ActionScript3.0入門

林　拓也

Flash オーサリングエンジニア
アドビ認定インストラクター(ACI)

Adobe

Who is this guy?

- 林拓也
- STUDIO HapHands　http://www.haphands.com/
- Flash オーサリングエンジニア
- アドビ認定インストラクタ (ACI)
- マルチメディア系制作会社を退職後、フリーランスとしてDirector、Flashのオーサリングを中心に活動。各種アプリケーションのインストラクターとしても多くの実績があり、Adobe認定資格も多数取得。また、書籍や教材用テキストの執筆も行うなど、活動は多岐にわたる（http://www.haphands.com/profile/index.html）

本日のテーマ

- ActionScript 1.0 または 2.0 の基本を習得されている方を対象とした、ActionScript 3.0 の初歩をハンズオン (操作演習) で習得できるトレーニングです。
- ActionScript 3.0 の特徴的な所や、今後の Flash アプリケーション作成に役立つ新しいコンポーネントなどを取り扱います。
- **カスタムクラス制作を通して、オブジェクト指向プログラミングの基本にふれる。**

本日の内容

1. ActionScript 3.0の概要
2. ハンズオン : ActionScript 3.0プログラミングをしてみよう
   熱闘！カーチェイスゲーム
   - イベント処理の復習
   - オブジェクト指向の概要
   - 実習（1） 今までのやり方で作る
   - 実習（2） カスタムクラスの導入
   - 実習（3） メソッドのオーバーライドによる機能拡張
3. 今後に役立つTips
4. おわりに

ActionScript 3.0の特徴

- メリット
  - オブジェクト指向プログラミング
    - プログラム言語として、書き方の一貫性が保たれている。
    - プログラムを追加するときに楽になる（「継承」による拡張性よい）
  - イベント処理の書き方が統一された。
  - 正規表現が使える。
  - プログラマのためのFlexでも採用されている。
  - パフォーマンスが良い、とされている
  - 新しいクラスやコンポーネントが使えてより便利に、より楽に（e4xなど）
  - 実行時に発生するエラー（ランタイムエラー）が表示される。など
- ちょっとデメリット
  - ActionScript 1.0 & 2.0 と比較すると、少し難易度が高い。

さぁ、今こそActionScript 3.0の世界へ！

- パフォーマンスが良いこと。
- 特に機能追加や同じプログラミングを繰り返すようなときに楽になる。
- 今まで何となく使っていたプロパティやメソッドに対する理解が深まる。
- 今まで使ったことのない組み込みのクラスのプロパティやメソッドの理解も早くなる。

