helion.pl
KSIĘGARNIA INTERNETOWA · WYDAWNICTWO

IDŹ DO
PRZYKŁADOWY ROZDZIAŁ
SPIS TREŚCI

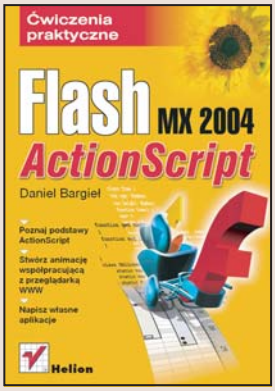
KATALOG KSIĄŻEK
KATALOG ONLINE
ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK
DODAJ DO KOSZYKA

CENNIK I INFORMACJE
ZAMÓW INFORMACJE O NOWOŚCIACH
ZAMÓW CENNIK

CZYTELNIA
FRAGMENTY KSIĄŻEK ONLINE

Flash MX 2004 ActionScript. Ćwiczenia praktyczne



Autor: Daniel Bargieł
ISBN: 83-246-0005-1
Format: B5, stron: 104
Przykłady na ftp: 2323 kB

Odkryj kolejne zastosowania Flasha MX

Po opanowaniu narzędzi graficznych oferowanych przez Flasha MX 2004 czas na następny krok – poznanie potęgi języka ActionScript. ActionScript to obiektowy język programowania, dzięki któremu Twoje animacje zyskają setki nowych możliwości i zastosowań. Wykorzystując ActionScript możesz kontrolować nie tylko animację, ale wszystkie zawarte w niej obiekty. Możliwości tego języka pozwalają również na stworzenie aplikacji, która wymienia dane z przeglądarką, bazą danych lub innym serwerem, interaktywnych formularzy, rozwijanych menu i wskaźników postępu ładowania animacji. Dopiero zastosowanie ActionScript pozwala na wykorzystanie wszystkich możliwości Flasha.

Książka „Flash MX 2004 ActionScript. Ćwiczenia praktyczne” to wprowadzenie do programowania w języku ActionScript. Wykonując zawarte w niej ćwiczenia poznasz podstawowe zagadnienia związane ze stosowaniem tego języka w swoich aplikacjach. Nauczysz się dołączać kod ActionScript do elementów animacji, korzystać z klas i obiektów, tworzyć mechanizmy komunikacji animacji z przeglądarką WWW i przesyłać strumieniowo obraz wideo za pomocą Flash Communication Server MX.

- Umieszczanie kodu ActionScript w obiektach Flasha
- Zasady tworzenia kodu sterowanego zdarzeniami
- Obiekty i klasy języka ActionScript
- Wymiana danych pomiędzy animacją a przeglądarką WWW
- Integracja aplikacji z Flash Communication Server MX
- Tworzenie prostego chata w ActionScript

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Wstęp **5**
Co daje ActionScript 5
Gdzie dołączać skrypty 6
Prosta interaktywna animacja 7

Rozdział 1. Podstawy ActionScript **11**
Interakcje w animacjach — przyciski i klipy filmowe 11
Detektor on — onMouseEvent 11
Detektor onClipEvent 16
Kontrola odtwarzania animacji 18
Programowanie w ActionScript 21
Zmienne w ActionScript 21
Tworzenie funkcji w ActionScript 23
Instrukcje warunkowe 25
Debugowanie skryptów 27

Rozdział 2. Programowanie zdarzeniowe w ActionScript **31**
Podstawy programowania zdarzeniowego 31
Obiekty nasłuchujące zdarzeń 34

Rozdział 3. Klasy i obiekty ActionScript **43**
Obiekty w ActionScript 2.0 43
Klasy obiektów 44
Najciekawsze klasy wbudowane w ActionScript 48
Obiekt tablicowy — Array 48
Obiekt koloru — Color 52
Obiekt daty — Date 53
Obiekt matematyczny — Math 56
Obiekt ciągu tekstowego — klasa String 56
Menu podręczne — klasa ContextMenu i ContextMenuItem 59

Rozdział 4. Komunikacja animacji SWF ze środowiskiem zewnętrznym **65**
Komunikacja SWF z przeglądarką WWW 65
Komunikacja przeglądarki WWW z animacją SWF 69
Pobieranie danych z plików 72

Rozdział 5. Transmisja strumieni wideo — Flash Communication Server MX 75
Co potrafi Flash Player 7 75
Flash Communication Server MX 79
Transmisja strumieni wideo 81

Rozdział 6. Aplikacja chat 91
Projekt aplikacji 91
Narzędzia administracyjne 99

Rozdział 1.

