순서는 서로 달라야 한다. 예를 들어 두 개의 자식

무비가 부모 무비에서 사용되고, 각 무비가 .tabindex 값이 각각 1, 2, 3 인 3개의 요소를 가지고 있다고 하면, 화면 낭독 프로그램은 첫 번째 로드한 무비의 첫 번째 값을 읽어주며, 두 번째 로드한 무비의 첫 번째 값을 읽어준다. 이어서 화면 낭독 프로그램은 첫 번째 무비클립의 두 번째 값을 읽어주며, 이어서 두 번째 무비클립의 두 번째 값을 읽어준다. 따라서 화면 낭독 프로그램이 첫 번째 무비를 로드하여 콘텐츠를 읽어주고, 이어서 두 번째 무비의 콘텐츠를 읽어줄 경우, 첫 번째 무비의 .tabindex 값이 1, 2, 3이라면, 두 번째 무비의 .tabindex 값은 앞의 수치와 다른 값(예를 들면 4, 5, 6)으로 설정하여야 한다. 이 값은 순차적일 필요는 없으나 서로 달라야 한다.

나) 오프스테이지의 이용방법

동적인 Flash 콘텐츠의 경우에 읽는 순서를 사전에 결정하는 것은 매우 어렵다. 혼란 경우는 아니지만 동적 Flash 콘텐츠는 오프스테이지에 단일 컬럼의 콘텐츠를 여벌로 구성하여 해결할 수 있다. 온스테이지(onstage) 콘텐츠는 화면 낭독 프로그램이 읽어주지 못하게 접근할 수 없도록 한다. 여벌의 콘텐츠는 컬럼으로 구성하여 읽는 순서에 관계가 없는 요소 간을 이동하지 않도록 해야 한다.

이 방법은 두 가지 문제점이 있다. 첫째는 무비 내에 포함되는 객체의 수를 증가시켜 파일 크기를 키우고 무비의 성능을 떨어뜨린다는 점이다. 두 번째는 화면 확대 프로그램 사용자에게 심각한 혼란을 야기 시킬 수 있다는 점이다. 화면 확대 프로그램은 스테이지의 요소들을 확대할 뿐 아니라 이들을 화면의 중앙에 보이도록 한다. 그런데 온스테이지 콘텐츠가 접근할 수 없도록 되어 있기 때문에 오프스테이지의 콘텐츠에 초점이 주어져 화면 확대 프로그램 사용자로 하여금 혼란을 야기하게 한다. 따라서 이 방법은 화면 확대 프로그램을 사용할 경우에 적합하지 않다. 그러나

화면 낭독 프로그램 사용자에게는 매우 유용한 방법이다.

Flash 콘텐츠는 화면 낭독 프로그램을 사용하고 있는가를 자동으로 파악할 수 있다. 따라서 화면 낭독 프로그램을 사용하고 있는 경우에만 오프스테이지를 이용한 읽는 순서 제어 방법을 사용하도록 할 수 있다. 그러나 화면 낭독 프로그램과 화면 확대 프로그램을 동시에 사용하는 경우에는 달리 대책이 없다.

화면 낭독 프로그램이 사용되는지를 자동으로 확인하기 위해서 Flash가 지원하는 MSAA를 이용한다. Accessibility.isActive() 메서드는 화면 낭독 프로그램이 실행 중이고, Flash 콘텐츠에 초점이 제공되었을 때에 결과치가 '참'이 된다. 중요한 점은 이 메서드가 무비의 실행초기에는 호출결과를 '거짓'으로 응답하기도 하므로 Flash 무비의 버튼을 누를 때 호출되도록 ActionScript로 작성하는 것이 바람직하다.

Accessibility.Active()의 리턴 값이 '참'이면 화면에 나타난 콘텐츠는 접근이 불가능하다. 이렇게 만드는 가장 쉬운 방법은 모든 화면상의 콘텐츠를 하나의 무비클립으로 구성하고, .silent 프로퍼티를 '참'으로 만드는 것이다. .silent 프로퍼티를 사용하는 것이 무비클립의 표시여부(visibility)를 변경하는 것보다 좋다. 그 이유는 화면 낭독 프로그램 사용자와 온스테이지 콘텐츠를 볼 수 있어야 하는 일반 사용자들이 공동으로 작업하는 경우도 있기 때문이다.

화면에 표시되지 않는 콘텐츠는 단일 컬럼으로 로드된다. 이 컬럼은 무비의 높이와 일치할 필요는 없다. 중요한 점은 온스테이지의 변화에 대응하여 오프스테이지의 콘텐츠도 갱신되어야 한다는 점이다.

3.9 (구조의 제공) 무비의 레이아웃, 구도, 사용법 등은 대체 텍스트를 제공해야 한다.

국가표준 <항목 3.1>에 따르면, 웹 콘텐츠는 보조 기술 사용자가 사용 방법을 알 수 있도록 논리적으로 구성하여야 한다. 또한 국가표준 <항목 3.3>에 따르면, 온라인 서식에서는 서식 제어 요소와 레이블이 대응되어야 하며, 레이블이 서식 제어 요소에 선행하여야 한다.

국가표준 <항목 1.1>에 따르면, 웹 콘텐츠에서 온라인 서식 작성은 매우 중요한 구성요소 중의 하나이다. 특히 온라인 서식을 작성하기 위해서는 입력할 내용, 입력하는 방법, 절차 등을 대체 텍스트나 레이블로 제시하여야 한다.

3.9.1 요구조건

- (1) 레이아웃, 구도, 내비게이션이 복잡한 Flash 콘텐츠는 그 사용법이나 구도 등에 관한 정보를 대체 텍스트로 제공한다.

3.9.2 적용방법

레이아웃, 구도, 내비게이션이 복잡한 Flash 콘텐츠는 화면 낭독 프로그램 사용자가 사용하기 어렵다. 따라서 무비의 레이아웃이나 구도, 사용법 등에 전체 무비를 대상으로 하나의 대체 텍스트를 제공한다. 대체 텍스트를 제공하기 위해서 root-level description을 사용하거나 별도의 정보 제공화면(information screen)을 이용한다. 대체 텍스트를 제공하는 방법은 <IV- 3.1 대체 텍스트 제공>을 참고하라.

3.10 (대체 페이지 제공) Flash 콘텐츠에 대한 접근성을 지원할 수 없을 경우에는 Flash 콘텐츠가 포함되지 않은 대체 웹 페이지를 제공한다.

국가표준 <항목 4.2>에 따르면, Flash 콘텐츠가 접근성을 지원할 수 없을 경우에는 Flash 콘텐츠가 포함된 웹 페이지 전체에 대한 대체페이지를 제공하여야 보조 기술 사용자의 접근성을 제공하게 된다.

또한 국가표준 <항목 1.1>에 따르면, 그 내용이 중요하지 않기 때문에 보조 기술 사용자에게 감추어도 무방한 Flash 콘텐츠는 콘텐츠에 관한 정보를 보조 기술로 제공하지 않는다.

3.10.1 요구조건

- (1) 접근이 불가능한 Flash 콘텐츠는 대체 수단을 제공하여야 한다.
- (2) 내용이 중요하지 않은 Flash 콘텐츠는 보조 기술 사용자에게 보이지 않도록 처리한다.

3.10.2 적용방법

가) 대체 수단의 제공

접근성을 지원하지 못하는 Flash 콘텐츠를 이용하여 웹 콘텐츠를 구성하면 이로 인하여 전체 페이지의 접근성이 떨어진다. 이 경우에 Flash 콘텐츠가 포함된 웹 콘텐츠를 대신하여 별도의 대체 페이지를 제공할 수 있다.