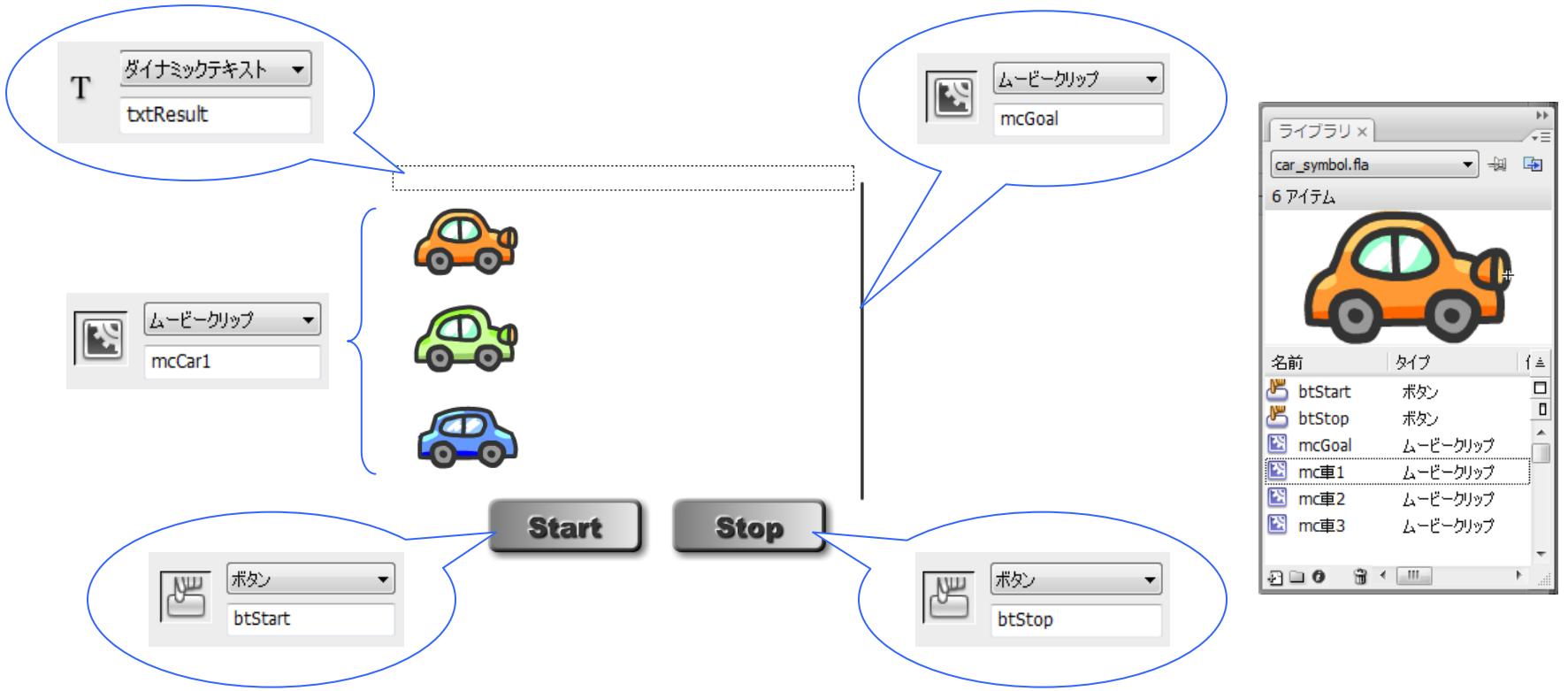
- さぁ、今こそAS3.0の世界へ！

ActionScript 3.0 プログラミングをしてみよう

熱闘！カーチェイスゲーム

Adobe

実習の概要



- 3台の「車」が競争するゲーム（競争を見るだけですが・・）を作ります。
- 従来のプログラムの書き方から、AS3.0のオブジェクト指向の書き方まで段階的にプログラムを書いていきます。

イベント処理の復習①

- 本題に入る前に、イベント処理の復習をしましょう。

- 「mc車」ムービークリップを、左から右へ自動移動するスクリプトを書く。
  - 復習（1） クリップアクション
    - ［実習ファイル：00_reviewフォルダ・EventReview1.flaファイル］
    - 「mc車」ムービークリップを選択してスクリプトを書く。
  - 復習（2） mcCarムービークリップ内のタイムラインに書く
    - ［実習ファイル：00_reviewフォルダ・EventReview2.flaファイル］

イベント処理の復習②

- mcCar ムービークリップを、左から右へ自動移動するスクリプトを書く。
  - 復習（3） 先ほどの実習を、addEventLister()を使った書き方に変更する
    - ［実習ファイル：00_reviewフォルダ・EventReview3.flaファイル］
- AS3.0のイベント処理

【追加対象のインスタンス名】.**addEventListener** (**イベント名** , **処理する関数名**);

イベント名

- 文字列（String型）で設定する。
- イベント名を文字列で直接指定できる。
  - （例）"click", "enterFrame"
- 定数でも指定できる。
  - （例）MouseEvent.CLICK, Event.ENTER_FRAME

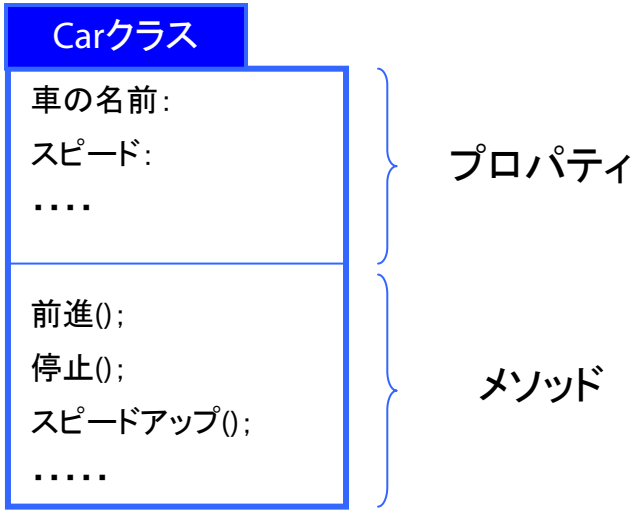
処理する関数名

```
function xEnterFrame ( evt: Event ) : void {
    this.x += 10;
}
```

