情報処理技法
(マルチメディアと表現)2

第9回
インタラクティブなFlash

人間科学科コミュニケーション専攻

白銀 純子

第9回の内容

- Flashでボタン作成(続き)
- インタラクティブなFlash

ボタン作成(続き)～アニメーションの切り替え～

アニメーションの切り替え(1)

1. アニメーションのFlashのファイルを複数用意
   - 「.swf」の形式でファイルに保存しておく
2. 新しいキャンバスを作成
   - アニメーションのキャンバスよりも大きなキャンバスにすること
3. アニメーションを表示させる領域を、四角形で描画
4. 3. で描画した四角形を選択し、「修正」→「シンボルに変換」
5. 表示されたウィンドウで「ムービークリップ」を選択し、名前を入力
6. 四角形を選択し、「プロパティ」ウィンドウの「インスタンス名」の欄に「animeDisplay」と入力

アニメーションの切り替え(2)

6. ボタンシンボルを、アニメーションの個数分キャンバスに配置
7. 6. で配置したボタンを1つずつ選択し、「プロパティ」ウィンドウの「インスタンス名」の欄に名前を設定
   - ❖ 半角英数で、それぞれ違った名前にすること
8. レイヤを1つ作成し、6. で配置したボタンの上に画面表示用の名前を配置
9. レイヤを1つ作成し、新しいレイヤの1フレーム目を選択
10. 「ウィンドウ」→「アクション」で表示されたウィンドウに、アクションスクリプトの内容を記述

アニメーションの切り替え(3)

```
var anime:Loader = new Loader();
anime.x = animeDisplay.x;
anime.y = animeDisplay.y;
removeChild(animeDisplay);

function animeLoad(e) {
    if(anime.root) {
        removeChild(anime);
        removeChild(animeDisplay);
    }
    e.target.content.stop();
}
function animeStart(e) {
    addChild(anime);
    addChild(animeDisplay);
    anime.mask = animeDisplay;
    e.target.content.play();
}
```

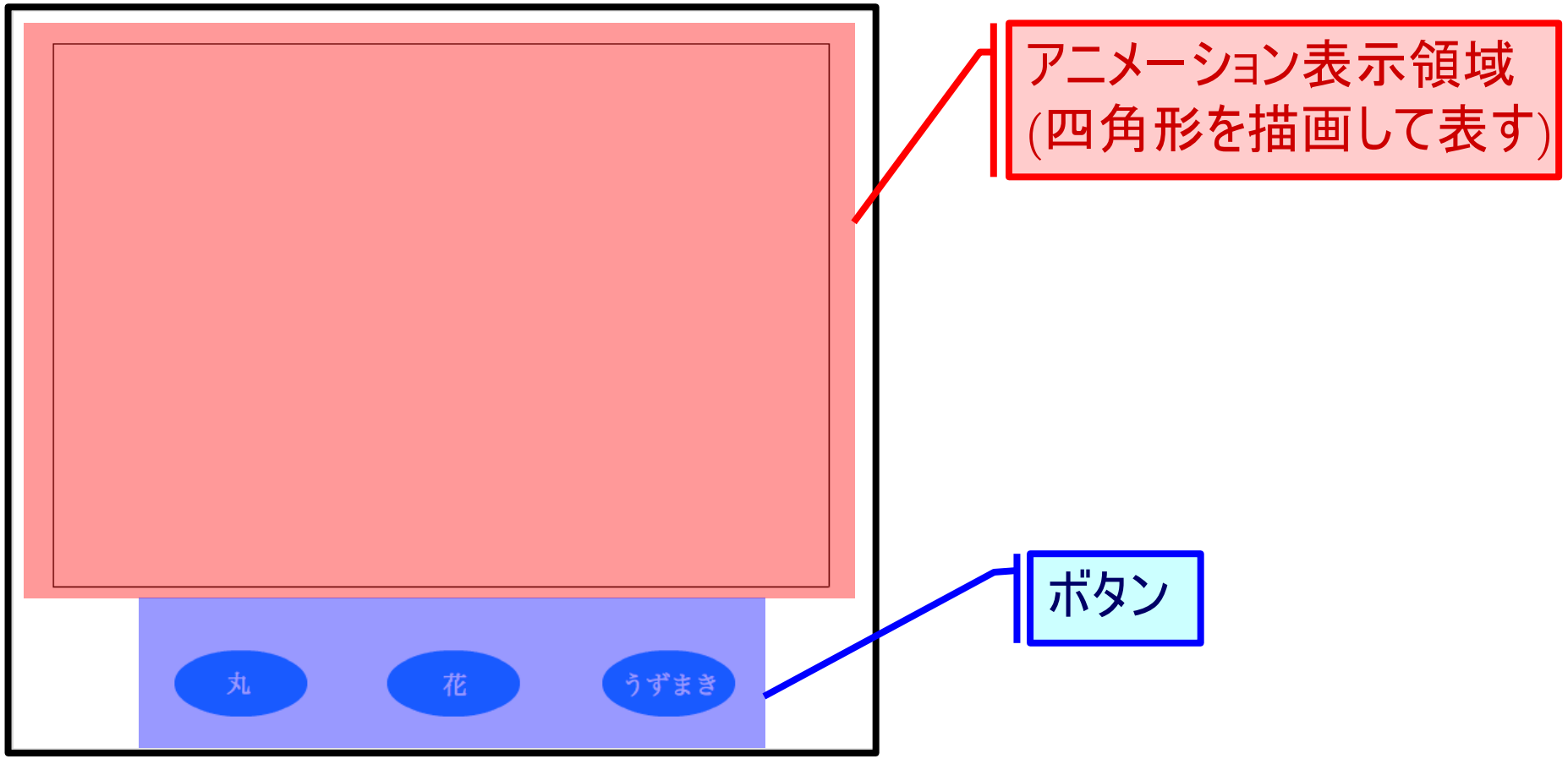
```
var animeList:Dictionary = new Dictionary();
animeList[1つ目のボタンの名前] = "アニメーションFlashのファイル名1";
animeList[2つ目のボタンの名前] = "アニメーションFlashのファイル名2";
animeList[3つ目のボタンの名前] = "アニメーションFlashのファイル名3";

function loadContent(e) {
    if(animeList[e.target]) {
        anime.load(new URLRequest(animeList[e.target]));
    }
}
anime.contentLoaderInfo.addEventListener(Event.INIT, animeLoad);
anime.contentLoaderInfo.addEventListener(Event.COMPLETE, animeStart);
addEventListener(MouseEvent.CLICK, loadContent);
```