아래의 스크립트는 화면 낭독 프로그램의 동작 여부를 검사하여 화면 낭독 프로그램이 동작하지 않을 경우에는 원래의 웹 페이지를 다운로드하여 화면에 표시하지만, 화면 낭독 프로그램이 동작 중일 경우에는 대체 페이지를 다운로드하여 화면에 표시하는 스크립트이다.

```
if (Accessibility.isActive()) {  
    getURL(screenreaderpage.htm);  
} else {  
    getURL(normalpage.htm);  
}
```

위의 스크립트에서 screenreaderpage.htm 와 normalpage.htm 는 각각 화면 낭독 프로그램이 동작 중인 경우와 동작하지 않을 경우에 로드할 웹 페이지를 의미한다.

나) Flash 콘텐츠의 무시

광고 배너, 화면 치장용 Flash 콘텐츠는 그 내용을 보조 기술 사용자에게 전달할 필요성이 낮은 콘텐츠이다. 그러나 이들 Flash 콘텐츠는 자체적으로 화면을 갱신하도록 설계한 경우가 많기 때문에 콘텐츠에 변화가 발생하면, 화면 낭독 프로그램에게 콘텐츠에 변화가 있음을 알려주어 자동적으로 웹 페이지의 첫 부분부터 읽도록 한다. 이러한 문제는 나머지 웹 콘텐츠에 대한 접근을 차단한다. 따라서 그 내용이 중요하지 않은 Flash 콘텐츠는 화면 낭독 프로그램이 판독하지 않도록 감추는 것이 바람직하다.

Flash 콘텐츠에 화면 낭독 프로그램이 접근하는 것을 감추기 위해서는

아래 스크립트와 같이 Flash 무비를 포함하고 있는 웹 페이지의 wmode 옵션과 embed 태그를 opaque로 설정한다.

```
<object ...>  
  <param name="wmode" value="opaque">  
  <embed wmode="opaque" ...> ... </embed>  
</object>
```

위의 HTML 코드는 Flash 무비를 임베딩하면서 화면 낭독 프로그램으로부터 감추는 부분이다. wmode 속성을 opaque 또는 transparent 모드로 설정하면 비록 Flash 무비가 화면에 나타난다고 하더라도 화면 낭독 프로그램으로 인식하지 못하므로 보조 기술 사용자에게는 이들 Flash 무비가 없는 것으로 간주된다. 따라서 이 기법은 보조 기술 사용자에게 의미가 없는 Flash 콘텐츠에 적용되어야 한다. 반대로 wmode 옵션을 생략하거나 windows로 설정한 Flash 콘텐츠는 화면 낭독 프로그램이 항상 그 내용을 보조 기술로 전송하여 읽어준다. 이와 관련한 자세한 설명은 <IV- 2.2 Flash 콘텐츠의 임베딩>을 참고하라.

3.11 (플러그인 정보 제공) Flash 콘텐츠가 포함된 웹 페이지는 Flash Player를 다운로드 할 수 있는 웹 사이트에 대한 정보를 제공하여야 한다.

국가표준 <항목 4.1>에 따르면, 웹 콘텐츠는 웹 페이지에 포함된 콘텐츠를 구성하는데 필요한 모든 스크립트, 플러그인 등에 대한 정보와 링크를 제공하여야 한다. 또한 사용자가 이를 설치할 수 있어야 한다.

3.11.1 요구조건

- (1) Flash 콘텐츠가 포함된 웹 페이지에는 Flash Player 최신 버전을 다운로드 할 수 있는 웹 사이트에 대한 링크가 제공되어야 한다.
- (2) 콘텐츠를 사용하기 위해서 필요한 플러그인은 사용자가 설치하여 실행할 수 있어야 한다. 만일 플러그인을 사용자가 설치할 수 없다면 플러그인의 설치방법 등을 제공하여야 한다.

3.11.2 적용방법

Flash 콘텐츠는 실행을 위해서는 다양한 Flash Player와 여러 가지 플러그인을 필요로 한다. Flash 콘텐츠를 실행하는 과정에서 필요한 플러그인은 자동적으로 다운로드 받아 설치할 수 있어야 한다. 만일 자동적인 설치가 불가능 하거나 설치하는 과정에서 사용자의 개입이 필요할 경우에는 해당 플러그인을 제공하는 웹 사이트 정보, 플러그인의 설치방법, 사용법 등에 관한 정보를 제공하여야 한다.

3.12 (접근성 지원 컴포넌트의 사용) Flash 콘텐츠 개발시에 사용하는 컴포넌트는 Flash CS4가 제공하는 접근성 지원 컴포넌트를 우선적으로 사용한다.

국가표준 <항목 4.1>에 따르면, Flash 콘텐츠가 제공하는 중요한 정보는 보조 기술을 이용해 읽을 수 있어야 하며, 키보드를 이용하여 사용할 수 있도록 Flash가 제공하는 접근성 지원 컴포넌트를 사용하여야 한다.

3.12.1 요구조건

- (1) Flash 콘텐츠 개발 시에는 Flash가 제공하는 접근성 지원 컴포넌트들을 우선적으로 사용한다.
- (2) 사용자가 컴포넌트를 자체적으로 만들어 사용할 경우에 이들 컴포넌트는 접근성을 지원해야 한다.

3.12.2 적용방법

가) 접근성 지원 컴포넌트

Flash는 접근성을 지원하는 콘텐츠를 빠르게 개발하는데 필요한 사용자 인터페이스 컴포넌트를 제공한다. 개발자는 이들 컴포넌트를 이용하면 접근성 있는 레이블 제공, 키보드를 이용한 접근 및 기타 접근성 관련 테스트 등을 자동적으로 수행할 수 있다.

Flash CS4가 제공하는 접근성관련 컴포넌트는 다음과 같다.

- Button (버튼)
- Checkbox (체크박스)
- Radio button (라디오 버튼)
- Label (레이블)
- TextInput (텍스트입력)
- TextArea (텍스트영역)
- ComboBox (콤보박스)
- ListBox (리스트박스)
- DataGrid (데이터그리드)

- Window (창)
- Alert (경고)

나) ActionScript 3.0 컴포넌트

ActionScript 3.0 컴포넌트는 `enableAccessibility()` 명령어를 사용하여 접근성 관련 객체를 활성화 시킬 수 있다. 어떤 객체를 컴포넌트에 추가한 후에는 이 객체를 간단히 제거할 수 있는 수단이 없으므로 접근성 관련 옵션은 초기 값이 '끔' 상태이다. 따라서 개발자는 각 컴포넌트의 접근성 기능을 활성화('켄')시켜야 한다. 이 절차는 매 컴포넌트 별로 한번만 하면 되며, 컴포넌트에 속한 인스턴스들은 접근성을 개별적으로 활성화시킬 필요가 없다. 따라서 이 명령어는 무비의 첫 번째 프레임에 삽입하는 것이 좋다. 아래 코드는 체크박스 컴포넌트에 접근성 활성화 명령어를 추가한 것이다.

```
import fl.accessibility.CheckBoxAcclImpl;
CheckBoxAcclImpl.enableAccessibility( )
```

다) 사용자 컴포넌트

컴포넌트가 접근성을 지원하기 위해서는 MSAA(Microsoft Active Accessibility) 규격에 의거하여 컴포넌트의 역할과 상태에 관한 정보를 알려줄 수 있어야 한다. 만일 개발자들이 컴포넌트를 직접 개발하여 사용하는 경우라면, 개발할 Flash 컴포넌트들이 기존의 컴포넌트와 동일한 특성을 가지도록 개발 초기부터 MSAA에 대한 검토와 구현계획을 세워야 한다.

MSAA가 컨트롤에 추가되면 이들 컴포넌트가 보조 기술(예를 들면 화면 낭독 프로그램)을 정상적으로 작동시키는 가를 확인해야 한다. 화면 낭독 프로그램은 MSAA 규격을 전부 지원하지는 않는다. 대개 12가지 기본적인 컨트롤을 지원하며, 이들 컨트롤은 HTML 표준에서도 지원하고 있다. 사용자가 기존의 컴포넌트를 변형하던지 새로운 컨트롤을 추가한 컴포넌트를 만들어 사용하고자 할 경우에는 화면 낭독 프로그램 개발자와의 협력이 필수적이다.

