Chapter 2

ActionScriptの要素と書きかた

ここでは、ActionScriptを構成する要素や
書きかたのルールを説明します。
スクリプトの基礎知識として、頭に入れておきましょう。
特に「オブジェクトの指定」は、
ActionScriptを記述するときのポイントです。
実際にレッスンをして、書きかたを理解してください。

Flash ActionScript

2-01 まずこれだけは覚えておこう

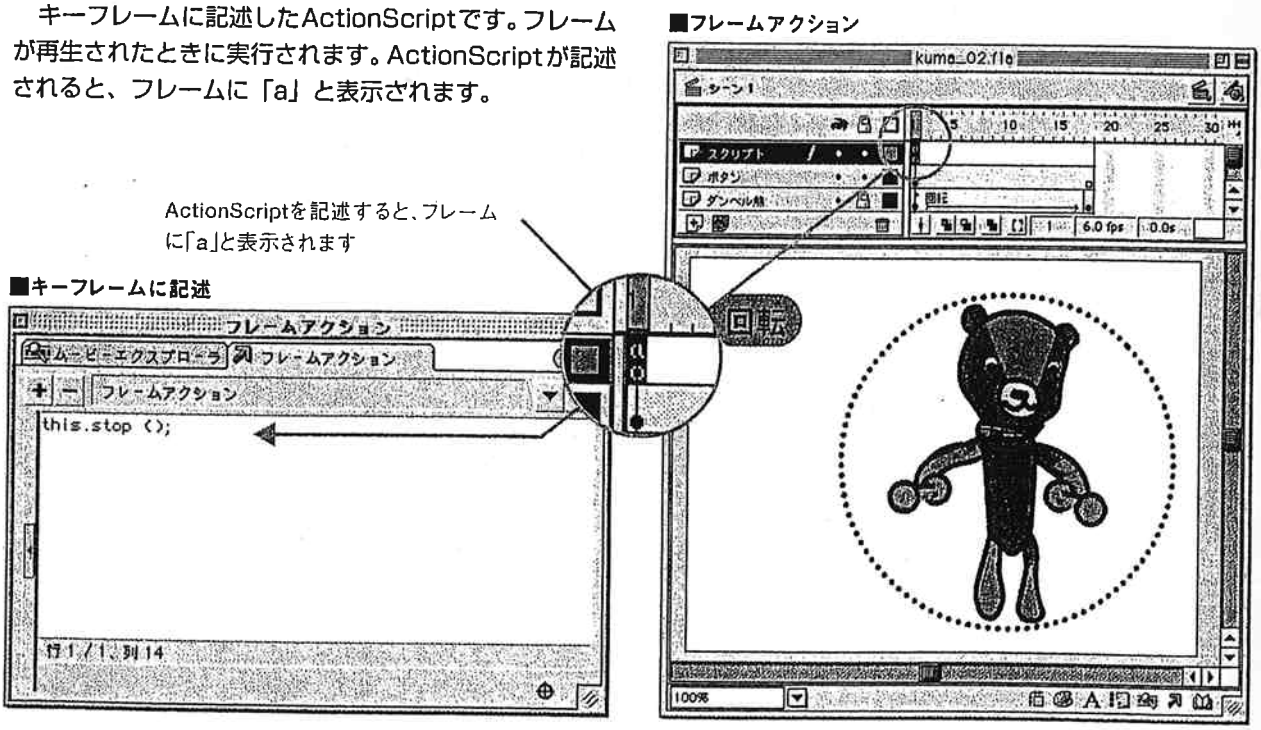
ここで説明することは、ActionScriptを使う上で、はじめに覚えておくべきことです。Chapter 1の説明と重複するところもありますが、おさらいしておきましょう。

ActionScriptを記述できるのは3カ所

ActionScriptは、キーフレーム、ボタンのインスタンス、ムービークリップのインスタンスの3カ所に記述できます。記述する場所によって、次の3種類に分けられます。

■フレームアクション

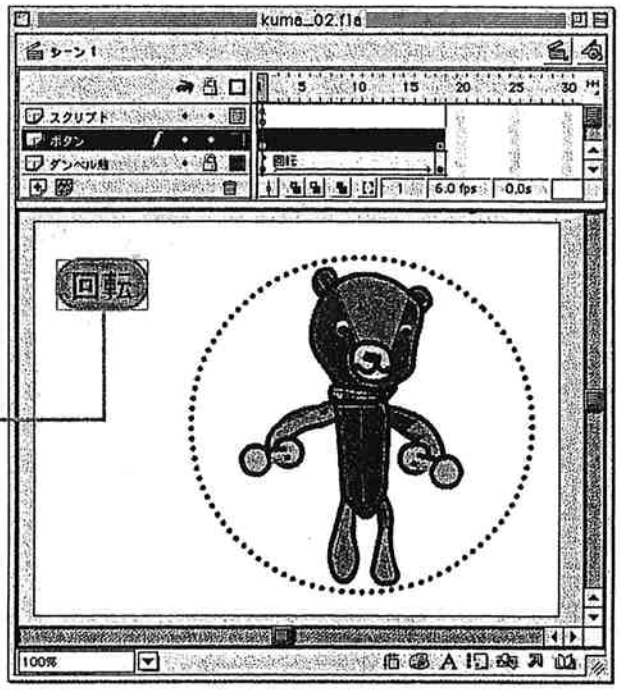
キーフレームに記述したActionScriptです。フレームが再生されたときに実行されます。ActionScriptが記述されると、フレームに「a」と表示されます。



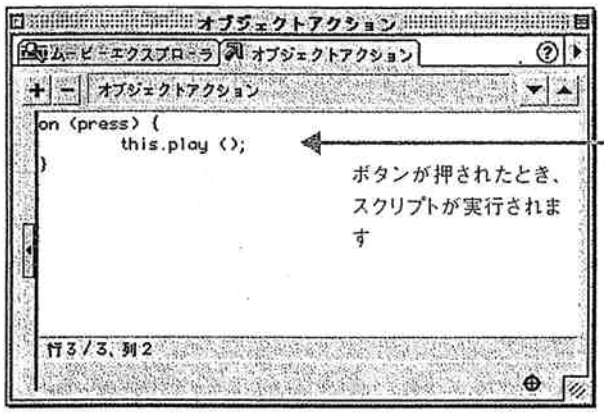
■ボタンアクション

ボタンのインスタンスに記述したActionScriptです。指定したイベントが行われたときに実行されます。例えばイベントを「press」と指定すると、ボタンが押されたときにスクリプトが実行されます。

■ボタンアクション



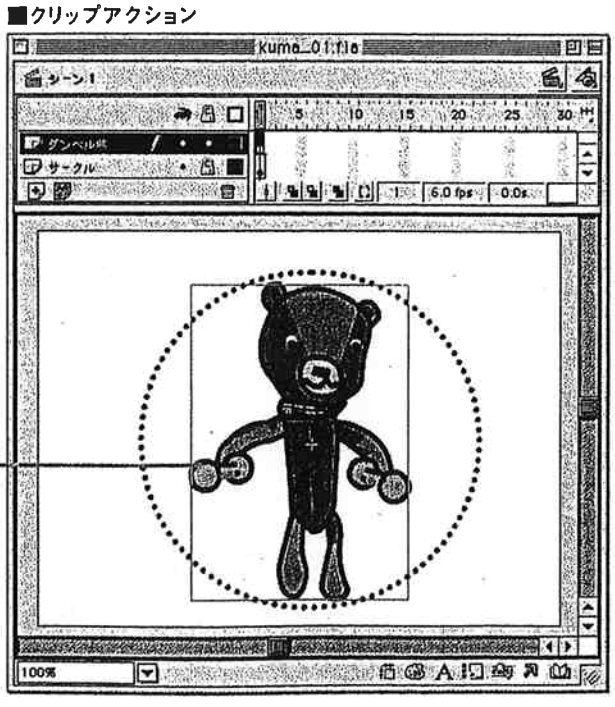
■ボタンのインスタンスに記述



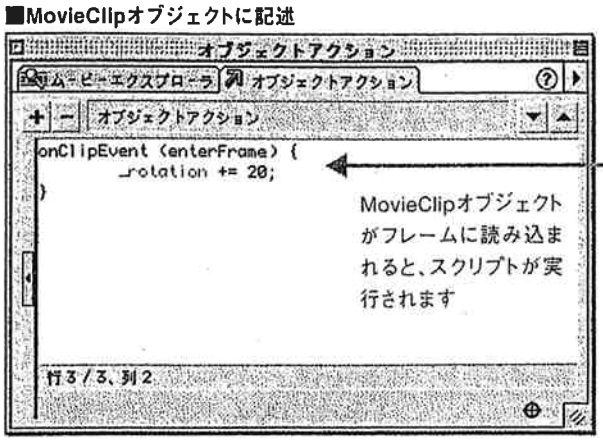
■クリップアクション

ムービークリップのインスタンス（MovieClipオブジェクト）に記述されたActionScriptです。指定したイベントが行われたときに実行されます。例えばイベントを「enterFrame」と指定すると、MovieClipオブジェクトがフレームに読み込まれるたび、つまりMovieClipオブジェクトを配置したフレームが再生されるたびにスクリプトが実行されます。

