



ActionScript

*Wie ging das noch?
Code-Häppchen für den Alltag*

*Zusammengestellt
von Andreas Baumgart*

O'REILLY
www.oreilly.de

SUFFIXE FÜR CODEHINWEISE

Wichtige Endungen für Objektnamen zur Aktivierung der Codehinweise und für übersichtliche Skripten

```
meinClip_mc, meinButton_btn, meinText_txt, meinTextFormat_fmt, meinArray_array,  
meinString_str, meinXML_xml, meinDatum_date, meinSharedObject_so u.a.
```

CLIP-PFADANGABEN

```
_level0 // Unterster Film im Dateienstapel des Flash Players (Basisfilm)  
_root // Name der Hauptzeitleiste eines Films
```

Verschachtelte Clips steuern

```
_root.matriushkaGross_mc.matriushkaMittel_mc.visible = false;  
_root.matriushkaGross_mc.matriushkaMittel_mc.matriushkaKlein_mc.play();  
_parent._x = 300; // X-Position für matriushkaGross_mc aus matriushkaMittel_mc oder  
// für matriushkaMittel_mc aus matriushkaKlein_mc heraus bestimmen
```

Dateienstapel (_leveln)

```
loadMovieNum("matriushkas.swf", 1); // Film liegt auf _level1  
loadMovie("matriushkas.swf", _root.position_mc); // Mit globaler Funktion Film in Ziel-Clip laden. Matriushkas.swf liegt  
// auf dem Level der swf-Datei, in deren _root sie geladen wird.  
_root.position_mc.loadMovie("matriushkas.swf"); // Mit MovieClip-Methode Film in Ziel-Clip laden
```

VARIABLEN UND GÜLTIGKEITSBEREICH

Gültigkeitsbereich auf einer Zeitleiste

```
_root.alter = 22; // Variable auf der Hauptzeitleiste  
_root.matriushkaGross_mc.alter = 31; // Variable auf Clip-Zeitleiste
```

Gültigkeitsbereich auf allen Zeitleisten / globale Variable

```
_global.alter = 31; // Variable deklarieren und einen Wert zuweisen
```

Ohne _global von jeder Zeitleiste aus lesen

```
trace(alter);
```

EINFACHE SCHLEIFE

Eine for-Schleife mit Abbruchparameter

```
iMax = 100;  
for (i = 0; i < iMax; i++) {  
    trace("i: " + i);  
}
```

FUNKTION MIT PARAMETERN

Voraussetzung: Clip-Instanz kugel_mc auf der Bühne

```
function setzePosition(meinClip, xPos, yPos) { // Parameter für den Instanznamen des Clips,  
// Position auf der X-Achse und Position auf der Y-Achse  
    meinClip._x = xPos;  
    meinClip._y = yPos;  
}
```

Funktionsaufruf:

```
setzePosition(kugel_mc, 300, 300);
```

CLIPS DUPLIZIEREN

Duplizieren und Positionieren von Clips

Voraussetzung: Skript auf Bild1 der Hauptzeitleiste, Instanz ball1_mc auf der Bühne

```
ball1_mc.duplicateMovieClip("ball2_mc", 0); // Clip mit Objekt-Methode duplizieren. Parameter: (neuerName, tiefe)  
ball2_mc._x += 100;  
ball2_mc._y += 200;
```

ARRAYS ERZEUGEN

Position der Elemente beginnt bei 0

```
einkaufsListe = ["Orangen", "Bananen", "Äpfel"]; // Möglichkeit 1: erzeugt Array und weist es der Variablen als Wert zu  
einkaufsListe = new Array("Orangen", "Bananen", "Äpfel"); // Möglichkeit 2: erzeugt Array und weist es der Variablen als Wert zu  
einkaufsListe = []; // Erzeugt ein leeres Array  
einkaufsListe[3] = "Trauben"; // Fügt dem Array an Position 3 das Element Trauben hinzu
```