Podstawy ActionScript

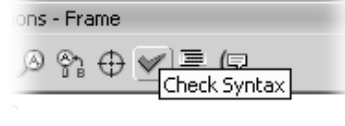
Interakcje w animacjach — przyciski i klipy filmowe

Czytelnik dowiedział się już, że aby dodawać skrypty do takich obiektów jak klip filmowy lub przycisk, niezbędne są detektory zdarzeń. Czym jednak tak naprawdę są owe detektory? Jakie zdarzenia mogą przechwytywać? Czym różni się detektor `on` od detektora `onClipEvent`? Na te wszystkie pytania odpowiemy sobie w tym rozdziale. Pokażemy też kilka ćwiczeń, aby ułatwić Czytelnikowi opanowanie sposobu dodawania skryptów do odnośników.

Detektor on — onMouseEvent

Detektor `on` jest funkcją, którą trzeba dodać do skryptu, aby umieścić go w przycisku. Jeśli użytkownik spróbuje dodać skrypt do przycisku bez detektora `on`, to przy próbie opublikowania animacji Flash wyświetli komunikat o błędzie a dodany skrypt nie będzie działał.

Aby sprawdzić poprawność skryptu, w trakcie wpisywania kodu można wcisnąć kombinację klawiszy *Ctrl+T* lub w panelu akcji kliknąć przycisk *Check syntax* (rysunek 1.1).



Rysunek 1.1.
Przycisk Check Syntax w panelu akcji pozwala na sprawdzenie, czy wprowadzony kod jest formalnie poprawny

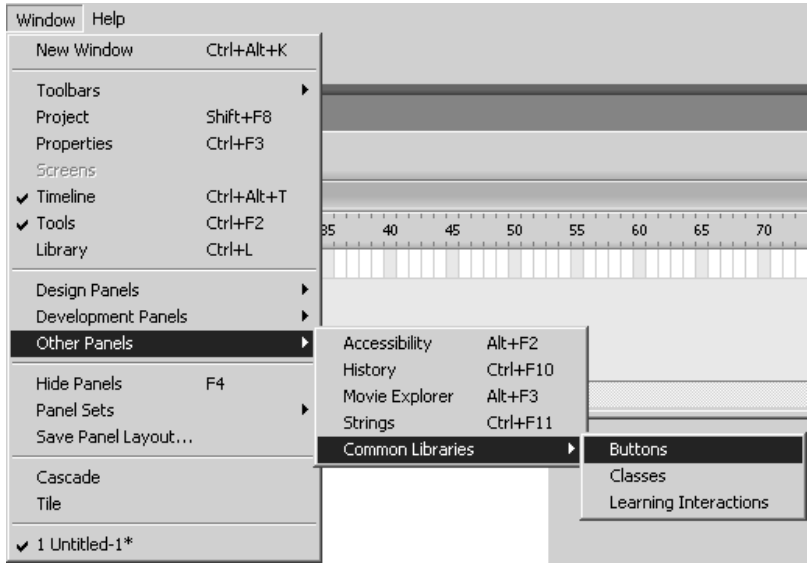
Jeśli skrypt jest poprawny, Flash wyświetli okno z komunikatem *This script contains no errors* (*Ten skrypt nie zawiera błędów*). W przeciwnym przypadku zostanie wyświetlony panel *Output* z dokładnym opisem błędu oraz miejscem jego wystąpienia.

Gdy podczas lektury tej książki Czytelnik spotka się z określeniem w stylu: *dodaj skrypt do przycisku* lub *dodaj skrypt do klipu filmowego*, oznacza to, że należy dodać ów skrypt do odnośnika tego obiektu, znajdującego się w obszarze roboczym, a nie do samego symbolu. Częstym błędem początkujących użytkowników Flasha jest próba dodawania skryptów, które mają być wykonywane po kliknięciu przycisku, do samego symbolu owego przycisku, np. do ujęcia *Down*. Nie tędy droga. Takie podejście jest nieefektywne, ponieważ jeśli np. użytkownik zmieni zdanie i tak dodany skrypt będzie miał być wykonywany po naprowadzeniu wskaźnika myszy na przycisk, konieczne będzie przeniesienie całego kodu skryptu do ujęcia *Over*. Drugą niedogodnością takiego sposobu pracy jest fakt, że wszystkie odnośniki, jakie zostaną umieszczone w obszarze roboczym, będą zachowywały się tak samo.