Flash에서 제공하는 컴포넌트는 애플리케이션 콘텐츠의 빠른 개발을 위하여 제공하는 것이다. 사용자가 접근성을 지원하는 컴포넌트를 개발하여 사용하는 것은 MSAA나 화면 낭독 프로그램과의 호환성을 고려한다면 평범한 일은 아니다. 따라서 개발자들은 가능하면 Flash가 제공하는 컴포넌트를 사용할 것을 권장한다.

4. Flash 콘텐츠의 접근성 평가방법

이 문서에서는 접근성 있는 콘텐츠를 개발하는데 있어서 매우 제한적인 지침을 제공할 뿐이다. 따라서 Flash 개발자는 접근성을 평가하기 위한 다양한 방법을 적용해 보아야 한다. 아래의 체크리스트는 Flash 부분과 관련한 접근성 검사목록을 발췌한 것이다.

또한, Flash를 포함하고 있는 웹 콘텐츠의 평가방법에 대해서는 <I - 4. RIA 콘텐츠 접근성 평가방법>을 참고하라.

| 번호 | 항목 | 검사목록 | 국가 표준 |
|----|----------|--|-------|
| 1 | 대체 텍스트 | (1) '텍스트가 아닌 콘텐츠에 대해서는 적절한 대체 텍스트를 제공하고 있는가? (2) 동적으로 교체된 이미지는 이미지에 알맞게 대체 텍스트도 교체하고 있는가? | 1.1 |
| 2 | 자막 | (3) 멀티미디어 콘텐츠는 동기화된 자막을 제공하고 있는가? | 1.2 |
| 3 | 색상 | (4) 색상 이외에도 명암이나 패턴으로 콘텐츠 구분이 가능한가? (5) 흰 바탕에 밝은 회색글자처럼 판독이 어려운 색조합을 피하고, 흰 바탕에 검정 글자처럼 대비 차이가 큰 색조합을 사용하고 있는가? (6) 저시력자나 다양한 환경의 사용자도 텍스트를 쉽게 읽을 수 있도록 적절한 크기로 텍스트를 제공하고 있는가? | 1.3 |
| 4 | 깜빡거리는 객체 | (7) 깜빡이는 콘텐츠가 있을 경우 사전 경고를 제공하고 있는가? | 2.3 |

| | | | |
|----|-------------|--|-----|
| 5 | 키보드 | (8) 키보드만으로 콘텐츠가 제공하는 모든 기능에 대한 제어가 가능한가? (9) 초점이 주어지는 것만으로 상황이 바뀌지 않는가? | 2.4 |
| 6 | 애니메이션 | (10) 무비의 움직임이 웹 페이지의 갱신을 유발하지 않는가? | 2.6 |
| 7 | 오디오/비디오 컨트롤 | (11) 키보드만으로 비디오 또는 오디오 재생 컨트롤을 사용할 수 있는가? (12) 배경음악의 크기를 사용자가 조절할 수 있는가? | 2.3 |
| 8 | 읽는 순서 | (13) 키보드로 폼 컨트롤, 링크, 객체 사이를 이동할 때 논리적 이동 순서가 적절한가? | 3.2 |
| 9 | 구조 | (14) 복잡한 Flash 콘텐츠는 사용법에 대한 대체 텍스트를 제공하고 있는가? | 3.2 |
| 10 | 대체 페이지 | (15) Flash 콘텐츠가 접근성을 제공하지 못할 경우 대체 페이지를 제공하고 있는가? (16) 불필요한 Flash 콘텐츠는 초점이 주어지지 않도록 하였는가? | 4.1 |
| 11 | 플러그인 | (17) 콘텐츠를 사용하는데 필요한 플러그인은 사용자가 설치하여 사용할 수 있는가? ※ 설치할 수 없다면 설치 방법을 제공하여야 함. | 4.1 |

4.1 대체 텍스트 제공

4.1.1 체크리스트

- (1) ‘텍스트가 아닌 콘텐츠’에 대해서는 아래와 같이 대체 텍스트를 제공하고 있는가?
 - (2) 스크립트로 교체하는 이미지는 아래와 같이 교체되는 이미지에 알맞은 대체 텍스트도 교체하고 있는가?
- ‘텍스트가 아닌 콘텐츠’에 대해서는 대체 텍스트를 제공한다. 이미지, 멀티미디어 콘텐츠, 그래픽 아이콘, 그래픽 글자 등은 비록 텍스트의 형태를 취하고 있더라도 코드로 사용될 수 없다. 왜냐하면 이들은 보조 기술에 제공되더라도 그 내용을 알 수 없기 때문이다. 따라서 보조 기술이 코드로 인식할 수 없는 모든 객체를 ‘텍스트가 아닌 콘텐츠’라고 정의할 수 있다. ‘텍스트가 아닌 콘텐츠’는 보조 기술이 인지할 수 있도록 대체 텍스트를 추가로 제공하여야 한다.
 - ~링크, ~버튼, ~로고, ~바로가기, ~메뉴와 같이 중복될 수 있는 내용을 대체 텍스트로 제공하지 않는다. 화면 낭독 프로그램은 Flash 콘텐츠에 포함된 버튼이나 무비클립 등의 요소를 만나면 해당 요소의 용도를 ‘버튼’ 또는 ‘무비클립’이라고 읽어준다. 따라서 이들 버튼의 대체 텍스트를 액세스 가능성 패널의 이름 필드나 설명 필드에 기록하는 과정에서 ‘... 하기 버튼’ 또는 ‘... 하기 무비클립’이라고 작성하면 ‘버튼, ... 하기 버튼’ 또는 ‘무비클립, ... 하기 무비클립’이라고 앞뒤에서 각 구성요소에 대한 이름을 두

번 읽어주므로 굳이 대체 텍스트에 ‘버튼’ 또는 ‘무비클립’이라는 단어를 기입할 필요가 없다. 가장 바람직한 경우는 불필요한 내용을 제거하고 핵심내용만 이름 필드 또는 설명 필드를 기입한다. 메뉴의 경우도 마찬가지이다.

- 대체 텍스트는 콘텐츠의 내용을 파악할 수 있는 핵심적인 설명을 제공한다. 대체 텍스트는 콘텐츠를 직관적으로 나타낼 수 있는 내용으로 구성하는 것이 바람직하다. <IV - 3.1.2 나) 이미지와 화면 낭독 프로그램>에서 예를 든 것처럼, 투명한 플라스틱 병에 생수가 가득 들어있는 이미지에 대한 대체 텍스트로는 ‘투명한 액체가 들어있는 플라스틱 병’이라고 하기보다는 ‘생수병’이라고 하는 것이 바람직하다.
- 대체 텍스트가 필요하지 않은 경우에는 대체 텍스트를 null로 제공한다. 때로는 대체 텍스트를 추가함으로 인하여 혼란을 야기시키는 경우가 있다. 예를 들면 이름 필드와 설명 필드를 동일하게 기입하면 두 번 연속해서 읽어주게 된다.
- 교체되는 이미지의 대체 텍스트가 교체 전과 동일하면 대체 텍스트를 교체할 필요가 없다. 교체하는 이미지의 대체 텍스트가 교체하기 전과 동일할 경우에는 대체 텍스트를 교체하는 것이 불필요하다. 스크립트를 이용하여 콘텐츠를 동적으로 교체하는 것은 불필요한 자원의 낭비를 초래하므로 가능한 한 대체 텍스트를 교체하지 않는다. 그러나 이미지의 내용이 교체전과 비교하여 확연한 차이를 보일 경우에는 대체 텍스트를 교체하는 것이 당연하다.