■クリップアクション



■MovieClipオブジェクトに記述



Flash ActionScript

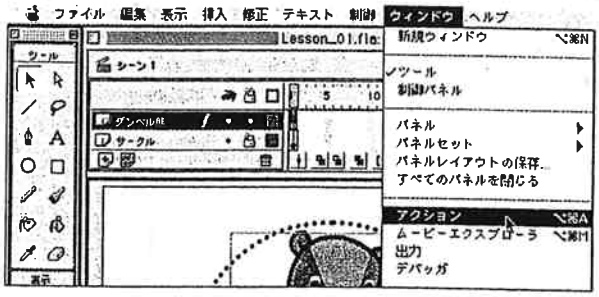
2-01 まずこれだけは覚えておこう

[アクション] パネルを使って記述する

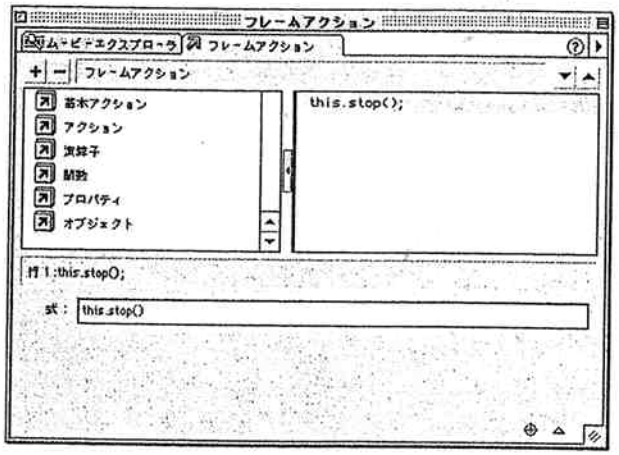
ActionScriptは [アクション] パネルを使って記述します。[アクション] パネルを開くには、[ウィンドウ] メニューの [アクション] を選択するか、ランチャーバーの [アクション] パネルボタン をクリックします。

パネルの表示は、フレームアクションを記述するときは [フレームアクション]、ボタンアクションとクリップアクションを記述するときは [オブジェクトアクション] となります。

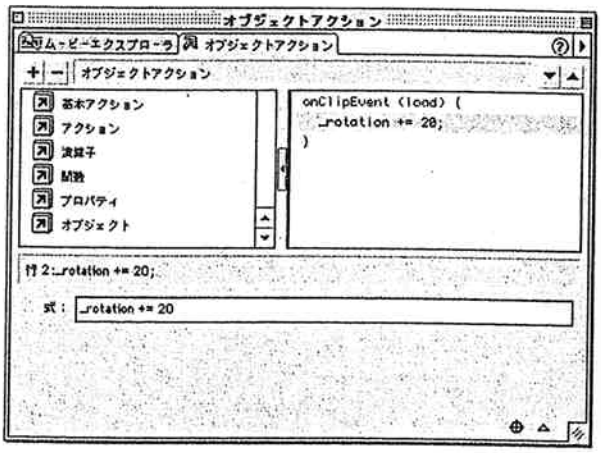
■[アクション]パネルを開く



■[フレームアクション]パネル



■[オブジェクトアクション]パネル

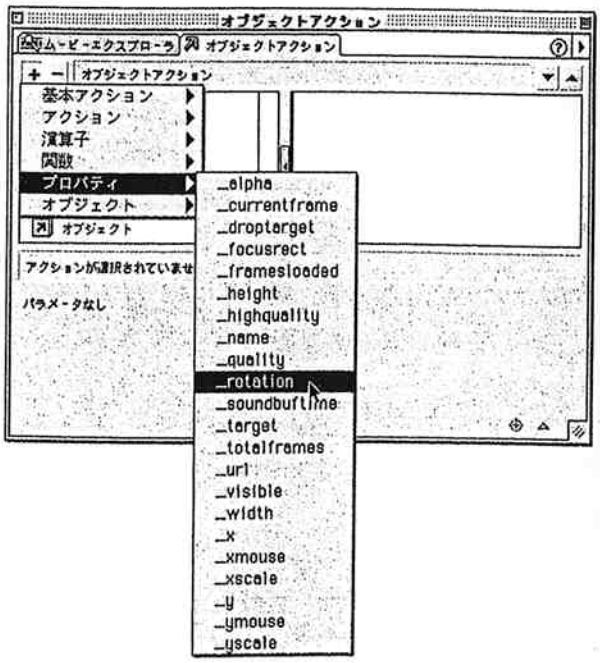


スクリプトはメニューから選べる

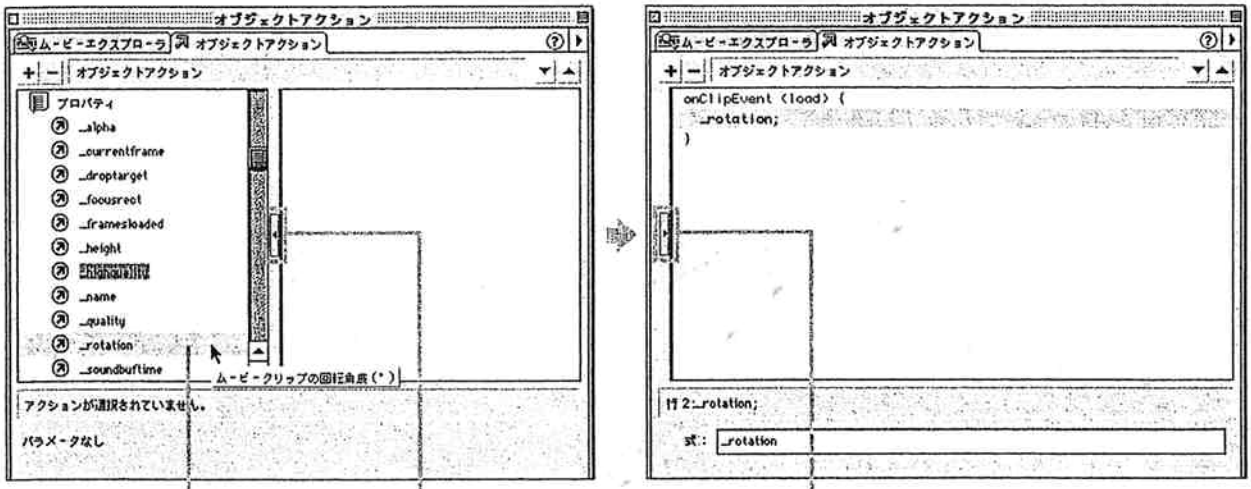
[アクション] パネルの + (ステートメントの追加ボタン) をクリックすると、ActionScriptのメニューが開きます。ここから記述したいプロパティやメソッドなどを選ぶことができます。または左側の [ツールボックスリスト] で、記述したいプロパティやメソッドなどをダブルクリック (またはドラッグ) しても構いません。

