

Example: Animation portfolio piece

This example is designed to give you a first opportunity to see how you can piece together bits of ActionScript into a complete, if not ActionScript-heavy, application. The animation portfolio piece is an example of how you could take an existing linear animation (for example, a piece created for a client) and add some minor interactive elements appropriate for incorporating that animation into an online portfolio. The interactive behavior that we'll add to the animation will include two buttons the viewer can click: one to start the animation, and one to navigate to a separate URL (such as the portfolio menu or the author's home page).

The process of creating this piece can be divided into these main sections:

1. Prepare the FLA file for adding ActionScript and interactive elements.
2. Create and add the buttons.
3. Write the ActionScript code.
4. Test the application.

Preparing to add interactivity

Before we can add interactive elements to our animation, it's helpful to set up the FLA file by creating some places to add our new content. This includes creating actual space on the Stage where buttons can be placed, and also creating "space" in the FLA file for keeping different items separate.

[⚡To set up your FLA for adding interactive elements:](#)

1. If you don't already have a linear animation to which you'll be adding interactivity, create a new FLA file with a simple animation such as a single motion tween or shape tween. Otherwise, open the FLA file containing the animation that you're showcasing in the project, and save it with a new name to create a new working file.
2. Decide where on the screen you'll want the two buttons to appear (one to start the animation and one to link to the author portfolio or home page). If necessary, clear or add some space on the Stage for this new content. If the animation doesn't already have one, you might want to create a splash screen on the first frame (you'll probably want to shift the animation over so it starts on Frame 2 or later).
3. Add a new layer, above the other layers in the Timeline, and rename it **buttons**. This will be the layer where you'll add the buttons.
4. Add a new layer, above the buttons layer, and name it **actions**. This will be where you'll add ActionScript code to your application.

Creating and adding buttons

Next we'll need to actually create and position the buttons that will form the center of our interactive application.

⇒To create and add buttons to the FLA:

1. Using the drawing tools, create the visual appearance of your first button (the "play" button) on the buttons layer. For example, you might draw a horizontal oval with text on top of it.
2. Using the Selection tool, select all the graphic parts of the single button.
3. From the main menu, choose Modify > Convert To Symbol.
4. In the dialog box, choose Button as the symbol type, give the symbol a name, and click OK.
5. With the button selected, in the Property inspector give the button the instance name **playButton**.
6. Repeat steps 1 through 5 to create the button that will take the viewer to the author's home page. Name this button **homeButton**.

Writing the code

The ActionScript code for this application can be divided into three sets of functionality, although it will all be entered in the same place. The three things the code needs to do are:

- Stop the playhead as soon as the SWF file loads (when the playhead enters Frame 1).
- Listen for an event to start the SWF file playing when the user clicks the play button.
- Listen for an event to send the browser to the appropriate URL when the user clicks the author home page button.

⇒To create code to stop the playhead when it enters Frame 1:

1. Select the keyframe on Frame 1 of the actions layer.
2. To open the Actions panel, from the main menu, choose Window > Actions.
3. In the Script pane, enter the following code:
4. `stop();`

⇒To write code to start the animation when the play button is clicked:

1. At the end of the code entered in the previous steps, add two empty lines.
2. Enter the following code at the bottom of the script:
3. `function startMovie(event:MouseEvent):void`
4. `{`
5. `this.play();`
6. `}`

This code defines a function called `startMovie()`. When `startMovie()` is called, it causes the main timeline to start playing.

7. On the line following the code added in the previous step, enter this line of code:
8. `playButton.addEventListener(MouseEvent.CLICK, startMovie);`

This line of code registers the `startMovie()` function as a listener for `playButton`'s `click` event. In other words, it makes it so that whenever the button named `playButton` is clicked, the `startMovie()` function is called.

≡To write code to send the browser to a URL when the home page button is clicked:

1. At the end of the code entered in the previous steps, add two empty lines.
2. Enter this code at the bottom of the script:
3. `function gotoAuthorPage(event:MouseEvent):void`
4. `{`
5. `var targetURL:URLRequest = new URLRequest("http://example.com/");`
6. `navigateToURL(targetURL);`
7. `}`

This code defines a function called `gotoAuthorPage()`. This function first creates a `URLRequest` instance representing the URL `http://example.com/`, and then passes that URL to the `navigateToURL()` function, causing the user's browser to open that URL.

8. On the line following the code added in the previous step, enter this line of code:
9. `homeButton.addEventListener(MouseEvent.CLICK, gotoAuthorPage);`

This line of code registers the `gotoAuthorPage()` function as a listener for `homeButton`'s click event. In other words, it makes it so that whenever the button named `homeButton` is clicked, the `gotoAuthorPage()` function is called.

Testing the application

At this point, the application should be completely functional. Let's test it to make sure that's the case.