- 引数は、該当するイベント型を受けるようにする。
- AS3.0からは、型宣言を絶対する必要があるため、関数の引数と戻り値の型の指定を忘れない。

オブジェクト指向の概要

- クラス



- クラスを作る。
- クラスは、「設計書」のようなもの。
- クラスは、プログラムの世界の中（=コンピューター上の仮想世界）での、「1つのモノ」を表す。
- クラスをプログラムして、ムービークリップ に関連づけたり、「インスタンス化」ということをして使う。
- クラスは、以下の2つで構成される。
  - **プロパティ**: クラスの中で定義する変数のこと。数値や文字列などでそのクラスの状態や性質を保持する。
  - **メソッド** : クラスの中で定義する関数のこと。動き（動作）に関する内容を記述する。
- プロパティやメソッドは、1クラスの中にいくつあってもかまわない。

オブジェクト指向の概要

- インスタンス
  - クラスはあくまでも「設計書」なので、動かせない。動かすためには設計書を元にして作った「実態」が必要である。
  - この実態が「インスタンス（またはオブジェクトという）」
  - 通常のプログラムでは、インスタンスは、以下のようにして作る。
    - **var** mc1:MovieClip = **new** MovieClip();
  - しかし、FlashのMoveClipでは、以下の画面でクラスとMoveClipを関連づけすることで、インスタンス化をしなくてもよい。

オブジェクト指向を理解する

- 携帯電話

ディスプレイサイズ	176 x 208 px
主な仕様	
色深度	16 ビット
ピクセルタイプ	RGB 16 565
OS	Generic 1.0
プラットフォーム	Generic 1.0
機能パック	1
サイズ (mm)	53 x 109 x 23
重さ (g)	126
キャリア	
複数	

デバイス機能	
FSCommand	通常
電子メール	○
SMS	○
MMS	○
SVG	○
バッテリーレベル	10
信号のレベル	10
CodePage	-
ループ	○
パフォーマンスインデックス	1.00

架電

①番号を入力する。
②通話ボタンを押す。
③相手の着信待ち。
④着信したら、通話。
・・・・・

着信

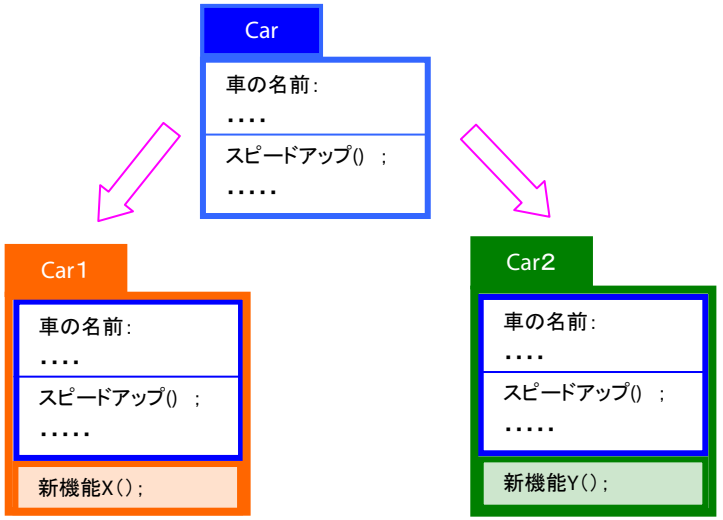
①相手の番号を表示。
②着信待ち。
③10秒以内に通話ボタンを押したら、通話。
④10秒経過したら、留守電応答開始。
・・・・・

オブジェクト指向を理解する

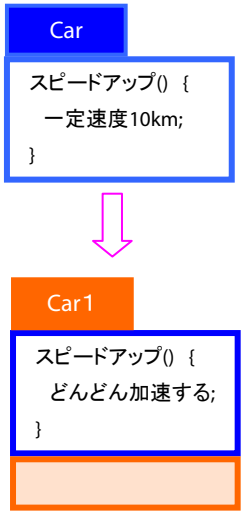
- 桃太郎

オブジェクト指向の概要

- 継承
  - 新しくクラスを作るときに、違うクラスのプロパティとメソッドを引き継いで作ることができる。
  - 新しいクラスで、プロパティやメソッドをコピペしなくてもいい。
  - 同じプロパティやメソッドを持つクラスで修正が必要なときは、親クラスでの修正でよい場合がある。