なお [ツールボックスリスト] を使わないときは、中央の三角ボタンをクリックして閉じておきます。

■メニューからプロパティを選ぶ



■ツールボックスリストからプロパティを選ぶ



ダブルクリックします

ここをクリックすると左側の［ツールボックスリスト］が閉じます

もう一度ここをクリックすると［ツールボックスリスト］が表示されます

2つの編集モードがある

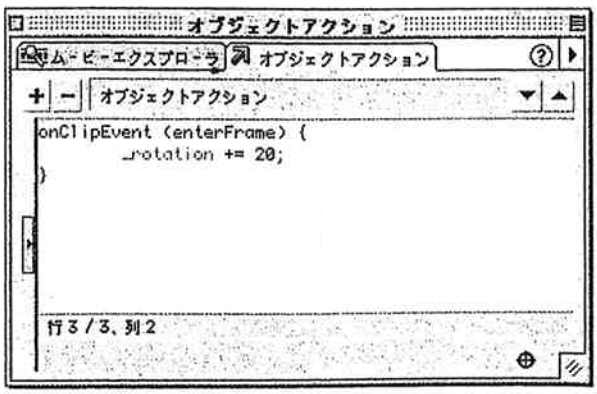
［アクション］パネルの編集モードには「ノーマルモード」と「エキスパートモード」があり、初期設定では「ノーマルモード」になっています。「ノーマルモード」は選択式でスクリプトを記述するモードです。記述は簡単ですが、細かい修正が面倒です。「エキスパートモード」はスクリプトを直接記述するモードです。ワープロ感覚で文字列を扱えるので、編集が楽です。

本書では「エキスパートモード」を使って作業します。モードを切り替えるときはパネル右上の▶をクリックし、ポップアップメニューからモードを選びます。

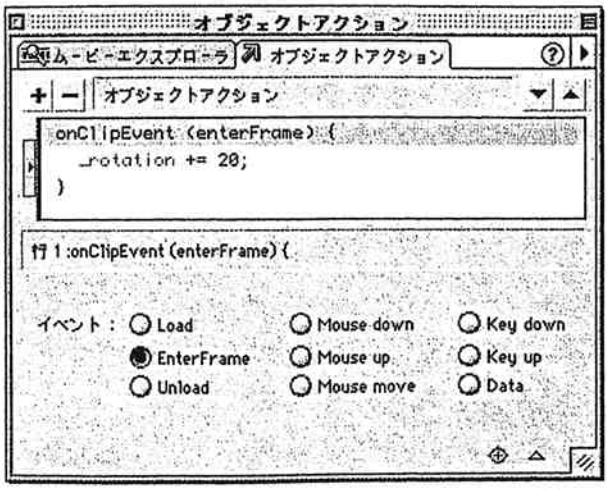
ポップアップメニューからモードを選びます

■編集モードを切り替える

■エキスパートモードのパネル表示



■ノーマルモードのパネル表示



CHECK

ノーマルモードとエキスパートモード 18

Flash ActionScript

2-02 ActionScriptの文法はとてもシンプル

ActionScriptの勉強は、外国語の勉強に似ています。実践も大切ですが、基本となる文法を覚えることも必要です。ここでは記述のルール、文法を確認しましょう。

「どこの」「何を」「どうするか」をドット（.）でつなげる

ActionScriptでは、「命令を実行する対象」と「命令」をドット（.）でつなげて記述します。「何を」「どうするか」をドットでつなげて書いていく、と考えるとわかりやすいでしょう。

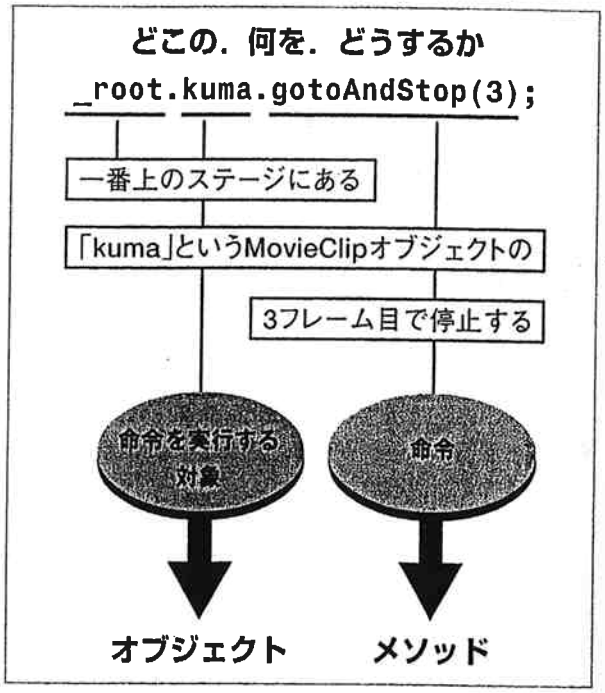
「命令を実行する対象」は「オブジェクト」と呼ばれ、オブジェクトの前には「_root」のような場所を示す言葉を記述します。「どこの」「何を」「どうするか」をドットでつなげて書く、と考えましょう。

オブジェクトに出す「命令」にはいろいろな種類がありますが、代表的なのは「メソッド」と呼ばれるものです。これはオブジェクトに何かの動作をさせるもので、「フレームにジャンプして停止する」という「gotoAndStop()」もメソッドの1つです。

MEMO このようにドットでつなげて記述する方法を「ドットシンタックス」といいます。JavaScriptでも使われています。

CHECK オブジェクト 44
メソッド 46

■ドットでつなげて記述する



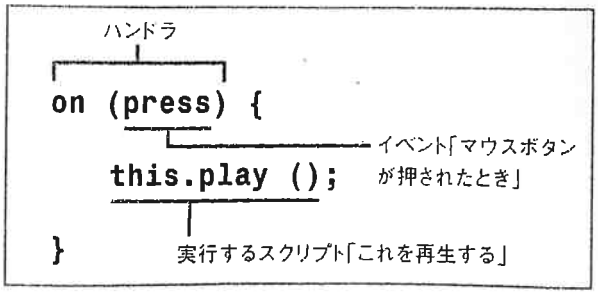
クリップアクションとボタンアクションにはイベントを付ける

イベントは、命令を実行するタイミングを指定するものです。クリップアクションとボタンアクションには、必ず「〜したとき」というイベントを付けます。

イベントはハンドラの（）内に記述し、イベントが行われたときに実行するスクリプトは{}内に記述します。

CHECK ハンドラとイベント 190

■イベントを付けたスクリプト



■クリップアクションのハンドラとイベント

ハンドラは「onClipEvent ( )」です。( ) 内にイベントを指定します。よく使うイベントは、次の２つです。

イベント	意味
load	MovieClipオブジェクトが最初にフレームに読み込まれたとき（フレームが最初に再生されたとき）
enterFrame	MovieClipオブジェクトがフレームに読み込まれるたびに(フレームが再生されるたびに)

■ボタンアクションのハンドラとイベント

ハンドラは「on ( )」です。( ) 内にイベントを指定します。よく使うイベントは、次の３つです。

イベント	意味
press	ボタンの上でマウスボタンを押したとき
release	ボタンの上でマウスボタンを離したとき
rollOver	ボタンの上にマウスカーソルが重なったとき

その他のルール

ActionScriptの記述には、他に次のようなルールがあります。

■半角英数字を使う

全角文字を使うとエラーになります。スペースも半角で入力しましょう。

■大文字と小文字

基本的には、大文字と小文字のどちらを使っても構いません。ただし、ActionScriptが「キーワード」に指定している次の言葉は、すべて小文字で記述します。

break continue delete else for function if in new return this typeof var void while with

MEMO キーワードはインスタンス名など、ユーザーが設定する名前にも使うことができません。

■字下げにはタブを使える

ハンドラや条件文の中に命令を記述するときは、字下げをした方が見やすくなります。ノーマルモードでは自動的に字下げされますが、エキスパートモードではtabキーを押して字下げするといいでしょう。