4.1.2 관련 국가 표준 : 항목 1.1

4.1.3 Flash 콘텐츠 지침 : 항목 3.1

4.2 자막 제공 방법

4.2.1 체크리스트

(3) 멀티미디어 콘텐츠는 동기화된 자막을 제공하고 있는가?

- Flash를 이용한 오디오와 비디오 콘텐츠에는 FLV 또는 H.264 규격의 비디오에 간단히 자막을 추가할 수 있다. Flash에 의하여 만들어진 동영상 파일에는 아래 그림과 같이 자막이 포함되어야 한다.



그림 IV - 10. 자막파일을 제공한 예제

- 사용자의 필요에 따라 자막이 표시되는 위치변경, 폰트변경, 전체 화면 보기 시에 폰트 크기 변경 등이 가능하면 더욱 바람직하다.
- 동기화된 자막을 제공하는 것이 원칙이지만, 콘텐츠의 내용을 충분히 이해할 수 있는 원고를 제공하는 것도 허용된다.

4.2.2 관련 국가 표준 : 항목 1.2

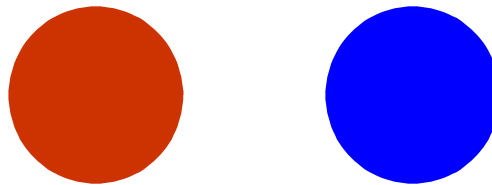
4.2.3 Flash 콘텐츠 지침 : 항목 3.2

4.3 색상과 접근성

4.3.1 체크리스트

(4) 색상 이외에도 명암이나 패턴으로 콘텐츠 구분이 가능한가?

- 웹 콘텐츠는 색상을 사용할 경우에 색상만으로 정보를 제공하지 않도록 한다. 예를 들어 아래 <그림 IV - 11(a)>와 같이 파란 버튼과 빨간 버튼을 제공하고 “파란 버튼을 누르시오” 또는 “빨간 버튼을 누르시오” 라고 사용법이 제시된 콘텐츠의 경우, <그림 IV - 11(b)>와 같이 흑백화면에서는 색상 구분이 불가능하기 때문이다.



(a) 칼라모니터에 나타난 모양



(b) 흑백모니터에 나타난 모양그림

IV - 11. 색상으로만 표시된 버튼

- 색상과 함께 추가적인 정보를 제공하면 이 문제가 해결될 수 있다. 예를 들어, “앞으로 이동하려면 왼쪽의 빨간 버튼을 클릭하십시오” 또는 “뒤로 이동하려면 오른쪽의 파란색 버튼을 클릭하십시오” 처럼 위치 정보를 제공하면 색각이상자도 접근이 가능하게 된다.
- 더욱 바람직한 방법은 <그림 IV - 11(a)>를 아래 <그림 IV - 12(a)>와 같이 색상과 점선 또는 점으로 도형을 표시하는 것이다. <그림 IV - 12(b)>는 흑백 모니터에서도 충분히 구별이 가능하다.

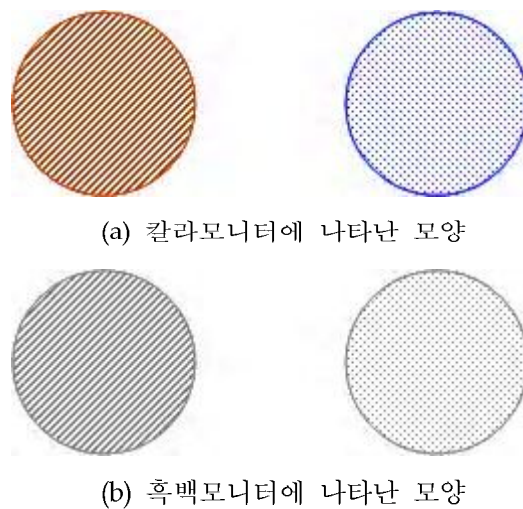


그림 IV - 12. 색상으로만 표시된 버튼

- (5) 흰 바탕에 밝은 회색글자처럼 판독이 어려운 색조합을 피하고, 흰 바탕에 검정 글자처럼 대비 차이가 큰 색조합을 사용하고 있는가?
- 웹 콘텐츠의 가독성을 향상시키기 위해서는 전경색과 배경색의 대비를 충분히 차이가 나게 하여야 한다. 콘텐츠의 가독성을 평가

하는 한 가지 방법은 흑백 화면에서 콘텐츠를 충분히 판독할 수 있는가를 확인하는 것이다. 색상의 대비 차이가 작으면 화면에 보이는 요소를 읽기 어렵다.

(6) 저시력자나 다양한 환경의 사용자도 텍스트를 쉽게 읽을 수 있도록 적절한 크기로 텍스트를 제공하고 있는가?

- 콘텐츠를 만들 때 화면에 표시되는 화면 구성 요소(텍스트, 버튼 등)의 크기를 브라우저에서 조절할 수 있어야 한다. 만일 크기를 조절할 수 없다면 텍스트의 글씨 크기는 12호 이상을 사용하여야 한다.

4.3.2 관련 국가 표준 : 항목 1.3

4.3.3 Flash 콘텐츠 지침 : 항목 3.3

4.4 깜빡거리는 객체 사용 제한

4.4.1 체크리스트

(7) 깜빡이는 콘텐츠가 있을 경우 사전 경고를 제공하고 있는가?

- 웹사이트에 처음 접속했을 때에 화면에 나타나는 홈 페이지에는 최소한 깜빡거림이 없어야 한다. 만일 다음 페이지부터 깜빡거림이 있는 콘텐츠가 포함되어 있다면, 홈 페이지에 이러한 내용을 표시하여 사용자가 다음 페이지로 이동할 것인가를 선택할 수 있도록 해야 한다.

- Flash 콘텐츠를 목적상 깜박거리게 만들어야 하는 경우에는 임베드되는 Flash 콘텐츠의 크기를 가로와 세로 모두 200 픽셀 이하가 되도록 한다.

4.4.2 관련 국가 표준 : 항목 2.3

4.4.3 Flash 콘텐츠 지침 : 항목 3.4

4.5 키보드의 이용

4.5.1 체크리스트

(8) 키보드만으로 콘텐츠가 제공하는 모든 기능에 대한 제어가 가능한가?

- 웹 콘텐츠는 키보드만으로 사용이 가능하여야 한다. 또한 화면 낭독 프로그램과 같은 보조 기술과의 단축키 충돌이 없어야 한다.
- 특히 Tab 키와 Shift + Tab 키에 의한 초점 이동이 원활하여야 한다. 이 과정에서 가능한 한 불필요한 요소로 초점이 이동하지 않도록 하여 사용성을 높인다.

(9) 초점이 주어지는 것만으로 상황이 바뀌지 않는가?

- 화면 낭독 프로그램이 사용되는 경우에도 Tab 키와 Shift + Tab 키에 의한 초점 이동의 순서가 일치해야 한다. 또한 화면 낭독 프

로그래밍에 의하여 초점의 사라지는 현상 등이 발생하지 않아야 한다.

- 초점이 주어지는 것만으로 상황이 바뀌어서는 안 된다. 반대로 활성화 시에는 상황이 바뀌어야 한다.
- 화면 낭독 프로그램을 사용할 때와 사용하지 않을 때 키보드 내비게이션에 차이가 없어야 한다.

4.5.2 관련 국가 표준 : 항목 2.4

4.5.3 Flash 콘텐츠 지침 : 항목 3.5

4.6 접근성 지원 애니메이션

4.6.1 체크리스트

(10) 무비의 움직임이 웹 페이지의 갱신을 유발하지 않는가?