Ćwiczenie 1.1.

Detektor on posiada argument, który decyduje, kiedy skrypt zostanie wykonany. Poniższe, proste ćwiczenie pozwoli na dokładne przeanalizowanie działania tego detektora.

1. Najpierw należy utworzyć przycisk. Do odnośnika tego przycisku zostaną dołączone akcje. Symbol można utworzyć ręcznie lub skorzystać z jednego z gotowych przycisków, które można znaleźć wybierając z menu *Window* polecenie *Other panels*, a następnie *Common Libraries* i *Buttons* (rysunek 1.2).

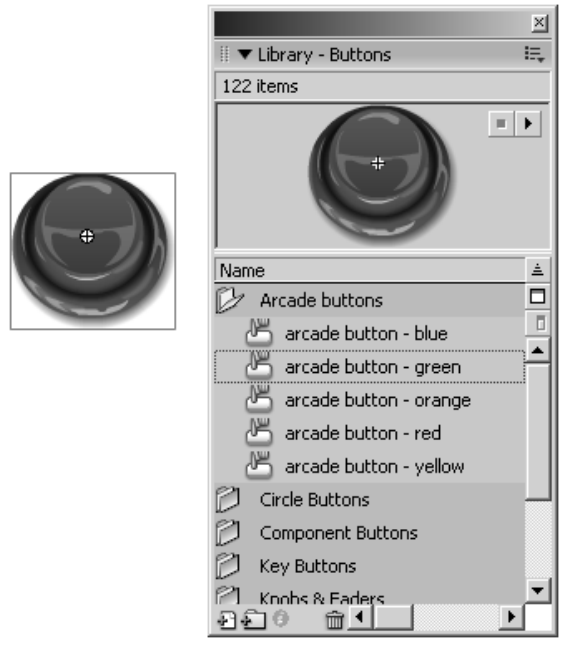


Rysunek 1.2. *W menu Common Libraries znajduje się wiele gotowych obiektów, takich jak np. gotowe przyciski*

Flash wyświetli okno biblioteki, w której symbole przycisków są pogrupowane w katalogach (rysunek 1.3).

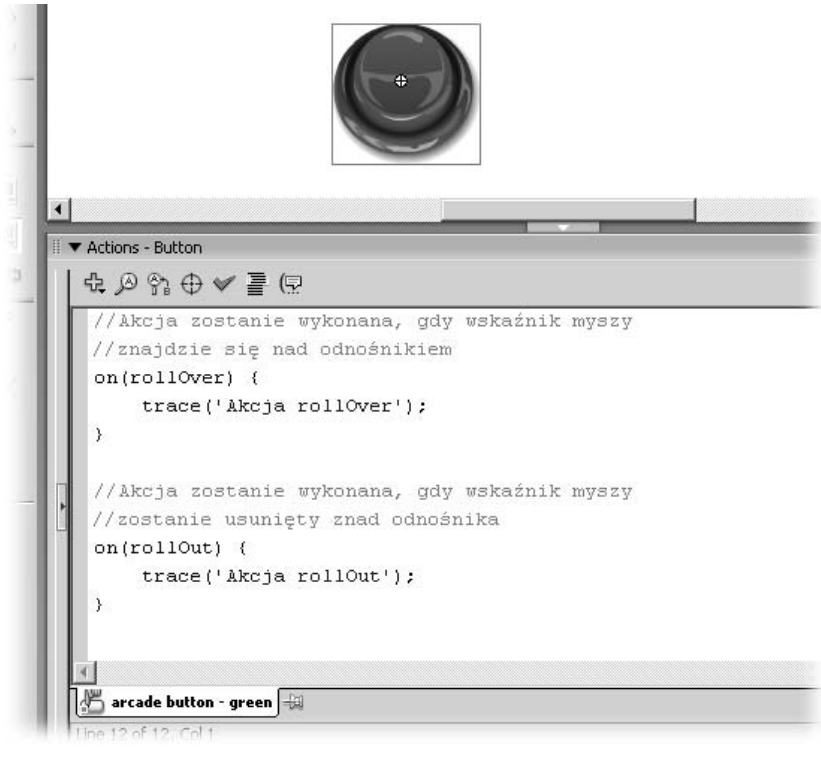
2. Umieść dowolny z przycisków w obszarze roboczym. W tym celu przeciągnij go z otwartego panelu biblioteki (rysunek 1.3).
3. Flash automatycznie skopiuje symbol do biblioteki projektu i umieści jego odnośnik w obszarze roboczym.