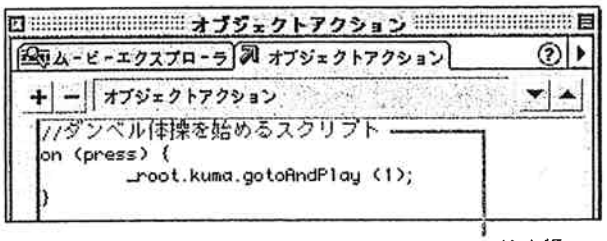
■( )と{ }

( )はメソッドなどの命令で値を指定するとき、{ }はスクリプトをブロック単位で区切るときに使います。いずれも対になっていないとエラーになります。( )や{ }の前後に、見やすいように半角スペースを入力してもかまいません。

■コメント行には // を付ける

スクリプトの中に注釈やコメントを付けたいときは、行頭にスラッシュを2つ重ねて入力します。複数行をコメントにする場合は、「/*」と「*/」で囲んでも構いません。

■コメント行を入れる



コメント行

■文章（ステートメント）の最後には ; を付ける

文章の終わりにはセミコロン（ ; ）を付けます。ただし付け忘れてもエラーにはなりません。ステートメントごとに改行していれば大丈夫です。なお、エキスパートモードからノーマルモードに切り替えると自動的に付きます。

■式には演算子を使う

数値の計算、プロパティなどへの値の代入、値の比較をするときは、それぞれ決められた演算子を使います。演算子には、数値演算子、代入演算子、比較演算子、論理演算子などがあります。

式	意味
A = B+C	BとCを足してAに代入する
_rotation += 20	角度に20をプラスする
_width = 100	横幅を100にする
A == B	AとBが等しい
A > B	AがBより大きい
A == 50 && B != 100	Aが50かつBが100でない

CHECK 演算子一覧 188

Flash Action Script

2-03 ActionScriptの各要素をみてみよう

スクリプトを構成する要素には、オブジェクト、メソッド、プロパティ、アクション、変数などがあります。それぞれの意味を押さえておきましょう。

オブジェクト ～命令を実行する対象～

オブジェクトは、ActionScriptの基本となる要素です。ActionScriptに記述した命令は、オブジェクトに対して実行されます。ですからスクリプトの中では、オブジェクトの場所と名前を正確に記述する必要があります。

■MovieClipオブジェクト

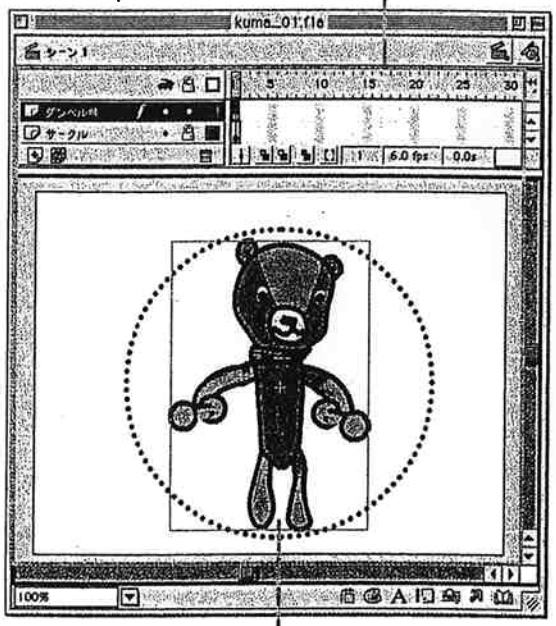
オブジェクトの中で、最もよく使われるのは「MovieClipオブジェクト」です。MovieClipオブジェクトは、ステージ上に配置したムービークリップのインスタンスです。

例えば、14ページのLesson 1では、ステージ上に配置した「ダンベル熊」の角度を変えました。また32ページのLesson 3では、ステージ上に配置した「ダンベル熊」のタイムラインを操作しました。これらは「ダンベル熊」のインスタンス、つまりMovieClipオブジェクトに命令を出した例です。

なお、28ページのLesson 2では、ムービーのタイムラインを操作しました。実はActionScriptでは、ムービー自体もひとつのMovieClipオブジェクトとして扱うことができます。MovieClipオブジェクトの中にMovieClipオブジェクトが入っている、という階層構造になっていると考えましょう。

ムービー＝ひとつのMovieClipオブジェクトと考えます

■MovieClipオブジェクト



ムービークリップのインスタンス＝MovieClipオブジェクト

MEMO オブジェクトの中には、目に見えないものもあります。例えば、MovieClipオブジェクトの色を変えるためには「Colorオブジェクト」というオブジェクトを利用しますが、これは概念として定義するもので、実体のあるものではありません。

CHECK MovieClipオブジェクトの色を変える .. 124

■オブジェクトの指定

オブジェクトは場所と名前で指定します。場所は右の3つで示します。

表記	場所
_root	一番上のステージ（タイムライン）
_parent	1つ上のオブジェクト
this	自分自身

通常は、一番上の階層「_root」から目的のオブジェクトまでの名前を順番に記述します。MovieClipオブジェクトの名前はインスタンス名です。

例： `_root.kuma.arm1.play();`

一番上のステージに置かれた「kuma」内の「arm1」を再生する

自分自身を表すときは「this」を指定します。名前は不要です。

例： `this.play();`

これを再生する

MEMO ボタンはオブジェクトではありません。ボタンアクションで「this」と指定したときは、そのボタンが置かれているMovieClipオブジェクトを指します。

自分より下の階層を指定するときは「this」の後に名前を記述します。

例： `this.arm1.play();`

この中に置かれた「arm1」を再生する

MEMO オブジェクトはダイアログボックスのツリーから選んで記述することもできます。この操作は後で実際に練習します。

CHECK オブジェクトを正確に記述しよう 48

■プロパティを変えるときはオブジェクトを省略できる

14ページのLesson 1では、ダンベル熊のMovieClipオブジェクトに次のようなスクリプトを書きました。

```
onClipEvent (enterFrame) {
  _rotation += 20;
}
```

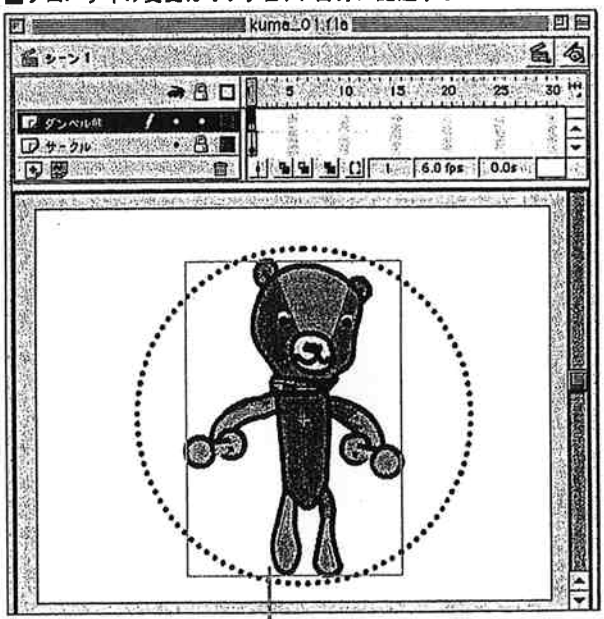
これにはオブジェクトの記述がありませんね。「_rotation += 20;」（角度に20度をプラスする）という命令しかありません。ここではオブジェクトの記述を省略しているのです。

命令といっても「_rotation」はメソッドではなく、MovieClipオブジェクトの角度を示すプロパティです。実は、ActionScriptには**「プロパティを変更するスクリプトは、そのオブジェクト自身に記述する」**という原則があります。つまり「命令を実行する対象」は、常に自分自身になります。

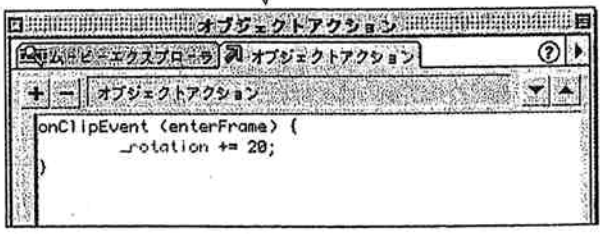
あえてオブジェクトを書くなら、自分自身を示す「this」を付けて「this._rotation += 20;」となりますが、上記の前提があるので、プロパティを変えるときは省略して構わないのです。