- 몇 개의 프레임을 반복해서 보여주도록 만든 어떤 광고용 배너를 Flash Player로 화면에 표시하는 경우에 이 배너는 화면의 변화를 계속적으로 유발시켜 화면 낭독 프로그램은 반복적으로 웹 페이지의 처음부터 읽어주려고 한다. 이러한 문제는 화면 낭독 프로그램 사용자들이 자주 봉착하는 현상이다.
- 만일 이러한 문제를 피할 수 없다면 무비의 움직임이 정지된 상

태로 Flash 콘텐츠를 제공한다. 즉 애니메이션의 구현시에 사용자가 클릭하거나 명시적으로 재생을 요구하지 않는 한 정지된 상태로 제공되어야 한다.

4.6.2 관련 국가 표준 : 항목 2.6

4.6.3 Flash 콘텐츠 지침 : 항목 3.6

4.7 오디오/비디오 컨트롤

4.7.1 체크리스트

(11) 키보드만으로 비디오 또는 오디오 재생 컨트롤을 사용할 수 있는가?

- Flash CS4가 제공하는 스킨을 이용하면 FLVPlayback 컴포넌트를 스테이지에 끌어다 놓기만 하면 생성된 비디오 또는 오디오 콘텐츠는 접근성이 제공되는 재생관련 컨트롤을 이용할 수 있다.

(12) 배경음악의 크기를 사용자가 조절할 수 있는가?

- Flash Player은 Flash 콘텐츠를 재생하기에 앞서 보조 기술이 동작하고 있는지를 검사하여 화면 낭독 프로그램이 동작 중일 경우에는 오디오 컨트롤을 음소거(Mute)로 설정하거나 볼륨을 매우 작게 조절하여 화면 낭독 프로그램 사용자의 불편함을 줄이도록 구현하여야 한다.

4.7.2 관련 국가 표준 : 항목 2.3

4.7.3 Flash 콘텐츠 지침 : 항목 3.7

4.8 읽는 순서의 결정

4.8.1 체크리스트

(13) 키보드로 Flash 구성 요소 간을 이동할 때 논리적 이동 순서가 적절한가?

- Flash 무비의 요소들을 읽는 순서는 화면 왼쪽 상단에서 화면 오른쪽 아래 방향이라는 전통적인 기준을 따르지 않는다. 따라서 순서가 뒤바뀐 내용을 화면 낭독 프로그램을 통하여 읽고 이를 이해하기는 매우 어렵다.
- 객체간의 이동은 Tab 키와 Shift + Tab 키를 이용하여 각각 순방향 및 역방향 이동이 가능하다. 이때 이동하는 순서는 화면에 보인 배치 순서와 같거나, 읽어주는 순서가 논리적으로 적합하게 구성되는 것이 바람직하다. 예를 들어 아래 <그림 IV - 13(a)>의 콘텐츠는 3줄로 버튼을 늘어놓아 키보드를 형상화 한 것이다. 초점의 이동 순서는 <그림 IV - 13(b)>와 같이 이동하여야 한다.



(a) 버튼의 구성



(b) 버튼의 초점 이동 순서

그림 IV - 13. 콘텐츠의 초점 이동 순서에 관한 예제

(한국 Adobe 제공)

화면에 보이는 위치와 초점이 이동하는 순서를 일치시키기 위해서는 초점의 이동 순서를 결정하는 과정이 필요하다. 초점의 이동 순서를 설정하는 방법은 <IV- 3.8 읽는 순서의 설정>을 참조하라.

4.8.2 관련 국가 표준 : 항목 3.2

4.8.3 Flash 콘텐츠 지침 : 항목 3.8

4.9 구조의 제공

4.9.1 체크리스트

(14) 복잡한 Flash 콘텐츠는 사용법에 대한 대체 텍스트를 제공하고 있는가?

- Flash를 이용하면 Flex나 데스크톱 수준의 애플리케이션을 구현할 수 있다. 따라서 복잡한 기능을 구현한 Flash 콘텐츠는 장애여부에 관계없이 처음 접한 사용자, 특히 화면 낭독 프로그램 사용자에게는 웹 사이트의 구성, 사용하는 컨트롤, 작업을 종료하는 방법 등에 대한 정보를 충실히 제공할 필요가 있다.
- 사용법을 제공하는 가장 간단한 방법은 Flash 무비 전체에 대한 대체 텍스트를 액세스 가능성 패널의 description 필드에 기입하는 것이다. 이 때 주의할 점은 사용법에 대한 설명을 간단히 작성하는 것이다. 그 이유는 자주 읽을 가능성이 많기 때문이다.
- 매우 복잡한 사용법의 경우에는 사용법을 별도의 화면으로 구성한다. 즉 버튼이나 링크를 클릭하면 새 창을 열어 콘텐츠의 사용법을 제시하는 것이다. 사용법 소개용 새 창 열기 링크나 버튼을 화면의 하단에 배치하여 장애인 뿐 아니라 일반인들도 사용법에 관한 정보를 제공받을 수 있도록 하는 것도 바람직한 방법이다. Flash에서 새 창 열기 방법은 <IV- 3.10.2 가) 대체 페이지 제공>을 참고하라.

4.9.2 관련 국가 표준 : 항목 3.2

4.9.3 Flash 콘텐츠 지침 : 항목 3.9

4.10 대체 페이지 제공

4.10.1 체크리스트

(15) Flash 콘텐츠가 접근성을 제공하지 못할 경우 대체 페이지를 제공하고 있는가?

- 접근성을 지원하지 못하는 Flash 콘텐츠를 이용하여 웹 콘텐츠를 구성해야 하는 경우에는 Flash 콘텐츠가 포함된 웹 콘텐츠를 대신하여 별도의 대체 페이지를 제공해야 한다.

(16) 불필요한 Flash 콘텐츠는 초점이 주어지지 않도록 하였는가?

- 광고 배너, 화면 치장용 Flash 콘텐츠는 그 내용을 보조 기술 사용자에게 전달할 필요성이 불필요하므로 화면 낭독 프로그램이 판독하지 않도록 감춘다.
- Flash 콘텐츠를 화면 낭독 프로그램이 접근하는 것을 감추기 위해서는 웹 페이지에서 Flash 부분으로 초점이 주어지지 않도록 해야 한다.

4.10.2 관련 국가 표준 : 항목 4.1

4.10.3 Flash 콘텐츠 지침 : 항목 3.10

4.11 플러그인 정보 제공

4.11.1 체크리스트

(17) 콘텐츠를 사용하는데 필요한 플러그인은 사용자가 설치하여 사용할 수 있는가?

- Flex 콘텐츠는 실행을 위해서는 다양한 플러그인을 필요로 한다. 만일 Flex 콘텐츠를 실행하는 과정에서 필요한 플러그인은 자동적으로 다운로드 받아 설치할 수 있어야 한다.
- 플러그인의 자동적인 설치가 불가능 하거나 설치하는 과정에서 사용자의 개입이 필요할 경우에는 해당 플러그인을 제공하는 웹사이트 정보, 플러그인의 설치방법, 사용법 등에 관한 정보를 제공한다.

4.11.2 관련 국가 표준 : 항목 4.1

4.11.3 Flash 콘텐츠 지침 : 항목 3.11

5. Flash 콘텐츠 구현 예

Flash 콘텐츠는 JavaScript를 이용하여 개발하는 웹 콘텐츠에 비하여 접근성을 제공하는 방법이 매우 간단하다. 따라서 제 5절에서는 여러 가지 복잡한 애플리케이션 예를 다루기보다는 Flash 콘텐츠의 접근성을 향상시키는 절차에 대한 설명을 중심으로 기술할 것이다.

또한 이 절에서는 Flash 콘텐츠를 제작하는 방법에 대한 사항은 자세하게 다루지 않는다. Flash 콘텐츠를 제작하는 자세한 과정이나 세부 제작 기법(예를 들면 부드러운 2단 내비게이션 등)에 관한 사항은 별도의 자료를 참고하라. 이 지침에서는 Flash 콘텐츠를 제작함에 있어서 접근성을 고려해야 하는 사항을 주로 다룰 것이다.

