C

하위 호환성

플래시에서는 .swf 파일을 이전 버전 플레이어와 호환되는 형식으로 저장할 수 있다. 이 책을 읽는 대부분의 독자들은 플래시 5 플러그인의 최신 버전을 가지고 있을 것이다. 하지만 경우에 따라 플래시 4 플러그인에서 실행시킬 수 있는 플래시 무비를 만들어야 하는 경우도 있다. 플래시 5 전용 무비를 만드는 경우에도 이 부분을 읽어보면 더 이상 쓰이지 않는 문법이나 속성, 메소드를 제외하고 플래시 5에서 권장하는 방법을 익히는 데 도움이 될 것이다. 다양한 플래시 플레이어 버전에 대한 통계 자료는 다음 URL에서 구할 수 있다.

http://www.macromedia.com/software/flash/survey/whitepaper

플래시 4에서 사용할 액션스크립트 코드를 작성할 때는 〔표 C-1〕에 수록된 더 이상 쓰이지 않는(구식 방법이긴 하지만 하위 호환성을 위해 아직도 지원하는) 방법을 이용해야 한다. 매크로미디어의 플래시 액션스크립트 레퍼런스 가이드의 'Writing Scripts with ActionScript'에 있는 'Using Flash 5 to Create Flash 4 Content' 부분도 읽어보면 도움이 된다.

플래시 4 플레이어에서 스크립트를 실행하려면 .swf 파일을 만들기 전에 File → Publish Settings의 Flash 탭에서 Flash 4를 선택해야 한다. 플래시 5 이후 버전의 .swf 파일을 플래시 4 플레이어에서 재생하면 코드가 실행되지 않는다.

〔표 C-1〕에는 플래시 4 액션스크립트와 플래시 5 액션스크립트 사이의 주된 차이점과 하위 호환성 문제를 정리해서 수록하였다.

〔표 C-1〕 하위 호환성 문제

주제	설명
변수 생성	플래시 4의 set 함수가 var 선언문으로 바뀌었다. 동적으로 정해진 이름을 가지는 변수를 만들고 싶다면 eva( )을 사용하면 되지만, 될 수 있으면 배열을 이용하여 데이터를 관리하는 것이 좋다('11장. 배열' 참조).
변수와 타임라인 레퍼런스	플래시 4 형식의 슬래시-콜론 구조(/square:area) 대신 점 표기법 (square.area)을 사용한다([표 2-1] 참조).
문자열 비교 연산	플래시 4의 문자열 비교 연산자인 eq, ne, ge, gt, le, lt 대신 ==, !=, >=, >, <=, <를 사용한다([표 4-2] 참조).
문자열 병합	플래시 5 이후 버전에서 플래시 4용 무비를 만들 때는 플래시 4의 & 연산자 대신 add 연산자를 사용해야 한다. 플래시 5용 무비를 만들 때는 + 연산자를 이용하면 된다([표 4-2] 참조).
문자열의 길이	플래시 4의 length( ) 함수(예: length(myString)) 대신 length 속성(예: myString.length)을 사용한다([표 4-2] 참조).
문자열 추출	플래시 4의 substring( ) 함수(예: substring(myString, 1, 3)) 대신 substring( ), substr( ), slice( ) 메소드를 사용한다. 플래시 4와 플래시 5에서 substring( )의 기능이 다르다는 점에 주의하자([표 4-2] 참조).
문자 코드 포인트 함수	플래시의 chr( ) 및 mbchr( ) 함수(코드 포인트로부터 문자를 만드는 함수) 대신 String.fromCharCode( ) 클래스 메소드를 사용한다. 플래시의 ord( )와 mbord( ) 함수(문자의 코드 포인터를 알아내기 위한 함수) 대신 String.charCodeAt( ) 메소드를 사용한다([표 4-2] 참조).
데이터형 변환	혼동을 피하기 위해 플래시 4 파일을 플래시 5에서 불러오면 다음과 같은 연산자의 피연산자로 쓰이는 숫자는 자동으로 Number( ) 함수로 감싸진다([표 3-5] 참조).
ifFrameLoaded 선언문	플래시 3의 ifFrameLoaded 선언문은 더 이상 쓰이지 않는다. 프리로더를 만들 때는 MovieClip의 _totalframes와 _framesloaded 속성을 이용한다.