MEMO 「プロパティを変更するスクリプトは、そのオブジェクト自身に記述する」という原則は、絶対ではありません。他のオブジェクトのプロパティを変えることも可能です。その場合はオブジェクトの指定をします。

■プロパティの変更はオブジェクト自身に記述する



ダンベル熊の角度を変えるときは、ダンベル熊のMovieClipオブジェクトに記述します



Flash ActionScript

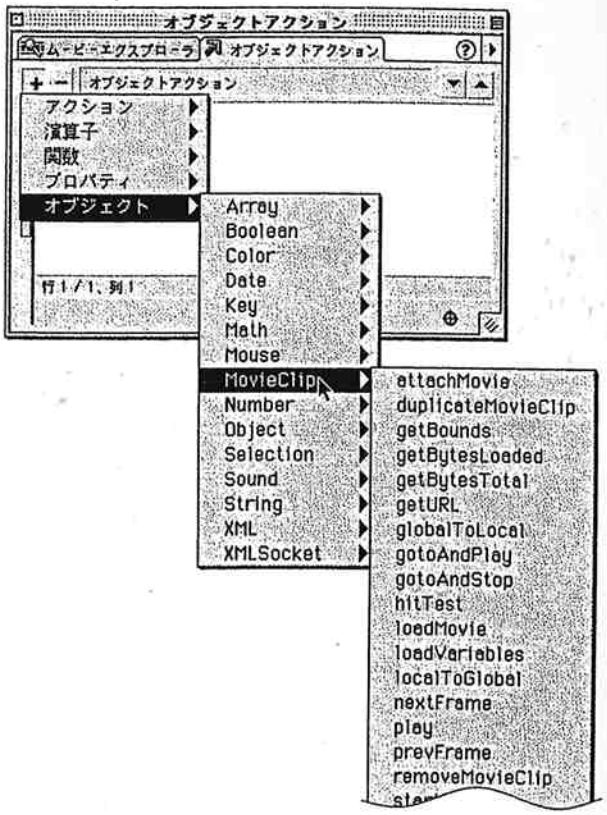
2-03 ActionScriptの各要素をみてみよう

メソッド　～命令～

「メソッド」は、オブジェクトに何かの動作をさせる命令です。ActionScriptには、数多くのメソッドが用意されていて、[アクション] パネルのメニューから選ぶことができます。一番よく使うのは、MovieClipオブジェクトのメソッドです。

例えば、「gotoAndPlay( )」は「指定したフレームにジャンプして再生する」というMovieClipオブジェクトのメソッドです。

■MovieClipオブジェクトのメソッド



MEMO 「function」というアクションを使って、自分でメソッドを定義することもできます。

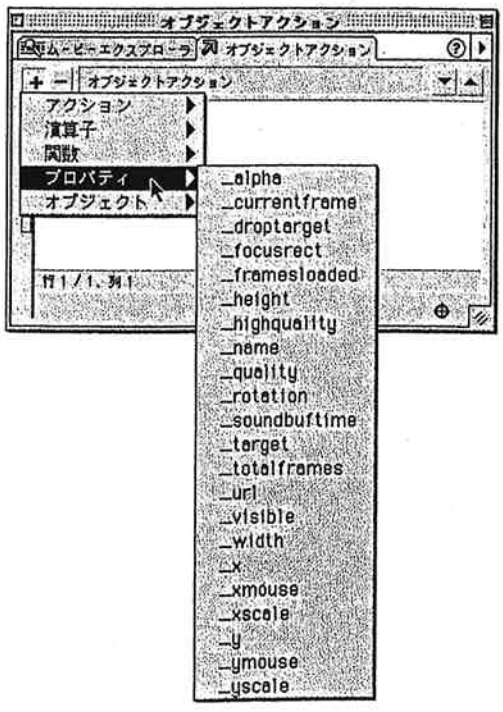
CHECK MovieClipオブジェクトのメソッド .. 192
自分でメソッドを定義する 128

プロパティ　～属性～

プロパティは、オブジェクトの属性のことです。MovieClipオブジェクトの属性には、透明度（_alpha）、角度（_rotation）、横幅（_width）などがあり、[アクション] パネルのメニューから選ぶことができます。

ActionScriptでは、プロパティの値を調べたり変更することができます。例えば、Aというオブジェクトの横幅と縦幅を調べて、Bのオブジェクトにその値を代入すれば、AをBを同じ大きさにすることができます。

■プロパティのポップアップメニュー

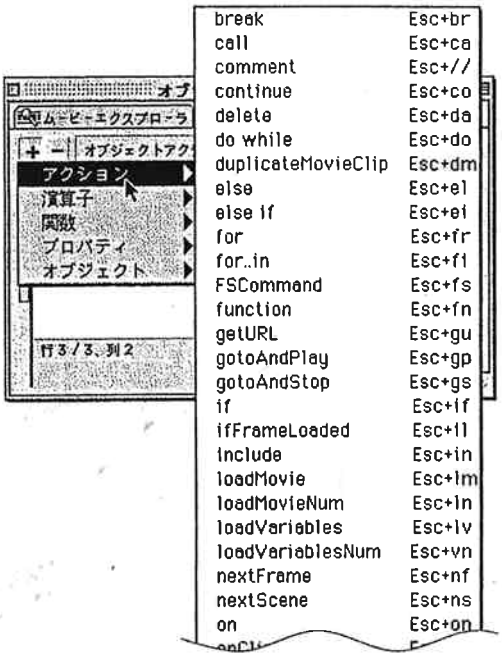


CHECK プロパティ 195

アクション

「アクション」は、オブジェクトに命令を与えるものではなく、単独の働きを持つものです。条件を指定する「if」、メソッドを作成する「function」、指定した範囲を印刷する「print」などがあります。

■アクションのポップアップメニュー



MEMO [アクション] パネルの [アクション] メニューには、たくさんのアクションが表示されますが、よく見るとMovieClipオブジェクトのメソッドとだぶっているもの数多くあります。アクションに用意されているものは、旧バージョンの機能を引きずっているものなので利用はおすすめしません。だぶっている場合は、メソッドを優先して使ってください。また、アクションの中には「tellTarget」のように今後のサポートが保証されていないものもあります。

CHECK アクション 189

変数

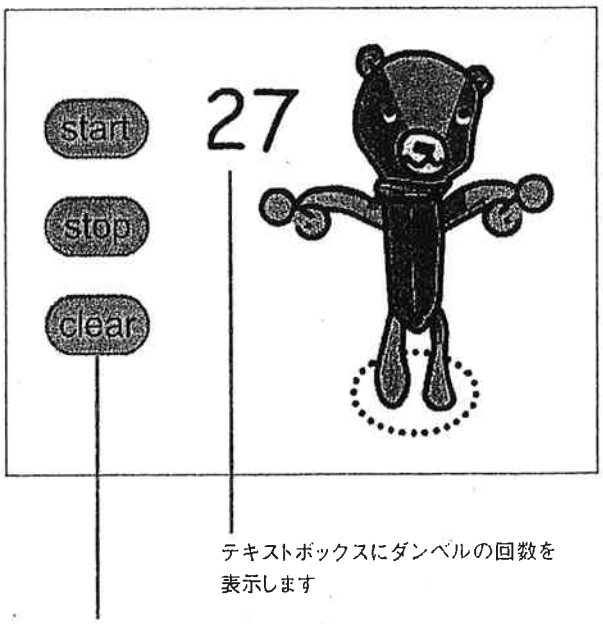
「変数」は、文字や数値といった値を入れておく容器です。必要に応じてActionScriptの中で定義して利用します。例えば、合計金額の計算結果を「total」という変数に入れておき、それを別のActionScriptから参照することができます。

また、テキストボックスを変数として利用することもできます。参照した「total」の金額が高すぎたら、テキストボックスに「高すぎます！」という文字を表示する、といった処理をすることもできるのです。

右のムービーも、テキストボックスを変数として利用している例です。ダンベル熊が腕を上げた回数をカウントして、その結果をムービー上のテキストボックスに表示しています。そして「clear」ボタンを押したときに、テキストボックス（変数）をゼロに戻しています。