5절에서는 웹 사이트 및 웹 애플리케이션 콘텐츠를 개발할 때 자주 사용되는 5가지 유형의 간단한 예제를 기술할 것이다. 따라서 이 문서에서 다른 유형별 예제를 숙지한다면 더 복잡한 Flash 콘텐츠의 접근성 제공도 가능하다.

또한 이 문서에서 사용한 소스 코드는 온라인 사이트를 통하여 제공할 것이다.

5.1 내비게이션 메뉴

5.1.1 가로 1단 메뉴

이 예제는 가로 1단 메뉴에서 접근성을 준수하여 제작하는 과정을 살펴볼 것이다. 메뉴는 아래 <그림 IV - 14>와 같이 5개의 링크가 가로 1단으로 구성된다.

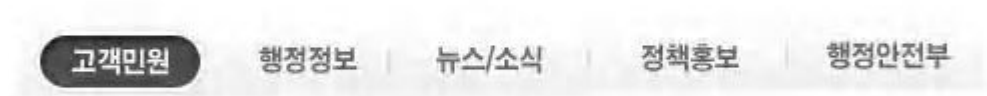


그림 IV - 14. 가로 1단 메뉴 예제

1) 메뉴 심볼의 준비

우선 <그림 IV - 14>와 같은 가로 1단 메뉴를 제작하기 위해서는 버튼의 역할을 하는 사각형에 정적 텍스트로 메뉴 레이블을 기입한 다섯 개의 심볼이 필요하다. 이 과정은 Flash 콘텐츠 제작의 기본 과정이므로 생략한다.

2) 접근성 기능의 활성화

메뉴를 구성하는 요소가 준비되었으면, Flash 저작도구에서 메뉴 컨트롤을 구성할 마우스 포인터를 스테이지에 위치하고 메뉴(윈도/기타 패널/액세스 가능성)를 이용하거나 Shift + F11 키를 눌러 아래 <그림 IV - 15>와 같은 액세스 가능성 패널을 화면에 띄운다.

액세스 가능성 패널의 세부 옵션에 대한 설명은 <IV- 2.1 접근 가능한 Flash 콘텐츠 구현>을 참고하라. 패널에서 무비를 액세스 가능하게 만들기 옵션, 자식 객체 액세스 가능 옵션 및 자동 레이블 옵션을 모두 활성화 시킨다. 이 옵션을 활성화시키면 앞으로 만들어지는 메뉴에는 접근성이 제공된다. 접근성이 제공된다는 말은 Tab 키를 이용하여 어떤 메뉴에 초점이 주어진다면, 해당 메뉴의 레이블을 보조 기술로 제공하게 되어, 화면 낭독 프로그램이 읽어준다는 의미이다.

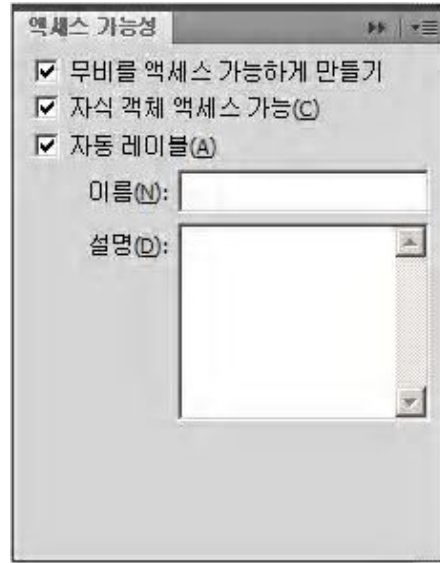


그림 IV - 15. 접근성을 제공하기
위한 액세스 가능성 패널 화면

또한 액세스 가능성 패널에서 이름란과 설명란은 비워둔다. 이 부분에 텍스트를 기입하면, Tab 키를 이용하여 메뉴에 초점이 주어질 때마다 이름란과 설명란에 기입한 텍스트를 읽어주게 된다. 따라서 메뉴의 구성이나 사용법이 기존의 메뉴 구성과 다르거나 독특할 경우에는 설명란을 이용하여 메뉴의 사용법을 제공할 수 있다.

3) 메뉴의 구성

메뉴 전체를 접근성 있게 준비하는 과정이 완료되면, 개별 메뉴 버튼에 대한 접근성 관련 설정을 하여야 한다. 즉 각 버튼별로 아래 <그림 IV - 16>과 같이 액세스 가능성 패널을 열어 객체를 액세스 가능하게 만들기 옵션을 활성화 시킨다.

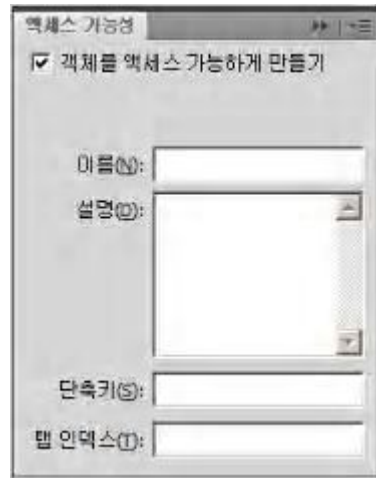


그림 IV - 16. 메뉴 전체를 점
근성 있게 하는 방법

이 옵션을 활성화 시키면 해당 메뉴에 초점이 주어졌을 때에 메뉴 레이블을 읽어주거나 이름난과 설명난을 읽어주게 된다. 만일 이름이나 설명난에 텍스트가 기입되어 있을 경우에는 메뉴 레이블 대신 이름이나 설명을 읽어준다. 이 과정을 모든 메뉴 버튼에 대해서 수행한다.

4) Tab 키의 이동 순서 설정

Tab 키에 의한 이동순서는 메뉴 버튼의 제작 순서에 따라 결정된다. 대부분 큰 문제가 없으나, Tab 키에 의한 초점의 이동 순서가 당초 예상했던 순서와 다를 경우에는 액세스 가능성 패널의 탭 인덱스란을 이용하여 Tab 키에 의한 이동 순서를 통제할 수 있다. 예를 들면 5개의 메뉴 버튼의 Tab 인덱스를 각각 1, 2, 3, 4, 5 로 부여할 수 있다. 여기서 Tab 인덱스의 값은 크기와 순서가 중요할 뿐 연속된 숫자인가는 중요하지 않다.

5) ActionScript의 작성

마지막으로 각 메뉴에 대한 ActionScript 패널을 열어 각 메뉴를 눌렀을 때에 이동할 URL을 기입하면 접근성 있는 가로 1단 메뉴가 완성된다.

이렇게 만들어진 메뉴는 Tab 키로 이동하는 과정에서 순차적으로 초점이 주어지며, 이때마다 해당 메뉴에 대한 이름과 설명 또는 레이블을 읽어준다. 센스리더의 경우에는 Flash에 대한 접근성 지원이 미흡하여 URL을 읽어준다.

5.1.2 가로 2단 롤오버 메뉴

Flash 콘텐츠 중에서 일반적으로 사용되는 가로 2단 메뉴의 경우에도 가로 1단 메뉴와 제작방법이 동일하다. 예를 들어 아래 <그림 IV - 17>과 같이 가로 2단 메뉴를 구성하고, 하위 메뉴는 주 메뉴에 초점이 주어졌을 때에 자동적으로 나타나도록 한다. 즉 롤오버 메뉴의 형태가 된다.

가로 2단 롤오버메뉴를 제작하는 과정은 가로 1단 메뉴를 제작하는 과정과 유사하다.



그림 IV - 17. 가로 2단 롤오버 메뉴 예제

1) 주 메뉴 및 하위 메뉴 심볼의 준비

<그림 IV - 17>과 같은 가로 2단 메뉴를 제작하기 위해서는 가로 1단 메뉴와 같이 주 메뉴를 구성할 다섯 개의 심볼이 필요하다. 이 과정은 Flash 콘텐츠 제작의 기본 과정이므로 생략한다.