Abfragen:

```
trace(einkaufsListe[0]); // gibt "Orangen" aus  
trace(einkaufsListe[2]); // gibt "Äpfel" aus  
trace (einkaufsListe.length); // gibt die Anzahl der Elemente des Arrays aus
```

DYNAMISCH GENERIERTE CLIPS IN EIN ARRAY SCHREIBEN

Drei Varianten: Eval, Array-Zugriffsoperator und Returnwert

Voraussetzung für alle drei Varianten: Clip-Instanz meineKugel auf der Hauptbühne

Variante 1. Mit der eval-Funktion dynamisch erzeugte Clips in einem Array speichern; Eval gilt in MX als veraltet

```
var kugeln = new Array(); // Ein leeres Array erzeugen und in einer Variable speichern  
for (var i = 0; i < 10; i++) { // Schleife, um 10 Clips dynamisch zu erzeugen  
    duplicateMovieClip("meineKugel_mc", "neueKugel" + i, i); // Clips duplizieren mit globaler Funktion. Parameter: (ziel, neuerName,  
// tiefe), wobei ziel den zu duplizierenden Basisclip angibt.  
    kugeln[i] = eval("neueKugel" + i); // Die neuen Kugel-Clips mit eval in das Array schreiben  
}  
trace(kugeln[2]); // z.B. das dritte gespeicherte Element im Array abfragen.  
// Gibt _level0.neueKugel2 aus
```

Variante 2. Mit dem Array-Zugriffsoperator erzeugte Clips in einem Array speichern; neuere Variante

```
var kugeln = new Array();  
for (var i = 0; i < 10; i++) { // Schleife, um 10 Clips dynamisch zu erzeugen  
    duplicateMovieClip("meineKugel_mc", "neueKugel" + i, i); // Clips duplizieren  
    kugeln[i] = _root["neueKugel" + i]; // Die neuen Kugel-Clips mit Array-Zugriffsoperator  
// in das Array schreiben  
}
```

Variante 3. Den Returnwert in ein Array speichern; die modernste und empfehlenswerteste Variante, mit duplicate als MovieClip-Methode

```
var kugeln = new Array();  
for (var i = 0; i < 10; i++) { // Schleife, um 10 Clips dynamisch zu erzeugen  
    var mc = meineKugel_mc.duplicateMovieClip("neueKugel" + i, i); // Clips mit MovieClip-Methode duplizieren und  
// Referenz auf die neuen Clips in mc speichern  
    kugeln[i] = mc; // Die neuen Kugel-Clips in das Array schreiben  
}
```

PROGRAMMGENERIERTE CLIPS LÖSCHEN

```
removeMovieClip("form_mc"); // Globale Funktion  
form_mc.removeMovieClip(); // MovieClip-Methode; vorzuziehen
```

CLIPS UND TEXTFELDOBJEKTE VIA PROGRAMMIERUNG ERZEUGEN

Dynamisches Clip-Objekt und dynamisches Eingabe-Textfeld generieren

```
this.createEmptyMovieClip("form_mc", 1); // Leerer Clip auf Hauptbühne (this == _root)
form_mc.createTextField("name_txt", 1, 0, 0, 200, 20); // Parameter: (instanzname, tiefe, x, y, breite, höhe)
form_mc.name_txt.border = true; // Textfeldrahmen sichtbar machen
form_mc.name_txt.type = "input"; // Texteingabe erlauben
```

BITMAPS IN ZEITINTERVALLEN LADEN

Kleine JPEG-Fotoshow mit 12 Bildern in festem Rhythmus

Voraussetzung: 12 JPEGs bild0.jpg bis bild11.jpg (keine progressiven JPEGs!)

```
function diaShow() { // Funktion für die Diashow
    i++; // Lauf- oder Zählvariable: zählt den Wert von i hoch
    posGalerie_mc.loadMovie("bild"+i+".jpg"); // Laden der einzelnen nummerierten Bilder (bild0 bis bild11)
    if (i>=11) { // Zählschleife, damit die Show wieder von vorne beginnt
        i = -1; // -1, damit es die Show erneut mit bild0 beginnt
    }
}
var intervallId = setInterval(diaShow, 1500); // Intervall-Funktion mit Variable; notwendig, um Intervall mit clearInterval wieder
// aufzuheben; Parameter: (Funktionsname und Zeitintervall in Millisekunden)
```