なお、変数の利用方法については、Chapter 3で詳しく説明します。

■変数を使う



CHECK
候補の中から文字を選んで表示する 68
ユーザーが入力した文字を表示する 72
外部ファイルの文章を表示する 88
ドラッグに条件を付ける 118

Flash Action Script

2-04 オブジェクトを正確に記述しよう
– Lesson 4 –

実行の対象となるオブジェクトを正しく指定しないと、ActionScriptは思った通りに動きません。オブジェクトは、ダイアログボックスのツリーから選ぶこともできます。ここで実際に練習してみましょう。

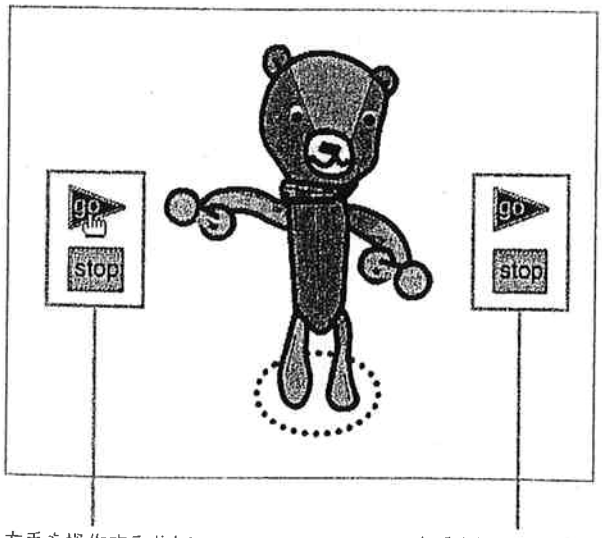
完成ファイル：「chap2」→「2_04」→「kuma_04.fla」
レッスン用ファイル：「chap2」→「2_04」→「Lesson_04.fla」

右手と左手を別々のボタンで動かす

練習に使うのは、またもやダンベル熊です。ここでは、右手と左手を別々に体操させます。それぞれ「go」ボタンを押したら体操を始め、「stop」ボタンを押したら体操をやめるようにします。「go」ボタンと「stop」ボタンに、腕を操作するスクリプトを記述しましょう。

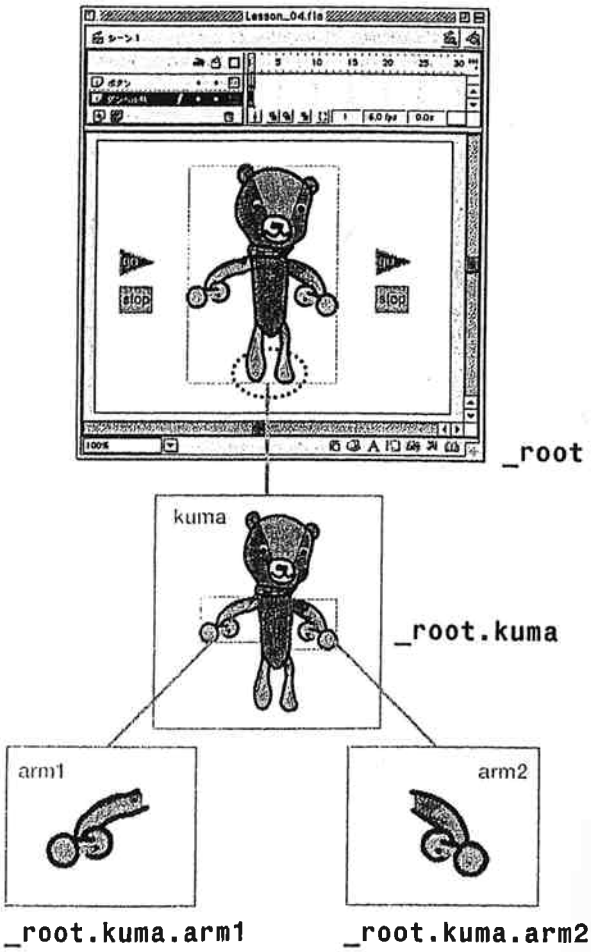
このムービーでは、一番上のステージに「ダンベル熊」のムービークリップを配置し、インスタンス名を「kuma」と付けています。さらに「ダンベル熊」の中では「右手」と「左手」が別のムービークリップになっていて、右手に「arm1」、左手に「arm2」というインスタンス名を付けています。

■右手と左手をボタンで動かす



それぞれのMovieClipオブジェクトは、次のように記述できます。これをオブジェクトのパスといいます。オブジェクトの住所のようなものと考えてください。

■オブジェクトは階層になっている



01 イベントを記述する

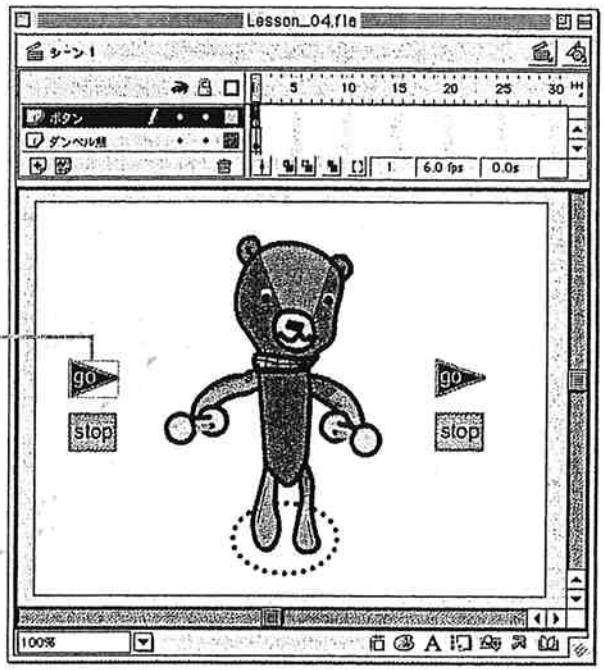
「Lesson_04.fla」を開きます。左側の「go」ボタンを選択し［アクション］パネルを開きます。

［アクション］パネルには、次のようなマウスのイベントが記述されています。

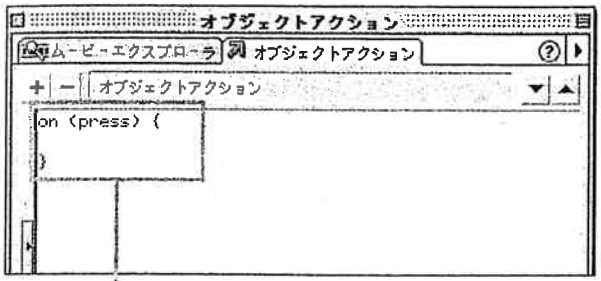
```
on (press) {

}
```

■［アクション］パネルを開く



ボタンを選択します

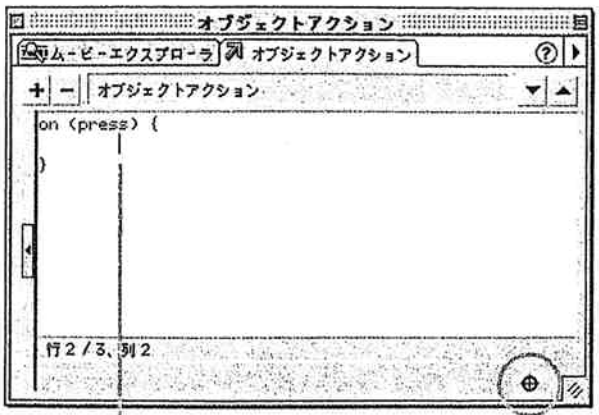


イベントが記述されています

02 ［ターゲットパスの挿入］ダイアログボックスを開く

オブジェクトの入力位置にカーソルを移動し、［アクション］パネル右下の［ターゲットパスの挿入］ボタン ⊕ をクリックします。

■［ターゲットパスの挿入］ダイアログボックスを開く



カーソル

ここをクリックします

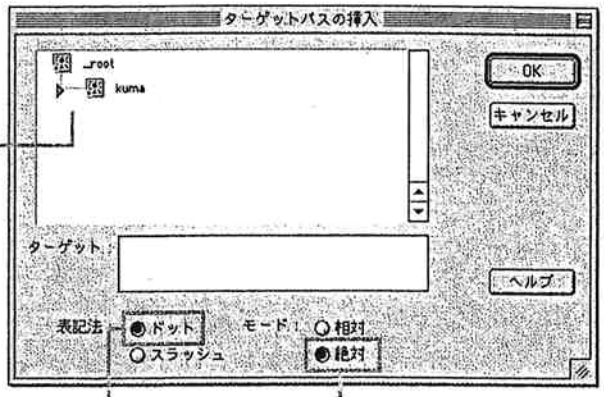
MEMO ここではイベントの下の行にカーソルを移動し、tab キーを押してカーソルを字下げしました。