이어서 하위 메뉴별로 별도의 레이어에 개별 주 메뉴의 하위 메뉴를 백색 직사각형 배경에 넣어놓는다. <그림 IV - 17>에서는 정적 텍스트를 심볼로 변경하여 만들었다.

2) 접근성 기능의 활성화

스테이지의 액세스 가능성 패널 설정은 <IV - 5.1.1 가로 1단 메뉴>와 동일하다.

3) 메뉴의 구성

주 메뉴의 액세스 가능성 패널 설정은 <IV- 5.1.1 가로 1단 메뉴>와 동일하다. 하위 메뉴의 접근성 설정 과정도 주 메뉴의 설정과정과 동일하다.

4) Tab 키의 이동 순서 설정

Tab 키에 의한 이동순서는 메뉴 버튼의 제작 순서에 따라 결정된다. 대부분 큰 문제가 없으나, Tab 키에 의한 초점의 이동 순서가 당초 예상했던 순서와 다를 경우에는 액세스 가능성 패널의 탭 인덱스란을 이용하여 Tab 키에 의한 이동 순서를 통제할 수 있다. 즉 가로 2단 롤오버 메뉴

의 초점 이동 순서는 주 메뉴, 해당 주 메뉴의 첫 번째 하위 메뉴, 두 번째 하위 메뉴, ... 마지막 하위 메뉴의 순서가 되어야 한다. 따라서 탭 인덱스는 가로 1단 메뉴와 조금 다르게 설정하여야 한다.

가장 쉬운 방법은 주 메뉴의 탭 인덱스를 10, 20, 30, 40, 50으로 부여하고, 첫 번째 주 메뉴의 하위 메뉴의 탭 인덱스를 각각 11, 12, 13, ... 등으로 부여한 후, 두 번째 주 메뉴의 하위 메뉴의 탭 인덱스를 21, 22, 23, ... 등으로 부여하는 것이다. 이 과정을 모든 하위 메뉴에 대해서 수행하면 접근성을 제공하기 위한 모든 준비가 완료된다.

5) 하위 메뉴의 연결과 ActionScript의 작성

가로 2단 메뉴에서는 Flash 저작도구의 타임라인을 이용하여 주 메뉴와 하위 메뉴를 연결하여야 한다. 이 과정은 전형적인 Flash 콘텐츠 제작 과정이므로 자세한 설명을 생략한다. 또한 하위 메뉴별로 이동할 URL을 기입하기 위하여 ActionScript를 작성하는 과정도 생략한다.

이렇게 만들어진 가로 2단 메뉴는 Tab 키로 이동하는 과정에서 해당 주 메뉴 또는 하위 메뉴에 초점이 주어지며, 이때마다 해당 메뉴에 대한 이름과 설명 또는 레이블을 읽어준다. 센스리더의 경우에는 Flash에 대한 접근성 지원이 미흡하여 URL을 읽어준다.

Flash 콘텐츠는 가로 2단 롤오버 메뉴의 경우에 Up/Down 키를 이용하여 하위 메뉴로의 이동이 가능하다. Left/Right 키를 이용하면 주 메뉴간의 이동이 가능하다. 또한 Enter 키를 이용하면 초점이 주어진 메뉴를 선택할 수 있다.

5.1.3 가로 2단 메뉴와 Enter 키의 조합

<그림 IV - 17>에 보인 가로 2단 롤오버 메뉴에서 롤오버 기능을 생략하고 주 메뉴에 초점이 있을 때 Enter 키를 누르면 주 메뉴에 속한 하위 메뉴가 펼쳐지고, 하위 메뉴의 첫 번째 메뉴로 초점이 이동하도록 할 수 있다.

이 경우의 제작 방법도 <IV - 5.1.2 가로 2단 롤오버 메뉴>와 동일하다. 다만 마우스 클릭에 의한 이벤트가 발생하였을 때에 하위메뉴가 나타나고 해당 하위 메뉴의 첫 번째 메뉴에 초점이 제공되도록 ActionScript를 변경하면 된다.

5.2 사용자 UI를 고려한 콘텐츠

웹 페이지에 가장 많이 사용되는 Flash 콘텐츠의 하나가 스크롤 배너이다. 스크롤 배너에도 일정한 시간 간격으로 스스로 스크롤하는 배너, 좌우 이동 버튼을 누를 때마다 좌우로 스크롤 하는 배너, 그리고 배너의 수만큼 버튼을 제공하여 직접 해당 배너로 이동할 수 있도록 한 배너 등이 대표적이다.

5.2.1 자동 스크롤 배너

자동 스크롤 배너는 일정한 시간 간격이 지나면 스스로 다음 배너로 스크롤 하는 배너이다. 사용방법은 마우스 포인터를 올려놓거나 Tab 키로 이동하여 초점이 주어지면, 배너가 더 이상 회전하지 않고 키보드의 반응이나 마우스 반응을 기다린다.

자동 스크롤 배너를 접근성 있게 만들기 위해서는 1차로 모든 배너 이

미지를 무비클립이나 버튼으로 만든다. Flash 콘텐츠에서 링크를 제공할 수 있는 요소는 무비클립이거나 버튼이어야 한다.

버튼에는 자식 객체가 없으므로 그 자체의 접근성을 설정하면 된다. 그러나 무비 클립에는 자식 객체가 존재하므로 각각의 접근성 여부를 설정해줘야 한다.

무비클립에서 자식객체 액세스 가능 옵션을 비활성으로 설정하면 버튼과 동일한 기능을 갖는다.

자동 스크롤 배너를 5개의 무비클립으로 구성한다고 하면, 이들 5개의 무비클립은 화면에 나타나는 순서와 동일한 순서로 탭 인덱스를 설정하여야 한다. 앞에서도 이야기 하였지만 액세스 가능성 패널에서 설정한 탭 인덱스는 Tab 키에 의한 이동 순서를 의미한다.

화면 낭독 프로그램은 초점이 주어진 무비 클립에 대한 대체 텍스트를 읽어주므로 모든 무비클립에 해당하는 액세스 가능성 패널의 이름란 또는 설명란에 적절한 대체 텍스트를 제공하여야 한다.

예를 들어 아래 <그림 IV - 18>과 같이 6개의 이미지로 구성된 자동 스크롤 배너를 구성하는 경우에 각 이미지의 액세스 가능성 패널 설정은 <표 IV - 2>와 같다.



그림 IV - 18. 6장의 배너로 구성된 자동 스크롤 배너 예제

표 IV - 2. 자동스크롤 배너의 액세스 가능성 패널 설정 구성

| 이미지 번호 | 하이퍼 텍스트 링크 | 액세스 가능성 패널 설정 | | | | |
|-----------|------------------|---------------------------|--------------------|-------------|------------------------------------|-------------|
| | | 무비를 액세스 가능하게 만들기 | 자식 객체 액세스 가능 | 이름 | 설명 | Tab인 텍스트 |
| 1 | http:// ... | 체크 | 언체크 | 질의응답 | 질의분야를 선택하신 후... 공간입니다. | 1 |
| 2 | http:// ... | 체크 | 언체크 | 나의질의 조회 | 질의하신 질문과 ... 공간입니다. | 2 |
| 3 | http:// ... | 체크 | 언체크 | 고객센터 FAQ | 질의를 신청하기전... 확인하실 수 있습니 다. | 3 |
| 4 | http:// ... | 체크 | 언체크 | 고객상담 안내 | 고객의 효과적인 상 담을 ... 안내페이지 입니다. | 4 |
| 5 | http:// ... | 체크 | 언체크 | 국민제안 | 행정정책과 ... 제안 하실 수 있습니다. | 5 |
| 6 | http:// ... | 체크 | 언체크 | 여론광장 | 생각을 자유롭게 ... 확인하실 수 있습니 다. | 6 |

자동 스크롤 배너 자체를 Flash로 제작하는 과정에 대한 설명은 이 문서에서 생략한다.