Rysunek 1.3.
Gotowe przyciski do wykorzystania w animacji



4. Po wybraniu nowego odnośnika do przycisku otwórz panel akcji i do okna programu wpisz kod przedstawiony na listingu 1.1 (rysunek 1.4).

Rysunek 1.4.
Kod skryptu z listingu 1.1 w panelu akcji



Listing 1.1. *Kod dołączony do odnośnika przycisku*

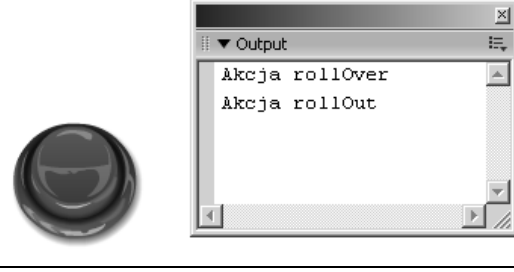
```
//Akcja zostanie wykonana, gdy wskaźnik myszy
//znajdzie się nad odnośnikiem
on(rollOver) {
    trace('Akcja rollOver');
}

//Akcja zostanie wykonana, gdy wskaźnik myszy
//zostanie usunięty znad odnośnika
on(rollOut) {
    trace('Akcja rollOut');
}
```

Podczas wpisywania kodu do panelu akcji Flash podpowiada programiście, jakie możliwości pozostają do wykorzystania. W przypadku skryptu z listingu 1.1 po wpisaniu ciągu znaków `on(` Flash powinien automatycznie wyświetlić listę możliwych do wyboru zdarzeń, jakie może przechwycić detektor `on` (rysunek 1.5).

Rysunek 1.5. *W trakcie tworzenia skryptu programista może korzystać z mechanizmu uzupełniania kodu*

5. Opublikuj teraz animację. W tym celu wybierz z menu *Control* polecenie *Test Movie* lub wciśnij kombinację klawiszy *Ctrl+Enter*.

Jeżeli w trakcie odtwarzania animacji użytkownik kilkukrotnie przesunie wskaźnik myszy nad odnośnik do przycisku, to w oknie *Output* pojawią się komunikaty informujące o wystąpieniu akcji `rollOver` oraz `rollOut` (rysunek 1.6).



Rysunek 1.6. *Efekt działania skryptu z listingu 1.1*

Dodając jakikolwiek skrypt do odnośnika do przycisku należy zdawać sobie sprawę, że przycisk reaguje tylko i wyłącznie na zdarzenia. W przypadku przycisku tym zdarzeniem może być przesunięcie wskaźnika myszy nad przycisk, kliknięcie przycisku itp.

W dalszej części tej książki często będziemy posługiwać się słowem *przycisk*, które będzie oznaczało odnośnik do symbolu przycisku (*Button*), chyba że w tekście wyraźnie zostanie zaznaczone inaczej.

Tak więc w skrypcie dodawanym do przycisku trzeba umieszczać detektory zdarzeń, które noszą nazwę `on`. Jeden detektor zdarzeń może przechwycić jedno zdarzenie, zatem jeżeli przycisk ma reagować na kilka zdarzeń, należy umieścić kilka detektorów `on`.

O tym, jakie zdarzenie przechwytuje detektor `on`, decyduje parametr podany jako argument detektora. Na listingu 1.1 widać, że utworzono dwa detektory, z których pierwszy przechwytywał zdarzenie `rollOver` a drugi — zdarzenie `rollOut`.

Zdarzenie `rollOver` zachodzi wtedy, gdy wskaźnik myszy znajdzie się nad przyciskiem, do którego przypisano to zdarzenie, natomiast zdarzenie `rollOut` zachodzi, gdy wskaźnik myszy opuści obszar przycisku, do którego przypisano to zdarzenie.

Gdy zajdzie zdarzenie, którego oczekuje określony detektor `on`, wtedy zostanie wykonany kod zawarty między znakami { oraz }. W przypadku pokazanym na listingu 1.1 jest to funkcja `trace()`, której zadaniem jest wyświetlenie tekstu w panelu *Output*.

Funkcja `trace()` działa tylko i wyłącznie wtedy, gdy animacja jest publikowana w środowisku Flasha, ponieważ jedynie tam istnieje okno *Output*, do którego funkcja wysyła wyniki.