2-04 オブジェクトを正確に記述しよう –Lesson 4–

03 モードを選ぶ

［ターゲットパスの挿入］ダイアログボックスが表示されたら、［表記法］に「ドット」、［モード］に「絶対」を選びます。

■［ターゲットパスの挿入］ダイアログボックス



オブジェクトの階層がツリー表示されます

「ドット」を選びます　「絶対」を選びます

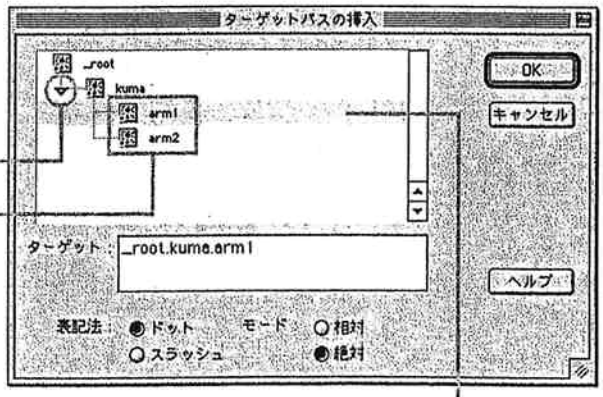
MEMO 一番上の階層「_root」からオブジェクトを記述したいときは、このように［モード］を「絶対」にします。ダイアログボックスには、オブジェクトの階層がツリーで表示されます。

04 オブジェクトを選択する

ダイアログボックスのツリーからオブジェクトを選び、［OK］ボタンをクリックします。

このスクリプトでは、右手の体操を始める命令を「arm1」に出すので「arm1」を選択します。

■オブジェクトを選択する



ここをクリックします

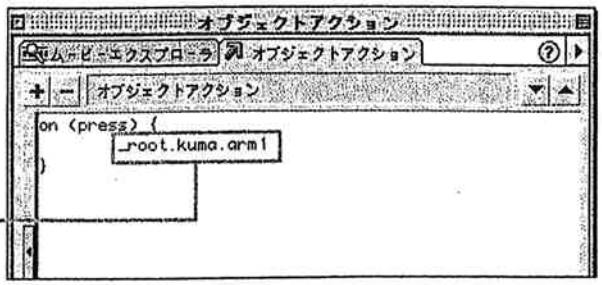
下の階層のオブジェクトが表示されます

右手の場合は「arm1」、左手の場合は「arm2」を選択します

これで［アクション］パネルにオブジェクト「_root.kuma.arm1」が記述されました。

```
on (press) {
  _root.kuma.arm1
}
```

■オブジェクトが記述された



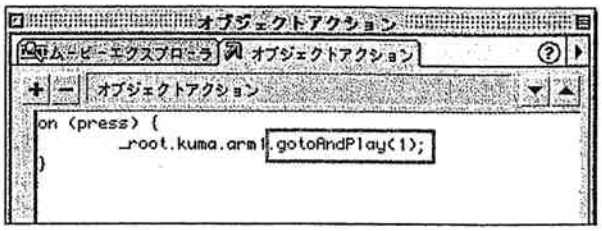
新たに「_root.kuma.arm1」が記述されます

05 メソッドを記述する

［アクション］パネルにオブジェクト「_root.kuma.arm1」が記述されたので、続けて、このオブジェクトに行うメソッドを記述します。メソッドの前にドットを入力するのを忘れないようにしましょう。

```
on (press) {
  _root.kuma.arm1.gotoAndPlay(1);
}
```

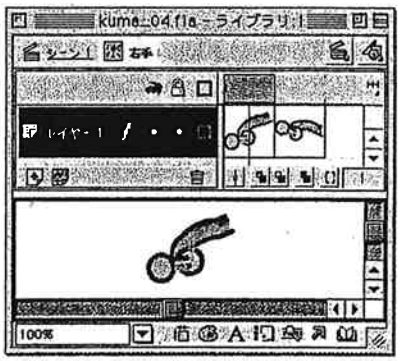
■メソッドを記述する



これで「ボタンが押されたら、一番上のステージに置かれた「kuma」内の「arm1」の１フレーム目から再生する」というスクリプトが書けました。

「arm1」は２フレームで右手を上下させるムービークリップですから、ボタンが押されるとダンベル体操が始まります。

■「arm1」(右手)のムービークリップ



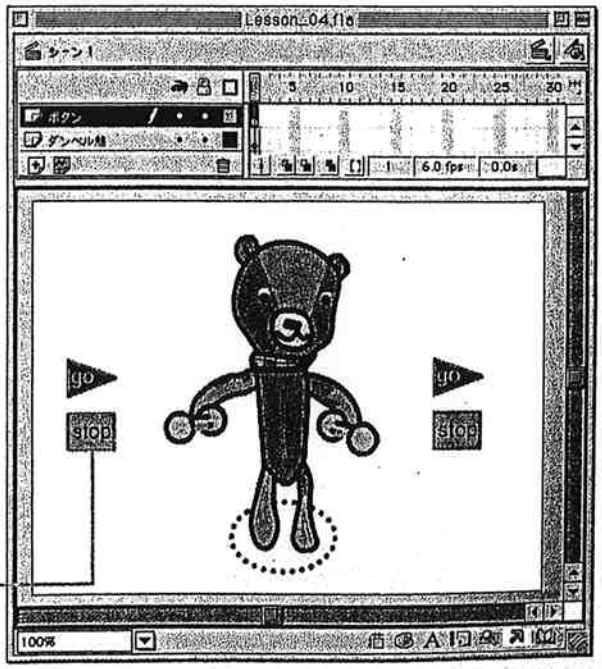
06 ボタンアクションを記述する

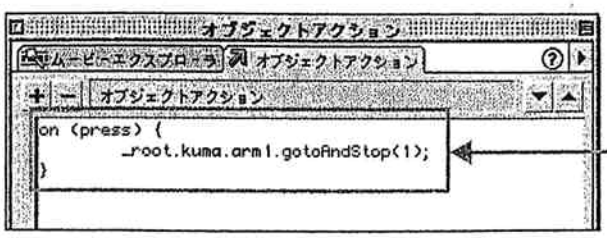
手順⓫～⓯と同様に操作して、左側（右手）の「stop」ボタンに次のようなスクリプトを記述します。

```
on (press) {
  _root.kuma.arm1.gotoAndStop(1);
}
```