- オーバーライド
  - 継承した先でメソッドの中の命令を上書きして、違う動作をさせることができる。



実習（1）今までのやり方でプログラムする

- 実習の概要
  - ActionScriptの記述
    - 車の動作は、「mc車1」～「mc車2」ムービークリップ内にフレームアクション記述。
    - 車をスタートさせたり、停止させたりは、メインタイムラインのフレームアクション記述。
  - 手順
    - まずは、プログラムを単純にするため「mc車1」だけプログラムします。
    - 「mc車1」分がプログラムできたら、「mc車2」「mc車3」に同じ内容をコピーします。
      - 車の動作はみんな同じだからです。
  - 5つのステップに分割して、ステップバイステップで実習を行います。
    - 実習（1）－1 「車を発進させる」
    - 実習（1）－2 「車を停止させる」
    - 実習（1）－3 「ゴールしたら車を停止させる」
    - 実習（1）－4 「ゴールした車の着順表示」
    - 実習（1）－5 「すべての車に反映させて、完成」

実習（1）今までのやり方でプログラムする

- 実習（1）－1
  「車を発進させる」
  - 実習ファイル
    01_car_symbolフォルダ・car_symbol00.fla ファイル

  1. 「mc車1」ムービークリップ内のフレームアクション
     - 車のスピードはランダムに決める。
     - メソッド定義　xStart()
     - メソッド定義　xEnterFrame()
  2. メインタイムラインのフレームアクション
     - addEventListener()を使用して、btStart（[start]ボタン）インスタンスへのイベント追加。
     - xStartClick()関数を定義し、関数内でmcCar1インスタンス（「mc車1」シンボル）の xStart()メソッドの呼び出し。

- 実習（1）－2
  「車を停止させる」
  - 実習ファイル
    01_car_symbolフォルダ・car_symbol01.fla ファイル

  1. 「mc車1」ムービークリップ内のフレームアクション
     - メソッド定義 xStop()
  2. メインタイムラインのフレームアクション
     - addEventListener()を使用して、btStop（[stop]ボタン）インスタンスへのイベント追加。
     - xStopClick()関数を定義し、関数内でmcCar1インスタンス（「mc車1」シンボル）の xStop()メソッドの呼び出し。

実習（1）今までのやり方でプログラムする

- 実習（1）－3
「ゴールしたら車を停止させる」
  - 実習ファイル
01_car_symbolフォルダ・car_symbol02.fla ファイル

1. 「mc車1」ムービークリップ内のフレームアクション
   - xEnterFrame() メソッドに goalX インスタンスに車が到達したとき(※)、xStop()メソッドを呼び出すようにする。

     （※）x座標で判断する。

- 実習（1）－4
「ゴールした車の着順表示」
  - 実習ファイル
01_car_symbolフォルダ・car_symbol03.fla ファイル

1. メインタイムラインのフレームアクション
   - xAddResultText()関数の追加
2. 「mc車1」ムービークリップ内のフレームアクション
   - xEnterFrame() メソッドのif文内で、メインタイムラインの xAddResultText() 関数を呼び出す。

実習（1）今までのやり方でプログラムする

- 実習（1）－５
「すべての車に反映させて、完成」
  - 実習ファイル
01_car_symbolフォルダ・car_symbol04.fla ファイル

1. 「mc車1」ムービークリップ内のフレームアクションを、そのまま「mc車2」と「mc車3」にコピーする
2. メインタイムラインのフレームアクション
    1. xStartClick()関数内でmcCar2, mcCar3それぞれのxStart()メソッドの呼び出し。
    2. StopClick()関数内でmcCar2, mcCar3それぞれのxStop()メソッドの呼び出し。

実習（2） カスタムクラスの導入

- 実習の概要
  - ActionScriptの記述
    - 車の動作は、カスタムクラスに記述。それを「mc車1」～「mc車3」に関連づける。

- 手順
  - [実習ファイル： 02_car_symbolフォルダ・car_symbol2.fla ファイル]