W ramach podsumowania poniżej przedstawiono zdarzenia, których obsługę zapewnia detektor `on` dodany do przycisku:

- *Press* — `on(press) {...}` — kod zawarty wewnątrz bloku detektora `on` jest wykonywany w razie wciśnięcia lewego przycisku myszy, gdy jej wskaźnik znajduje się nad przyciskiem;
- *Release* — `on(release) {...}` — kod zawarty wewnątrz bloku detektora `on` jest wykonywany po zwolnieniu lewego przycisku myszy, gdy jej wskaźnik znajduje się nad przyciskiem;
- *Release Outside* — `on(releaseOutside) {...}` — kod zawarty wewnątrz bloku detektora `on` jest wykonywany po zwolnieniu lewego przycisku myszy, gdy jej wskaźnik znajduje się poza obszarem przycisku, pod warunkiem jednakże, że wcześniej użytkownik kliknął ten przycisk;
- *Key Press* — `on(keyPress "klawisz") {...}` — kod zawarty wewnątrz bloku detektora `on` jest wykonywany po wciśnięciu na klawiaturze klawisza *klawisz*. Położenie wskaźnika myszy ani stan jej przycisków nie odgrywają w tym przypadku żadnej roli;
- *Roll Over* — `on(rollOver) {...}` — kod zawarty wewnątrz bloku detektora `on` jest wykonywany, gdy wskaźnik myszy znajdzie się w obszarze przycisku;
- *Roll Out* — `on(rollOut) {...}` — kod zawarty wewnątrz bloku detektora `on` jest wykonywany, gdy wskaźnik myszy opuści obszar przycisku;
- *Drag Over* — `on(dragOver) {...}` — technika *Drag&Drop*. Kod zawarty wewnątrz bloku detektora `on` jest wykonywany, gdy użytkownik przesunie wskaźnik myszy nad przycisk, wciśnie jej lewy przycisk i następnie — wciąż przytrzymując wciśnięty lewy przycisk myszy — cofnie jej wskaźnik z obszaru przycisku a następnie tam powróci;

- *Drag Out* — `on(dragOut) {...}` — technika *Drag&Drop*. Kod zawarty wewnątrz bloku detektora `on` jest wykonywany, gdy użytkownik przesunie wskaźnik myszy nad przycisk, wciśnie jej lewy przycisk a następnie — wciąż przytrzymując wciśnięty lewy przycisk myszy — cofnie jej wskaźnik z obszar przycisku.

Detektor onClipEvent

Drugim z detektorów zdarzeń dostępnych w ActionScript jest detektor `onClipEvent`, który jest dodawany do klipu filmowego. Działanie detektora `onClipEvent` jest dość podobne do działania detektora `on` ale istnieją pewne różnice. Aby Czytelnik łatwiej je zrozumiał, proponujemy wykonanie prostego ćwiczenia.

Ćwiczenie 1.2.

Poniższe ćwiczenie pozwoli zrozumieć działanie detektora onClipEvent:

1. Utwórz dowolny kształt, a następnie przekształć go do symbolu *Movie Clip*.
2. Wybierz odnośnik, otwórz panel akcji a następnie w oknie kodu wpisz skrypt z listingu 1.2.

Listing 1.2. *Skrypt dodający klip filmowy do wskaźnika myszy*

```
//Przechwycenie zdarzenia nastąpi po wciśnięciu lewego
//przycisku myszy. Położenie myszy nie ma znaczenia
onClipEvent(mouseDown) {
    if (hitTest(_root._xmouse, _root._ymouse, true)) {
        startDrag();
    }
}
```

Zdarzenie `mouseDown` zachodzi, jeśli użytkownik oglądający animację SWF wciśnie lewy przycisk myszy, podczas gdy jej wskaźnik znajduje się w oknie animacji. W tym przypadku położenie wskaźnika myszy nie ma znaczenia, zatem nie można stwierdzić, czy użytkownik kliknął klip filmowy, do którego przypisano kod wykrywający zdarzenie.

Dlatego też należy zastosować funkcję `hitTest()`, która sprawdza, czy w momencie wciśnięcia lewego przycisku myszy jej wskaźnik (którego pozycja jest reprezentowana poprzez współrzędne `_xmouse` i `_ymouse`) znajdował się nad klipem filmowym.

Jeżeli wskaźnik myszy w momencie wciśnięcia jej lewego przycisku znajduje się nad klipem filmowym, do którego przypisano skrypt, to następuje wykonanie funkcji `startDrag()`, która powoduje pewnego rodzaju przyłączenie klipu filmowego do wskaźnika myszy, dzięki czemu możemy ten odnośnik przesuwać w oknie animacji.