MEMO エキスパートモードで記述しているときは、「go」のボタンアクションをコピーして、メソッドの「play」を「stop」に書き換えても構いません。

■左側(右手)の「stop」ボタンのスクリプト





Flash ActionScript 1.0

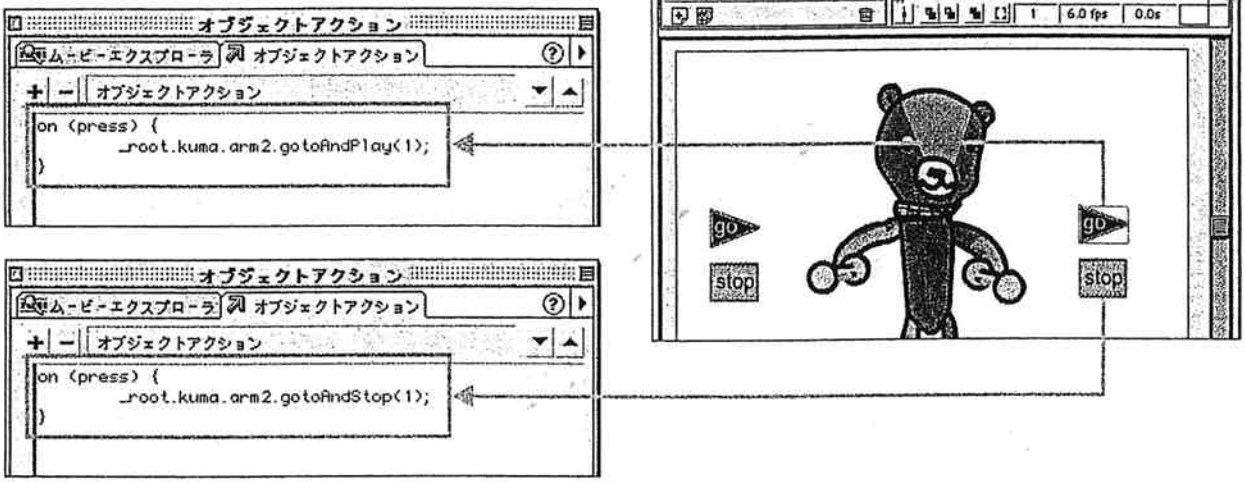
2-04 オブジェクトを正確に記述しよう –Lesson 4–

07 ボタンアクションを記述する

右側（左手）の「go」ボタンと「stop」ボタンにも「_root.kuma.arm2」へのメソッドを記述します。

「_root.kuma.arm2」も［ターゲットパスの挿入］ダイアログボックスから選ぶことができます。

■右側（左手）の「go」ボタン、「stop」ボタンのスクリプト



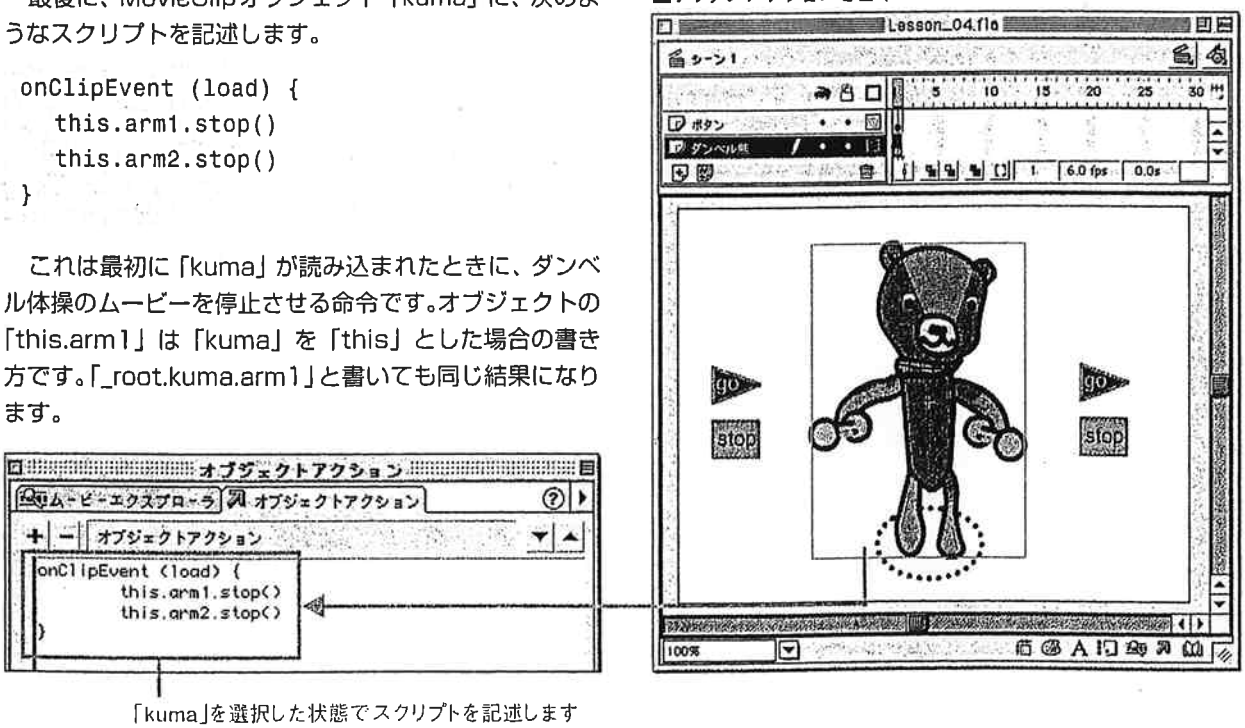
08 クリップアクションを記述する

最後に、MovieClipオブジェクト「kuma」に、次のようなスクリプトを記述します。

```
onClipEvent (load) {
   this.arm1.stop()
   this.arm2.stop()
}
```

これは最初に「kuma」が読み込まれたときに、ダンベル体操のムービーを停止させる命令です。オブジェクトの「this.arm1」は「kuma」を「this」とした場合の書き方です。「_root.kuma.arm1」と書いても同じ結果になります。

■クリップアクションを書く

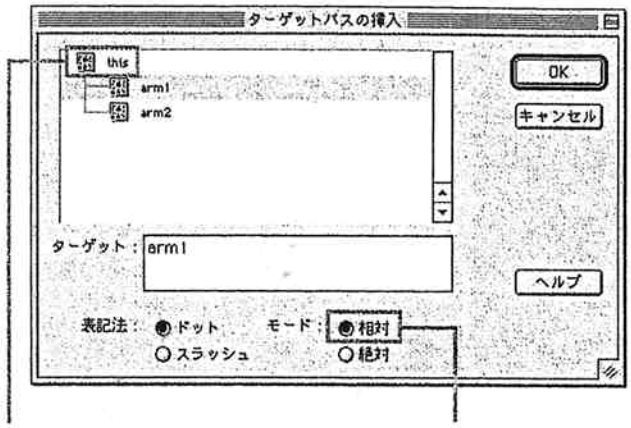


「kuma」を選択した状態でスクリプトを記述します

記述方法のヒント

自分自身（スクリプトを記述しているオブジェクト）を基準にして、他のオブジェクトを指定するときは、［ターゲットパスの挿入］ダイアログボックスで［モード］に「相対」を選びます。ただし、この場合は自分より下の階層のオブジェクトしか表示されません。また「this」は自分でスクリプトに記述する必要があります。わざわざダイアログボックスを開かず、キーボードから入力した方が早いかもしれません。

■「this」を基準にオブジェクトを表示する



この場合、「相対」モードでは「kuma」が基準になります

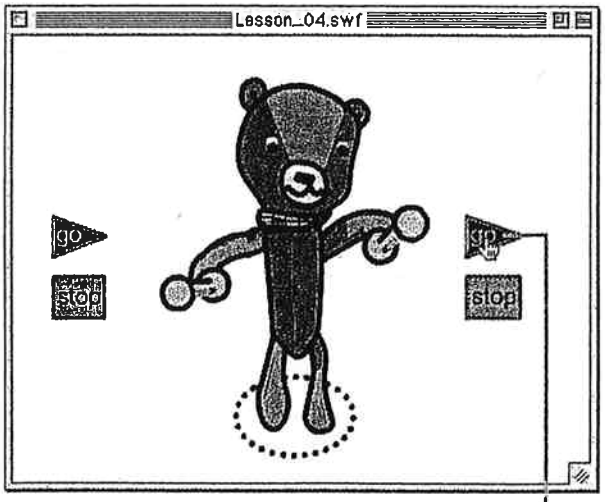
「相対」を選びます

動きをテストする

記述したスクリプト通りにムービーがちゃんと動くかどうか、［制御］メニューの［ムービープレビュー］で確かめます。「go」ボタンや「stop」ボタンをクリックして、ダンベル熊の左右の腕を動かしてみましょう。オブジェクトの指定が正しければ、ちゃんと動くはずです。

このムービーでは、左側のボタンで右手を動かし、右側のボタンで左手を動かすので、スクリプトを記述するときに混乱するかもしれません。逆の手が動いてしまったときは、「arm1」（右手）と「arm2」（左手）が正しく記述されているか確認しましょう。

■動きを確かめる



右側の「go」ボタンで左手が動きます