1. Carクラスの作成
2. 「mc車1」～「mc車3」ムービークリップ内にフレームアクションを全部削除する
3. Carクラスを「mc車1」～「mc車3」ムービークリップに関連づける。
4. 実行して、動作を確認する。メインタイムライン・フレームアクションの記述。

実習（3） メソッドのオーバーライドによる機能拡張

- 実習の概要
  - さらにオブジェクト指向の性質を使って、機能拡張をしてみる
  - それぞれの車で「スピード感」を変えよう。
    - 「スピード感」を演出しているのが、xGetSpeed()メソッド。
    - これを継承したクラスでオーバーライドして、違う「スピード感」になるようにする。
  - ActionScriptの記述
    - Carクラスを継承した、CarA.asクラスとCarB.asクラスを作成し、「メソッドのオーバーライド」でそれぞれ違う機能になるようにプログラムする。
    - それぞれのクラスを「mc車2」と「mc車3」ムービークリップに関連付ける。

- 手順
  - [実習ファイル： 03_car_symbol フォルダ・car_symbol3.fla ファイル]

1. Carクラスを継承した、CarA.asクラスを作成する。
2. CarA.asクラスのxGetSpeed()メソッドをオーバーライドし、10を返す（「一定速度」ということですね。）
3. Carクラスを継承した、Carb.asクラスを作成する。
4. CarB.asクラスのxGetSpeed()メソッドをオーバーライドし、以下の式の計算結果を返す。（ランダムに変速します。）
   Math.floor(Math.random() * 6) + 5;
5. CarA.asとCarB.asクラスをそれぞれ、「mc車2」と「mc車3」ムービークリップにそれぞれ関連づける。
6. カーチェイスを楽しむ。（※賭レース禁止）

今後に役立つTips

- ActionScript 3.0基本アクションコンポーネントの使い方
  - http://www.adobe.com/jp/devnet/flash/articles/67ws_basic_components.html

おわりに

- AS2.0は、AS3.0に移行する途中です。
- ぜひ、ここで少しずつやって、モノにしましょう。

ありがとうございました。

Adobe

イベント処理の復習

（00_review フォルダ）

復習（１）クリップアクション

EventReview1.fla ファイル
mc 車１シンボル　クリップアクション

```
1:      onClipEvent (enterFrame) {
2:          this._x += 10;
3:      }
```

復習（２）mcCarムービークリップ内のタイムラインに書く

EventReview2.fla ファイル
mc 車１シンボル　クリップアクション

```
1:      this.onEnterFrame = function  () {
2:          xEnterFrame();
3:      }
4:
5:      function xEnterFrame(){
6:          this._x += 10;          //進むスピード
7:      }
```

復習（３）　先ほどの実習を、addEventLister()を使った書き方に変更する

EventReview3.fla ファイル
mc 車１シンボル　Action:フレーム１

```
1:      this.addEventListener(Event.ENTER_FRAME, xEnterFrame);
2:
3:      function xEnterFrame(evt:Event):void{
4:          this.x += 10;           //進むスピード
5:
6:          //(注)MovieClip._x プロパティは、AS3.0 では使えない。
7:
8:      }
```

実習（１） 今までのやり方でプログラムする

(01_car_symbol フォルダ)

実習（１）－１ 「車を発進させる」

car_symbol00.fla ファイル

1.「mc 車 1」ムービークリップ内のフレームアクション

```
1:      //「mc 車 1」内:フレームアクション
2:      //プロパティ定義
3:      var speed:uint;
4:
5:      //親をムービークリップとして保持
6:
7:      //スピード値をランダムに設定
8:      speed = Math.floor(Math.random() * 10) + 3
9:
10:     //メソッド定義
11:     function xStart():void {
12:         this.addEventListener(Event.ENTER_FRAME, xEnterFrame);
13:     }
14:
15:     function xEnterFrame(evt:Event):void {
16:         this.x += speed;
17:     }
18:
```

2. メインタイムラインのフレームアクション

```
1:      //フレームアクション
2:      //インスタンスの生成
3:      //ゴールの X 座標をメインタイムラインの変数に
4:
5:      //イベントハンドラの設定
6:      btStart.addEventListener(MouseEvent.CLICK, xStartClick);
7:
8:      //関数定義
9:      function xStartClick(evt:MouseEvent):void {
10:         mcCar1.xStart();
11:     }
12:
13:     //ゴール時のテキスト追加
```