3. Pod kodem z listingu 1.2 umieść kod z listingu 1.3.

Listing 1.3. *Skrypt odświeżający grafikę w obszarze roboczym*

```
//Przechwycenie zdarzenia nastąpi, gdy wskaźnik myszy będzie poruszać się
//w obszarze roboczym. Położenie tego wskaźnika nie ma znaczenia
onClipEvent(mouseMove) {
    updateAfterEvent()
}
```

Zdarzenie `mouseMove` jest wykonywane za każdym razem, gdy animacja SWF wykryje ruch myszy (czyli w przypadku ciągłej pracy myszy zdarzenie to jest wykonywane bardzo często). Zawarta w bloku detektora `onClipEvent` funkcja `updateAfterEvent()` powoduje odświeżenie grafiki w obszarze roboczym niezależnie od aktualnie przyjętej prędkości animacji. Dzięki temu przesuwając odnośnik w obszarze roboczym możemy zaobserwować jego płynny ruch.

4. Ostatnim fragmentem skryptu, który należy dodać do klipu filmowego, jest kod z listingu 1.4.

Listing 1.4. *Kod powodujący odłączenie wskaźnika myszki od klipu filmowego*

```
//Przechwycenie zdarzenia następuje po zwolnieniu lewego przycisku
//myszy. Położenie tego wskaźnika nie ma znaczenia
onClipEvent(mouseUp) {
    stopDrag();
}
```

Zdarzenie `mouseUp` zachodzi po zwolnieniu wcześniej wciśniętego lewego przycisku myszy. Następuje wtedy wykonanie funkcji `stopDrag()`, która powoduje odłączenie klipu filmowego od wskaźnika myszy.

5. Teraz opublikuj animację, wciskając kombinację klawiszy *Ctrl+Enter*. Łatwo zauważyć, że za pomocą myszy można przesuwać klip filmowy znajdujący się w obszarze roboczym.

Na koniec warto jeszcze krótko scharakteryzować zdarzenia, na które reaguje detektor `onClipEvent`:

1. *Load* — `onClipEvent(load) {…}` — skrypt zawarty w bloku detektora zostanie wykonany, gdy klip w całości zostanie załadowany do pomięci operacyjnej komputera użytkownika;
2. *EnterFrame* — `onClipEvent(enterFrame) {...}` — skrypt umieszczony w bloku detektora zostanie wykonany za każdym razem, gdy zostanie wyświetlona nowa klatka animacji zawartej w klipie. Częstotliwość wykonywania skryptu zależy od prędkości odtwarzania animacji (liczba wyświetlanych klatek na sekundę);
3. *Unload* — `onClipEvent(unload) {...}` — skrypt zawarty w bloku detektora zostanie wykonany, gdy klip filmowy zostanie usunięty z pamięci komputera (np. po wywołaniu funkcji `removeMovieClip()`);
4. *Mouse down* — `onClipEvent(mouseDown) {...}` — skrypt zawarty w bloku wewnętrznym detektora zostanie wykonany, gdy użytkownik wciśnie lewy przycisk myszy, przy czym położenie jej wskaźnika nie ma żadnego znaczenia (może się znajdować w dowolnym miejscu, pod warunkiem, że aktywna jest aplikacja, odtwarzająca animację SWF);
5. *Mouse up* — `onClipEvent(mouseUp) {...}` — skrypt zawarty w bloku detektora zostanie wykonany, gdy użytkownik zwolni lewy przycisk myszy. Położenie wskaźnika myszy nie ma znaczenia;

6. *Mouse move* — `onClipEvent(mouseMove) {...}` — skrypt zawarty w bloku detektora zostanie wykonany, gdy użytkownik poruszy myszą;
7. *Key down* — `onClipEvent(keyDown){...}` — skrypt zawarty w bloku detektora zostanie wykonany, gdy użytkownik wciśnie dowolny klawisz na klawiaturze. Detektor sam w sobie nie zawiera informacji o tym, który klawisz został wciśnięty;
8. *Key up* — `onClipEvent(keyUp) {...}` — skrypt zawarty w bloku detektora zostanie wykonany, gdy użytkownik zwolni wcześniej wciśnięty klawisz klawiatury. Detektor nie przekazuje informacji o tym, który klawisz został zwolniony;
9. *Data* — `onClipEvent(data) {...}` — skrypt zawarty w bloku detektora zostanie wykonany, gdy do klipu filmowego zostaną załadowane zmienne z zewnętrznego źródła danych.