アニメーションの
個数分記述

アニメーションの切り替え(4)

- 1.～10. まで行った後のキャンバスの状態



アニメーションの切り替え(5)

- 「チェック」ボタンを押すと、エラーがあるかどうかが表示
  - タイムラインが表示されているところに新しいタブが追加され、表示される
  - エラーがあれば修正する(エラーの先頭にある数字が、
    エラーが存在する行番号なので、その行をよく見ること)

- 一旦「.fla」形式でファイルに保存し、「制御」→「ムービープレビュー」で動作確認
  - アニメーションのFlashのファイルと同じフォルダに保存すること
  - 動作確認時にも、エラーが出ることもあるので、その場合はまた修正すること

検索ボックス

検索ボックス(1)

- キーワードを入力し、「検索」ボタンを押すと、Googleでの検索結果を表示するFlash
  1. 検索キーワードの入力フィールドと検索ボタンを配置するレイヤを1つ作成
  2. 「ウィンドウ」→「コンポーネント」を選択
  3. 表示されたウィンドウで、「User Interface」をクリックし、選択肢から「TextInput」をキャンバス上にドラッグ&ドロップ
  4. 入力フィールドの隣にボタンを配置し、ボタンに画面表示用の名前を配置
  5. ボタンを選択し「プロパティ」ウィンドウの「インスタンス名」の欄に「searchText」と入力
     - 半角英数で入力すること

検索ボックス(2)

- キーワードを入力し、「検索」ボタンを押すと、Googleでの検索結果を表示するFlash
  6. レイヤを1つ作成し、1つ目のフレームを選択
  7. 「ウィンドウ」→「アクション」で表示されたウィンドウに、アクションスクリプトの内容を記述

```
this.searchBut.addEventListener(MouseEvent.CLICK, searchGoogle);

function searchGoogle(e:MouseEvent):void {
    var googleSite:String = "http://www.google.co.jp/search?q=" + this.searchText.text;
    var url:URLRequest = new URLRequest(googleSite);
    navigateToURL(url);
}
```

※HTMLファイルに埋め込んだとき、Webに公開していない状態(「file://～」のURL)で確認すると検索できないが、Webに公開した状態(「http://～」のURL)では検索可能

インタラクティブなFlash

インタラクティブって?

- マウスやキーボードの操作に反応すること
  - マウスのカーソルを上に置くと動き出す絵
  - ボタンを押すと、場面が切り替わるアニメーション
  - etc.