実習（１）－２　「車を停止させる」

car_symbol01.fla ファイル

1.「mc 車 1」ムービークリップ内のフレームアクション

```
1:      //「mc 車 1」内:フレームアクション
2:      //プロパティ定義
3:      var speed:uint;
4:
5:      //親をムービークリップとして保持
6:
7:      //スピード値をランダムに設定
8:      speed = Math.floor(Math.random() * 10) + 3
9:
10:     //メソッド定義
11:     function xStart():void {
12:          this.addEventListener(Event.ENTER_FRAME, xEnterFrame);
13:     }
14:
15:     function xStop():void {
16:          this.removeEventListener(Event.ENTER_FRAME, xEnterFrame);
17:     }
18:
19:     function xEnterFrame(evt:Event):void {
20:          this.x += speed;
21:     }
```

2. メインタイムラインのフレームアクション

```
1:      //フレームアクション
2:      //インスタンスの生成
3:      //ゴールの X 座標をメインタイムラインの変数に
4:
5:      //イベントハンドラの設定
6:      btStart.addEventListener(MouseEvent.CLICK, xStartClick);
7:      btStop.addEventListener(MouseEvent.CLICK, xStopClick);
8:
9:      //関数定義
10:     function xStartClick(evt:MouseEvent):void {
11:          mcCar1.xStart();
12:     }
13:
14:     function xStopClick(evt:MouseEvent):void {
15:          mcCar1.xStop();
16:     }
17:
18:     //ゴール時のテキスト追加
```

実習（１）－３　「ゴールしたら車を停止させる」

car_symbol02.fla ファイル

1. 「mc 車 1」ムービークリップ内のフレームアクション

```
1:      //「mc 車 1」内:フレームアクション
2:      //プロパティ定義
3:      var theParent:MovieClip;
4:      var speed:uint;
5:
6:      //親をムービークリップとして保持
7:      theParent = parent as MovieClip;
8:      //スピード値をランダムに設定
9:      speed = Math.floor(Math.random() * 10) + 3
10:
11:     //メソッド定義
12:     function xStart():void {
13:          this.addEventListener(Event.ENTER_FRAME, xEnterFrame);
14:     }
15:
16:     function xStop():void {
17:          this.removeEventListener(Event.ENTER_FRAME, xEnterFrame);
18:     }
19:
20:     function xEnterFrame(evt:Event):void {
21:          this.x += speed;
22:          if (this.x > theParent.goalX){
23:               xStop();
24:          }
25:     }
```

2. メインタイムラインのフレームアクション

```
1:      //フレームアクション
2:      //インスタンスの生成
3:      //ゴールの X 座標をメインタイムラインの変数に
4:      var goalX:Number = mcGoal.x;
5:
6:      //イベントハンドラの設定
7:      btStart.addEventListener(MouseEvent.CLICK, xStartClick);
8:      btStop.addEventListener(MouseEvent.CLICK, xStopClick);
9:
10:     //関数定義
11:     function xStartClick(evt:MouseEvent):void {
12:         mcCar1.xStart();
13:     }
14:     function xStopClick(evt:MouseEvent):void {
15:         mcCar1.xStop();
16:     }
17:
18:     //ゴール時のテキスト追加
```

実習（１）－４　「ゴールした車の着順表示」

car_symbol03.fla ファイル

1. メインタイムラインのフレームアクション

```
1:      //フレームアクション
2:      //インスタンスの生成
3:      //ゴールの X 座標をメインタイムラインの変数に
4:      var goalX:Number = mcGoal.x;
5:      var goalNum:uint = 0;
6:
7:      //イベントハンドラの設定
8:      btStart.addEventListener(MouseEvent.CLICK, xStartClick);
9:      btStop.addEventListener(MouseEvent.CLICK, xStopClick);
10:
11:     //関数定義
12:     function xStartClick(evt:MouseEvent):void {
13:          mcCar1.xStart();
14:     }
15:
16:     function xStopClick(evt:MouseEvent):void {
17:          mcCar1.xStop();
18:     }
19:
20:     //ゴール時のテキスト追加
21:     function xAddResultText(theName):void {
22:          goalNum++;
23:          txtResult.appendText(goalNum + "位:" + theName + "  ");
24:     }
```