주제	설명
무한 루프	플래시 4에서는 최대 200,000번까지 루프를 반복할 수 있다. 플래시 5에서는 15초까지 루프를 돌릴 수 있으며, 15초가 지나면 사용자에게 무비가 응답을 하지 않는다는 경고 메시지를 출력한다('8장. 순환문'의 '최대 반복 횟수' 부분 참조).
서브루틴과 함수	플래시 4에서는 레이블이 있는 프레임에 코드 블록을 추가하여 서브루틴을 만들고, call( ) 선언문을 이용하여 그 서브루틴을 호출할 수 있다. 플래시 5에서는 서브루틴 대신 함수를 사용한다.
클립 이벤트	플래시 4에서는 [표 10-1]에 수록된 버튼 이벤트(on( )으로 시작하는 함수)만 지원한다. 따라서 플래시 4 플레이어에서는 클립 이벤트(onClipEvent( ) 등)는 사용할 수 없다.
키 캡처	플래시 4에서 키의 상태를 알아내는 방법에는 keyPress밖에 없었다. 플래시 5에서는 Key 객체와 keyDown, keyUp 무비 클립 이벤트를 조합하여 키보드 이벤트를 더 다양한 방법으로 처리할 수 있다.
Tell Target	플래시 4의 Tell Target(원격 무비 클립을 제어하기 위한 구문) 구문은 더 이상 쓰이지 않고 대신 점 표기법과 with 선언문을 통해 속성 및 메소드를 사용한다('13장. 무비 클립' 참조).
Get Property	무비 클립 속성을 액세스할 때 플래시 4의 Get Property 명령을 더 이상 사용하지 않아도 된다. 대신 점 연산자를 사용한다(13장 참조).
int	플래시 4의 int( ) 함수(소수를 정수로 변환하는 함수) 대신 Math.floor( ), Math.ceil( ), Math.round( )를 사용한다.
난수 생성기	플래시 4의 random( ) 함수(난수를 생성하는 함수) 대신 Math.random( ) 함수를 사용한다.
toggleHighQuality	플래시 4의 toggleHighQuality 함수(플레이어의 렌더링 화질을 설정하기 위한 함수) 대신 _quality 전역 속성을 사용한다.
_highquality	플래시 4의 _highquality 속성 대신 _quality 전역 속성을 사용한다.
플래시 4의 Math 객체	Math 객체의 함수와 속성(예: Math.cos( ), Math.PI)은 플래시 4 플레이어 지원에서 직접 지원되지 않는다. 하지만 플래시 4 형식으로 무비를 저장할 때 근사값이 대신 사용된다.
loadMovie와 loadMovieNum	플래시 3의 loadMovie( )에서 특정 레벨로 무비를 로딩하는 기능 대신 플래시 5에서는 loadMovieNum( ) 함수를 사용한다. 플래시 4에서 대상 무비 클립으로 loadMovie( ) 함수를 실행하는 기능은 플래시 5의 loadMovie( )에서도 여전히 사용할 수 있다.
인쇄	플래시 5에서는 print( ) 함수를 이용하여 직접 인쇄 기능을 지원하며, 이 기능은 플래시 4의 빌드 20부터 변형된 Get URL 액션 형태로 지원되었다.
객체와 클래스 지원	플래시 4에서는 플래시 5의 내장 객체와 클래스를 전혀 지원하지 않는다.

플래시 5 플레이어, 빌드 41에서 업데이트된 부분

아래 목록에는 플래시 5 플레이어의 빌드 41(넷스케이프)과 42(인터넷 익스플로러)에서 바뀐 몇 가지 사항을 요약해 놓은 것이다(그 전에 발표되었던 것은 플래시 5 저작 도구와 함께 출시된 플래시 5 빌드 30이었다).