Ende der Vorstellung auf Button-Klick:

```
on (release) {
clearInterval(intervallId); // Intervall via Variable intervallId beenden
}
```

ZUFALL MIT FUNKTIONEN

Per Zufall zu einem Bild zwischen 1 und 6 springen

```
this.gotoAndStop(Math.floor(Math.random() * 6) + 1);
```

Zufallszahl zwischen 10 und 20 (einschließlich) erzeugen

```
function zufallsZahl(minWert, maxWert) {
    return minWert+Math.floor(Math.random()*(maxWert+1-minWert));
}
```

Funktionsaufruf:

```
zufallsZahl(10, 20);
```

BUTTON-KLASSE

Voraussetzung: Button-Instanz meinButton_btn auf der Bühne

```
meinButton_btn._alpha = 50; // Button-Eigenschaft steuern
meinButton_btn.onRollOver = function() { // Button-Objekt mit Rückruffunktion
    trace("Flash tut gut");
}
```

AUF TASTATUREINGABEN MIT LISTENER LAUSCHEN

Zum Testen im Test Player unbedingt »Steuerung / Tastenkombinationen deaktivieren« auswählen

```
keyListener = new Object(); // Neues Listener-Objekt erzeugen
keyListener.onKeyDown = function() { // Rückruffunktion erzeugen
    if (Key.getCode() == Key.ENTER) { // Key-Objekt mit der getCode-Methode verknüpfen
        trace("eingabe");
    }
}
Key.addListener(keyListener); // Das Listener-Objekt für Key registrieren
```

SOUND-OBJEKT

Voraussetzung: Verknüpfter Sound in der Bibliothek oder dynamisch geladener Sound mit soundObject.loadSound(url, isStreaming)

```
fanfare = new Sound("_root"); // Neues Sound-Objekt erzeugen und eine Zeitleiste bestimmen
fanfare.attachSound("Hoerner"); // Den Sound aus der Bibliothek mit dem neuen Objekt verknüpfen
vol = 50; // Variable für die Lautstärke deklarieren und Wert zuweisen (von 0 bis 100)
pan = 0; // Variable für die Lautstärke deklarieren und Wert zuweisen (von -100 bis 100)
fanfare.setVolume(vol); // Die Lautstärke bestimmen
fanfare.setPan(pan); // Die Balance bestimmen
fanfare.start(); // Den Sound starten
fanfare.onSoundComplete = function() { // Funktion, die nach vollständigem Abspielen ausgeführt wird
    trace("Fanfare zuende");
}
```

FLASH-COOKIES

Voraussetzung auf der Hauptbühne: Ein Eingabe-Textfeld namensEingabe_txt; ein dynamisches Textfeld namensAusgabe_txt; ein Button Instanzname sichern_btn; ein Button Instanzname abrufen_btn. Um dieses Feature lokal testen zu können, müssen Sie einen Projektor von dem Film erstellen; andernfalls müssen Sie die Datei auf einen Webserver uploaden.

```
var meinName_so = SharedObject.getLocal("Namens_Cookie"); // Ein neues SharedObject erzeugen; Parameter: (Beliebiger Name für die
// Speicherdatei; Standardgröße 100 Byte)
sichern_btn.onRelease = function() { // Rückruffunktion für den Sicherungs-Button
    meinName_so.data.name = namensEingabe_txt.text; // Dem Objekt den Wert aus dem Eingabe-Textfeld zuweisen
    meinName_so.flush(500); // Statt des automatischen Speicherns beim Schließen des Films die
// Cookiedaten an eine .sol-Datei auf der Festplatte des Users übergeben
// und dabei mehr Speicherplatz anfordern.
}
abrufen_btn.onRelease = function() { // Rückruffunktion für den Ausgabe-Button
    meinName_so = SharedObject.getLocal("Namens_Cookie"); // Aufruf der Eingabedaten aus der Speicherdatei der Festplatte
    namensAusgabe_txt.text = meinName_so.data.name; // Anzeige der Daten im Ausgabe-Textfeld.
}
```