2.「mc 車 1」ムービークリップ内のフレームアクション

```
1:      //「mc 車 1」内:フレームアクション
2:      //プロパティ定義
3:      var theParent:MovieClip;
4:      var speed:uint;
5:
6:      //親をムービークリップとして保持
7:      theParent = parent as MovieClip;
8:      //スピード値をランダムに設定
9:      speed = Math.floor(Math.random() * 10) + 3
10:
11:     //メソッド定義
12:     function xStart():void {
13:         this.addEventListener(Event.ENTER_FRAME, xEnterFrame);
14:     }
15:     function xStop():void {
16:         this.removeEventListener(Event.ENTER_FRAME, xEnterFrame);
17:     }
18:
19:     function xEnterFrame(evt:Event):void {
20:         this.x += speed;
21:         if (this.x > theParent.goalX){
22:             theParent.xAddResultText(this.name);
23:             //ゴールしたインスタンスのインスタンス名を表示
24:             xStop();
25:         }
26:     }
```

実習（１）－５　「すべての車に反映させて、完成」

car_symbol04.fla ファイル
メインタイムラインのフレームアクション

```
1:      //フレームアクション
2:      //インスタンスの生成
3:      //ゴールの X 座標をメインタイムラインの変数に
4:      var goalX:Number = mcGoal.x;
5:      var goalNum:uint = 0;
6:
7:      //イベントハンドラの設定
8:      btStart.addEventListener(MouseEvent.CLICK, xStartClick);
9:      btStop.addEventListener(MouseEvent.CLICK, xStopClick);
10:
11:     //関数定義
12:     function xStartClick(evt:MouseEvent):void {
13:         mcCar1.xStart();
14:         mcCar2.xStart();
15:         mcCar3.xStart();
16:     }
17:
18:     function xStopClick(evt:MouseEvent):void {
19:         mcCar1.xStop();
20:         mcCar2.xStop();
21:         mcCar3.xStop();
22:     }
23:
24:     //ゴール時のテキスト追加
25:     function xAddResultText(theName):void {
26:         goalNum++;
27:         txtResult.appendText(goalNum + "位:" + theName + "  ");
28:     }
```

実習（２）　カスタムクラスの導入

（02_car_class1 フォルダ）

car_class1.fla ファイル
（01_car_symbol¥car_symbol.fla ファイル の ActionScript:フレーム1 と 同等なので省略）

Car.as ファイル

```
1:      package{
2:          import flash.display.MovieClip;
3:          import flash.events.Event;
4:
5:          public class Car extends MovieClip{
6:                  //プロパティ定義
7:                  protected var theParent:MovieClip;
8:                  protected var speed:uint;
9:
10:                 //コンストラクタ
11:                 function Car() {
12:                         //親をムービークリップとして保持
13:                         theParent = parent as MovieClip;
14:                         //スピード値をランダムに設定
15:                         speed = Math.floor(Math.random() * 10) + 3
16:                 }
17:
18:                 //メソッド定義
19:                 public function xStart():void {
20:                         this.addEventListener(Event.ENTER_FRAME, xEnterFrame);
21:                 }
22:
23:                 public function xStop():void {
24:                         this.removeEventListener(Event.ENTER_FRAME, xEnterFrame);
25:                 }
26:
27:                 function xEnterFrame(evt:Event):void {
28:
29:                         this.x += speed;
30:
31:                         if (this.x > theParent.goalX){
32:                                 theParent.xAddResultText(this.name);
33:                                 //ゴールしたインスタンスのインスタンス名を表示
34:                                 xStop();
35:                         }
36:                 }
37:         }
38:     }
```

実習（３）　メソッドのオーバーライドによる機能拡張

（03_car_class2 フォルダ）

car_class2.fla ファイル
（01_car_symbol¥car_symbol.fla ファイル の ActionScript:フレーム1 と 同等なので省略）

Car.as ファイル
（02_car_class1¥Car.as ファイルと同等なので省略）

CarA.as ファイル

```
1:      package{
2:          public class CarA extends Car{
3:              //スピード設定用
4:              override protected function xGetSpeed():uint {
5:                  return 10;
6:              }
7:          }
8:      }
```

CarB.as ファイル

```
1:      package{
2:          public class CarB extends Car{
3:              //スピード設定用
4:              override protected function xGetSpeed():uint {
5:                  return Math.floor(Math.random() * 6) + 5;
6:              }
7:          }
8:      }
```