- 텍스트 필드에서 쓰이는 I자 모양의 막대가 텍스트 색과 같은 색으로 표시된다.
- 테이블 셀에 들어있는 무비 때문에 인터넷 익스플로러 5.5가 다운되는 문제를 해결하였다.
- 필드 내용을 수정하더라도 텍스트 필드의 scroll 위치가 리셋되지 않는다.
- 폰트가 내장된 무비의 텍스트 필드를 스크롤했을 때 경계선 밖으로 삐져나오는 문제를 해결하였다.
- XML.contentType 속성이 추가되었다.
- XML.ignoreWhite 속성이 추가되었다.
- XML 소스를 파싱할 때 텍스트 노드에 &, ‘, “, <, > 기호가 나오면 자동으로 &, ', ", <, >로 변환된다. XML 객체를 문자열로 바꿀 때는 그러한 문자들을 다시 원래 기호로 변환하기 때문에, 플래시에서는 이러한 변화를 크게 느끼지 못한다. 하지만 이 XML 소스를 서버로 전송하면 변환된 문자열 형태로 전송된다.
- Math.random()에서 1을 리턴하는 문제가 해결되었다. 이 메소드에서 리턴하는 최대값은 0.999이다.
- 전반적인 속도가 개선되었다(특히 윈도우 98).

무비 클립 제어

13장에서는 플래시 5를 기준으로 클립을 제어하는 방법에 대해 배웠다. 여기서는 플래시 4를 기준으로 무비를 제어하는 방법을 생각해 보자.

플래시 5가 나오기 전에는 무비 클립을 제어할 때 Movie Clip Actions라는 액션을 이용하였다. "eye라는 클립을 재생시켜라"와 같은 명령을 전달하려면 다음과 같은 코드를 사용해야 한다.

```
Begin Tell Target ("eyes")
  Play
End Tell Target
```

하지만 플래시 5 이후로는 내장 메소드를 통해 무비를 더 직접적인 방법으로 제어할 수 있다. 예를 들면 다음과 같다.

```
eyes.play();
```

이와 마찬가지로 플래시 5가 나오기 전에는 무비 클립에 내장된 속성을 액세스하려면, 다음과 같은 식으로 속성을 구하는 명령어와 속성을 설정하는 명령어를 직접 호출해야 했다.

```
Get Property ("ball", _width)
Set Property ("ball", X Scale) = 90
```

플래시 5부터는 다른 객체의 속성을 액세스하는 방법과 같은 식으로 점 연산자를 이용하여 무비 클립의 속성을 구하고 설정할 수 있다.

```
ball._width
ball._xscale = 90;
```

플래시 5가 나오기 전에는 무비 클립에 있는 변수를 액세스하려면, 콜론을 이용하여 클립 이름과 변수 이름을 분리해야 했다.

```
Set Variable: "x" = myClip:myVariable
```

하지만 플래시 5부터는 무비 클립에 있는 변수는 그 클립 객체의 속성으로 간주되기 때문에, 점 연산자를 이용하여 그 변수 값을 구하고 설정할 수 있다.

```
myClip.myVariable = 14;
x = myClip.myVariable;
```

마지막으로 플래시 5가 나오기 전에는 여러 단계로 중첩된 무비 클립을 액세스할 때는 다음과 같이 디렉토리 트리를 사용하는 것처럼 슬래시와 점을 이용해야 했다.

```
clipA/clipB/clipC
../../../clipC
```

플래시 5부터는 무비 클립도 객체와 비슷한 데이터로 간주되기 때문에 어떤 클립을 다른 클립의 속성 형태로 저장할 수 있다. 따라서 점 연산자를 이용하여 중첩된 클립을 액세스할 수 있으며, 그 클립을 포함하는 클립을 액세스할 때는 _parent 속성을 이용하면 된다.

```
clipA.clipB.clipC;
_parent._parent._parent.clipC;
```