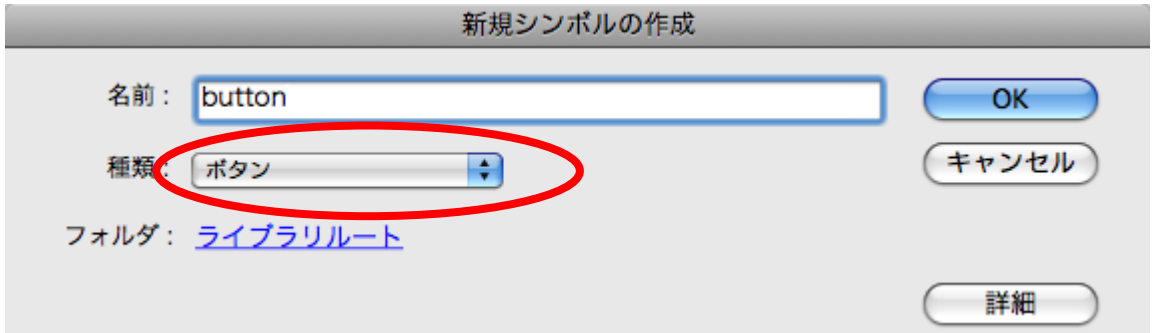
マウスカーソルを重ねると動く

作成方法概要

1. 動く絵をボタンとして作成する
2. ボタンの「アップ」、「オーバー」、「ダウン」のうち、どの場面で動かし始めたいかを決める
3. 動かしたい場面にアニメーションの設定をする

ボタン作成(1)

1. 「挿入」→「新規シンボル」を選択
2. 「名前」の欄にボタンの名前を入力、「種類」の欄で「ボタン」を選択
   - タイムラインが、「アップ」・「オーバー」などのボタンモードに



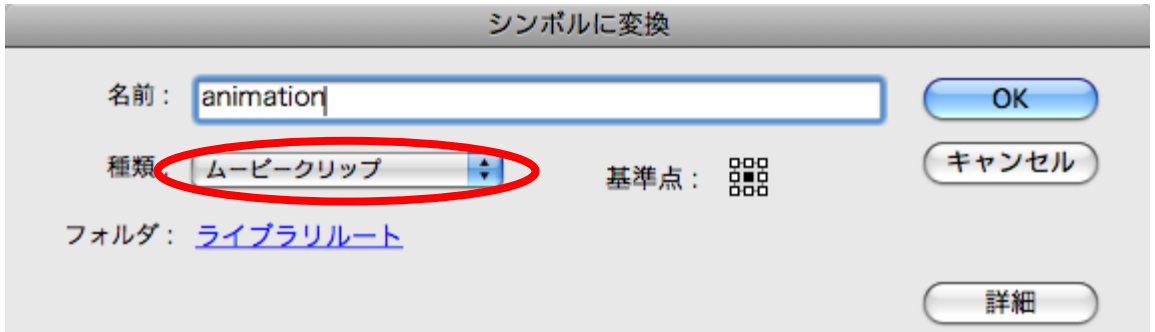
3. 「アップ」の状態の絵を作成

ボタン作成(2)

4. タイムラインの「オーバー」の下を選択し、その後マウス長押し→「キーフレームの挿入」
   - 絵の内容は変更しない
5. タイムラインの「ダウン」の下を選択し、その後マウス長押し→「キーフレームの挿入」
   - 絵の内容は変更しない

アニメーション設定(1)

1. 絵を選択し、「オーバー」の下を選択している状態で、メニューバーの「修正」→「シンボルに変換」
   - ダウンの状態でアニメーションしたいときは、ダウンの下を選択
2. 「名前」の欄にアニメーションの名前を入力、「種類」の欄で「ムービークリップ」を選択

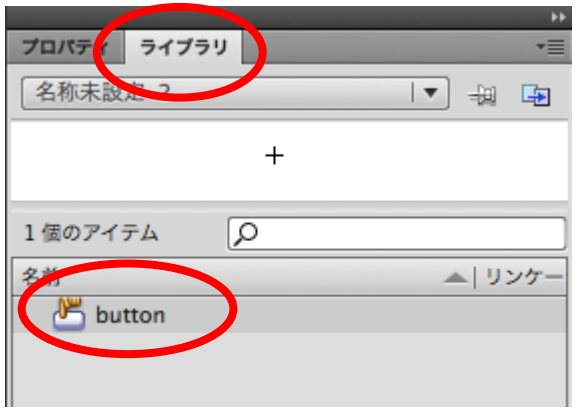


アニメーション設定(2)

3. 「編集」→「ドキュメントの編集」で、もとのキャンバスを表示
4. 「編集」→「シンボルの編集」でアニメーションの設定
5. 通常の方法でアニメーションを作成
6. 「編集」→「ドキュメントの編集」で、もとのキャンバスを表示