Testen

Aus dem Ordner die Projektordatei öffnen, in das Eingabefeld den Namen eintippen, anschließend auf den Sichern-Button klicken. Den Projektor schließen und anschließend wieder öffnen. Auf den Abrufen-Button klicken.

Einzelne Daten löschen

```
delete meinName_so.data.name;
```

HTML-SWF-KOMMUNIKATION: 2 VARIANTEN

Auf der Seite, die den SWF-Film referenziert, neue Variablen auf der Hauptzeitleiste der SWF erzeugen. Code im Quelltext der HTML-Seite jeweils für Internet-Explorer im Object-Tag und für Netscape im Embed-Tag:

Variante 1 für Flash 5 und Flash MX

```
<PARAM NAME=movie VALUE="main.swf?uebesetzerName=Baumgart+und+Reder">
<EMBED src="main.swf?uebesetzerName=Baumgart+und+Reder"...>
```

Variante 2 für Flash MX, mit dem FlashVars-Parameter (Maximal 63 kb, URL-encoded)

```
<PARAM NAME=FlashVars VALUE="uebersetzerName=Baumgart+und+Reder&gutachterName=Bokelberg">
<EMBED FlashVars="uebersetzerName=Baumgart+und+Reder&gutachterName=Bokelberg+und+Daum" ... >
```

JAVASCRIPT UND FLASH

Ein Browser-Popup-Fenster mit Button-Klick öffnen. Voraussetzung: Button auf der Bühne

```
on (release) {
    getURL ("javascript: var fensterAuf = window.open('http://www.moock.org','meinFenster','height=400,width=500');
fensterAuf.focus();");
}
```

PRELOADER

Voraussetzung: Ein Preloader-Movieclip mit einem Textfeld-Objekt vom Typ Dynamischer Text; Instanzname: geladen_txt; auf Bild 1 des Hauptfilms ein stop() und auf der Bühne eine Instanz des Clips. Ab Bild 2 der eigentliche Filminhalt; auf der Clip-Instanz das folgende Skript:

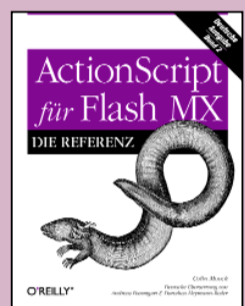
```
onClipEvent (enterFrame) {
    var geladenBytes = Math.round(_root.getBytesLoaded()/1000); // Fragt die geladenen Bytes ab und wandelt sie in Kbyte um
    var gesamtBytes = Math.round(_root.getBytesTotal()/1000); // Fragt die Gesamtzahl der Bytes des Films ab und
// wandelt sie in Kbyte um
    if (geladenBytes >= gesamtBytes) { // Vergleicht die geladenen Bytes mit der Gesamtzahl an Bytes
        _root.gotoAndPlay(2); // Schickt den Abspielkopf in Bild2, wo der vorgeladene
// Filminhalt beginnt
    }
    geladen_txt.text = geladenBytes + " kb von " + gesamtBytes + " kb sind geladen";
}
```

Testen: Im Test Player Ansicht / Streaming anzeigen und Bandbreitenprofiler aktivieren

Eine umfassende und verständliche Einführung in ActionScript und jede Menge weitere wertvolle Tipps finden Sie in den Büchern von Colin Moock:



ISBN: 3-89721-354-0
588 Seiten, 36,- €



ISBN: 3-89721-355-9
686 Seiten, 38,- €