5.2.2 버튼을 이용한 스크롤 배너

버튼을 이용한 스크롤 배너는 배너의 수만큼 버튼을 제공하고, 버튼에 초점이 주어질 때나 마우스 포인터를 해당 버튼에 올릴 때마다 버튼에 대응하는 배너가 화면에 나타난다. 만일 해당 배너로 이동하고자 할 경우에는 해당 버튼을 클릭하거나 초점이 주어진 상태에서 Enter 키를 누르면

된다.

이 스크롤 배너를 접근성 있게 만들기 위해서는 배너를 이미지로 구성한다. 만일 스크롤 배너용 배너를 무비클립이나 버튼으로 만들면 각각의 배너로 초점이 이동하므로 주의하여야 한다. 스크롤 배너를 4개의 이미지로 구성한다고 하면, 이들 4개의 이미지 파일은 대응되는 4개의 버튼에 연결되어야 한다. 따라서 이 스크롤 배너에서는 버튼의 접근성이 매우 중요하다.

버튼은 화면에 나타나는 순서와 동일한 순서로 초점이 주어지도록 탭 인덱스를 설정하여야 한다. 앞서서도 이야기 하였지만 액세스 가능성 패널에서 설정한 탭 인덱스는 Tab 키에 의한 이동 순서를 의미한다.

화면 낭독 프로그램은 초점이 주어진 버튼에 대한 대체 텍스트를 읽어주므로 모든 버튼은 액세스 가능성 패널의 이름란 또는 설명란에 적절한 대체 텍스트를 제공하여야 한다. 이러한 경우에 Flash에서는 배너로 사용되는 이미지에는 대체 텍스트를 제공할 수 없으며, 또한 제공할 필요도 없다.

자동 스크롤 배너와 같이 4개의 이미지를 아래 <그림 IV - 19>와 같이 4개의 버튼과 배너 창으로 이루어진 스크롤 배너 구성 방법을 살펴보자.



그림 IV - 19. 4개 버튼과 배너창으로
구성된 스크롤 배너 예제

우선 4개의 이미지는 대체 텍스트를 제공하지 않는다. 대신 4개의 버튼에 대한 액세스 가능성 패널을 이용하여 대체 텍스트를 제공하여야 한다. 각 버튼별로 액세스 가능성 패널의 설정 방법은 다음 <표 IV - 3>과 같다.

표 IV - 3. 스크롤 배너의 액세스 가능성 패널 설정 구성

| 버튼 번호 | 배너 이미지 파일이름 | 하이퍼텍스트 링크 | 액세스 가능성 패널 설정 | | | |
|----------|-----------------|--------------|---------------------------|-------------|----------------------------------|------------|
| | | | 객체를 액세스 가능하게 만들기 | 이름 | 설명 | Tab 인덱스 |
| 1 | 질의응답.jpg | http:// ... | 체크 | 질의응답 | 질의분야를 선택하 신후... 공간입니 다. | 1 |
| 2 | 나의질의조회 .jpg | http:// ... | 체크 | 나의질의 조회 | 질의하신 질문과 ... 공간입니다. | 2 |
| 3 | 고객센터FAQ .jpg | http:// ... | 체크 | 고객센터 FAQ | 질의를 신청하기 전... 확인하실 수 있습니다. | 3 |
| 4 | 고객상담안내 .jpg | http:// ... | 체크 | 고객상담 안내 | 고객의 효과적인 상담을... 안내페이 지입니다. | 4 |

5.2.3 스크롤 버튼을 이용한 배너

스크롤 버튼을 이용한 배너는 <5.2.2 버튼을 이용한 스크롤 배너>의 경우와 달리 배너의 이동을 위하여 좌우 이동 버튼을 제공하는 방법이다. 또한 좌우 이동 버튼을 클릭할 때마다 배너가 반 시계방향 또는 시계방향으로 회전하면서 나타난다. 만일 어떤 배너로 이동하고자 할 경우에는 해당 배너 상에 마우스 포인터를 위치시키고 마우스 버튼을 클릭하면 된다.

이러한 스크롤 배너를 Tab 키를 이용하여 조작하는 경우에는 다음과 같은 순서로 초점이 이동한다. 처음에는 좌측 버튼에 초점이 제공되며, 또한 Tab 키를 누르면 첫 번째 배너로 초점이 이동한다. 이후로 Tab 키를 누를 때마다 계속해서 다음 배너로 초점이 이동한다. 마지막 배너에서 Tab키를 누르면 배너를 빠져나와 초점이 우측 이동 버튼으로 이동한다.

그런데 Tab 키를 이용하여 초점을 이동하는 경우에 좌우 이동 버튼으로 초점이 이동하는 과정은 불필요한 과정이다. 따라서 스크롤 버튼을 이용한 배너를 구현할 때에 키보드 사용자는 좌우 이동 버튼으로 초점을 이동할 수 없으나, 마우스 사용자의 경우에는 좌우 이동 버튼을 사용할 수 있도록 구현한다면 접근성이 매우 좋은 콘텐츠를 제작할 수 있다.

Flash에서 배너는 무비클립 또는 버튼으로 구성하고, 각 배너에 링크를 제공한다. 또한 대체 텍스트를 제공하여 이동할 주소가 어디인지를 화면 낭독 프로그램으로 읽어줄 수 있도록 한다. 이 과정은 <5.2.1 자동 스크롤 배너>와 같다.

좌우 이동 버튼은 버튼의 모습을 가진 이미지로 표현하여 Tab 키로는 초점이 이동할 수 없도록 한다. 그러나 마우스 사용자는 좌우 이동 버튼 이미지를 실제 버튼으로 사용할 수 있도록 프로그래밍한다. 즉 좌우 이동

버튼 이미지 위에 마우스 포인터를 위치시키면 마우스 포인터가 손바닥 모양으로 변경되도록 설정한다. 또한 이 상태에서 마우스 클릭을 하면 실제로 좌우 이동 버튼을 클릭한 것과 같이 처리되도록 ActionScript를 작성한다.

이렇게 구현된 좌우 이동 버튼은 키보드 사용자에게는 초점이 이동하지 않는 화면 구성요소이나, 마우스 사용자는 좌우 이동 버튼으로 사용할 수 있으므로 모두가 편리하게 사용할 수 있는 스크롤 배너를 만들 수 있다. <그림 IV - 20>은 마우스 사용자나 키보드 사용자가 모두 사용할 수 있도록 구현한 스크롤 버튼을 이용한 배너의 모습이다.



그림 IV - 20. 스크롤 버튼을 이용한 배너의 예제

이미지 위에서 마우스 포인터의 모양을 변경하는 방법과 마우스를 클릭하였을 때 지정된 동작을 수행하도록 ActionScript를 작성하는 과정은 Flash 제작 기법에 관한 사항이므로 이 문서에서는 설명을 생략한다.

웹 접근성을 고려한 신기술 콘텐츠 제작기법

- JavaScript, Flex, Flash를 중심으로 -

서울특별시 강서구 공항로 188

발행일 : 2008. 12.

발행처 : 한국정보문화진흥원

전 화 : (02) 3660 - 2576 FAX : (02) 3660 - 2579

URL : <http://www.kado.or.kr>

1. 본 보고서의 내용은 한국정보문화진흥원의 공식적인 견해가 아님을 밝혀둡니다.
2. 본 보고서의 내용을 인용할 때에는 반드시 한국정보문화진흥원을 밝혀주시기 바랍니다.
3. 본 보고서가 작성된 시점에 스크린리더가 지원하지 못하는 기능은 개발사에 의해 추후 보완 및 개선 될 수 있습니다.