アニメーション設定(2)

7. 周辺にあるウィンドウの「ライブラリ」タブをクリックすると、作成したボタンシンボルが存在

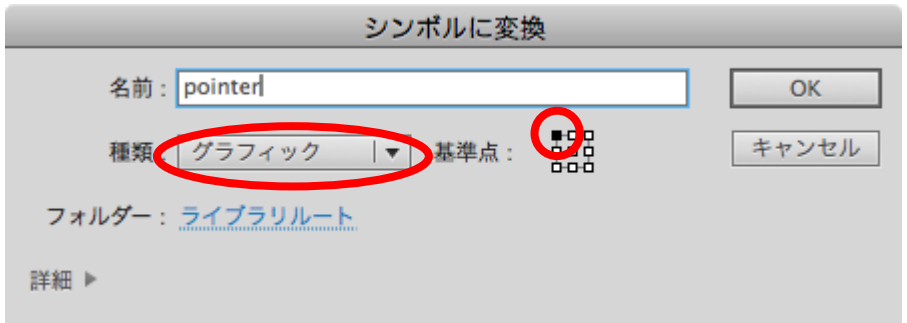


6. ボタンシンボルをキャンバス上にドラッグ&ドロップ
7. 「制御」→「ムービープレビュー」→「プレビュー」で動作確認
   - or メニューバーの「ファイル」→「書き出し」→「ムービーの書き出し」でファイルに保存し、Safariで開く
   - or メニューバーの「制御」→「ムービープレビュー」→「Device Central」を選択する
     - 「Device Central」だと、他の方法で確認できないものも確認できることがある

マウスを追って動く

グラフィックシンボル作成

1. 何か絵を作成
2. 描いた絵を選択ツールで選択
3. 「修正」→「新規シンボル」を選択
   - 「名前」の欄に何か名前を入力
   - 「種類」の欄で「グラフィック」を選択
   - 「基準点」で左上端を選択



マウスを追って動く設定(1)

4. タイムラインの1フレーム目を選択し、「ウィンドウ」→「アクション」
5. 表示されたウィンドウに、マウスを追うオブジェクトの動作を記述

```
stage.addChild(this);
function moveAlong(e) {
    x = e.stageX - 300;
    y = e.stageY - 200;
    e.updateAfterEvent();
}
stage.addEventListener(MouseEvent.MOUSE_MOVE, moveAlong);
```

マウスカーソルとオブジェクトとの距離
- xの方が横方向の距離
- yの方が縦方向の距離

6. 記述したら、動作確認

マウスでドラッグする

ムービーシンボル作成

1. 「挿入」→「新規シンボル」を選択
   - 「名前」の欄に何か名前を入力
   - 「種類」の欄で「ムービークリップ」を選択

2. 何か絵を作成
3. 「編集」→「ドキュメントの編集」で、もとのキャンバスを表示

ドラッグして動く設定(1)

1. 周辺にあるウィンドウの「ライブラリ」タブをクリックすると、作成したグラフィックシンボルが存在
2. グラフィックシンボルをキャンバス上にドラッグ&ドロップ
3. 選択ツールでシンボルを選択し、「プロパティ」タブの「インスタンス名」の欄に何か名前を入力
   - 名前は半角英数で
4. タイムラインの1フレーム目を選択し、「ウィンドウ」→「アクション」

ドラッグして動く設定(2)

5. 表示されたウィンドウに、マウスドラッグで動くオブジェクトの動作を記述

3. の手順でつけた「インスタンス名」の名前

```
function pressObject(e) {
    stage.addEventListener(MouseEvent.MOUSE_UP, releaseObject);
    XXXX.startDrag();
}
function releaseObject(e) {
    stage.removeEventListener(MouseEvent.MOUSE_UP, releaseObject);
    XXXX.stopDrag();
}
XXXX.addEventListener(MouseEvent.MOUSE_DOWN, pressObject);
```

6. 記述したら、動作確認

FlashのWebページへの掲載

FlashのWebページへの掲載

- 以下のタグをHTMLファイルに書く
  - HTMLファイルとFlashのファイルは同じフォルダ内に置く
  - Flashのファイルは「.swf」のファイルを使う

```
<object width="横幅の大きさ" height="縦幅の大きさ">
<param name="movie" value="Flashファイル名">
<param name="quality" value="high">
<embed src="Flashファイル名" quality="high"
width="横幅の大きさ" height="縦幅の大きさ" type="application/x-shockwave-flash">
</embed>
</object>
```