≡To test the application:

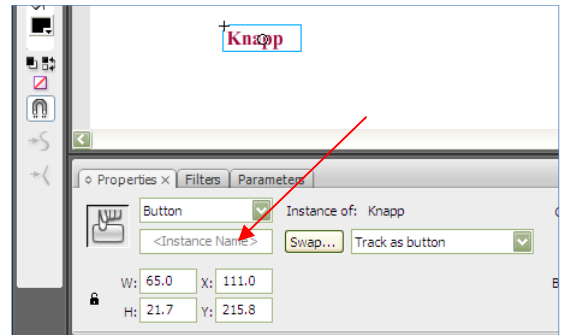
1. From the main menu, choose Control > Test Movie. Flash creates the SWF file and opens it in a Flash Player window.
2. Try both the buttons to make sure they do what you expect them to.
3. If the buttons don't work, here are some things to check for:
 - o Do the buttons both have distinct instance names?
 - o Do the `addEventListener()` method calls use the same names as the buttons' instance names?
 - o Are the correct event names used in the `addEventListener()` method calls?
 - o Is the correct parameter specified for each of the functions? (Both should have a single parameter with the data type `MouseEvent`.)

All of these and most other possible mistakes should give an error message either when you choose the Test Movie command or when you click the button. Look in the Compiler Errors panel for compiler errors (the ones that happen when you first choose Test Movie), and check the Output panel for run-time errors (errors which happen while the SWF is playing--such as when you click a button).

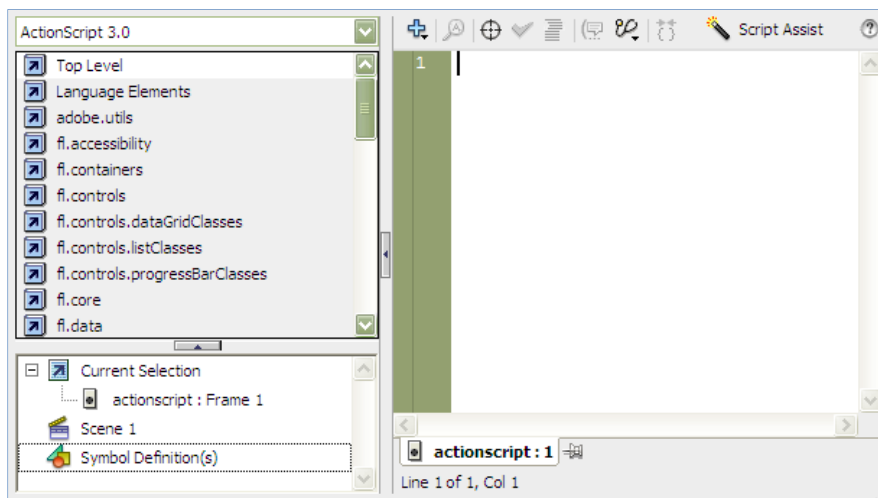
Flash – skapa interaktiva knappar

I den här övningen kommer du att lära dig att förflytta dig i tidslinjen med olika knappar genom att använda Action Script 3.0.

1. Använd Rectangel Tool för att skapa en knapp. Dra ut en rektangel och lägg sedan på en text på rektangeln.
2. Markera både rektangeln och texten med Selection Tool och tryck F8, eller Modify och Convert to Symbol. Döp din knapp och bocka i alternativet button.
3. Upprepa ovanstående steg så att du har fyra knappar.
4. Markera din första knapp med Selection Tool och gå ner till properties rutan för att ange ett Instance Name åt knappen, exempelvis btn 1. Detta görs för att kunna kalla på knappen när du skall lägga på Action Script på den. Upprepa samma procedur för varje knapp men tänk på att alla knappar måste ha olika instance name. (Jag har valt att kalla mina knappar btn_boll, btn_heart, btn_video och btn_musik.)



5. Nu är du färdig med dina knappar och då låser du det lagret som knapparna ligger i. Skapa ett nytt lager och döp det till Action. Ställ dig på den första keyframen och tryck F9 för att få upp Actions menyn.



Det vänstra fönstret kallas Actions Toolbox och innehåller alla olika klassers bibliotek. Du navigerar genom de olika biblioteken genom att klicka på de olika klasserna. Du kan lägga till klasser, metoder och egenskaper till Action Script genom att dra dem till kodfönstret. Actions Toolbox är designad för att fungera tillsammans med

Script Assist verktyget. Script Assist är en hjälpfunktion för att underlätta när du skall skapa Action Script.

6. Eftersom vi inte vill att flashfilmen skall börja spela automatiskt är det första som du skall skriva in ett stopp som gör att filmen kommer på frame 1. I actions fönstret skriv:

```
stop();
```

7. Skapa ett nytt lager och gör en animering (motion eller shape tween) som startar på frame två. När du trycker på din första knapp (i mitt fall boll) så ska animationen för bollen starta. För att göra detta krävs det några rader kod i Action Script. Ställ dig på Actions lagret och tryck F9. Hoppa ner ett par rader under stopfunktionen du skapade tidigare. Lägg till följande kod:

```
function boll(event:MouseEvent):void
{
    gotoAndPlay(2);
}
btn_boll.addEventListener(MouseEvent.CLICK, boll);
```

Nu har du skapat en funktion som i det här exemplet kallas **boll**, boll kan bytas ut till det som passar din animation bäst. Funktionen innehåller en händelse, **event:MouseEvent**, som talar om att det är vid en "mushändelse" som funktionen skall utföras. Händelsen sätts alltid inom parentes och följs av **:void** samt **{ }**. Det är vad som skall hända som sätts inom "måsvingarna" i det här fallet **gotoAndPlay(2);** som innebär man i tidslinjen går till frame två och spelar filmen därifrån. För att det skall hända något när du trycker på din knapp måste du koppla funktionen du skapat till knappen. Det gör du genom att ange det instance name som du gav knappen, följt av en **addEventListener** som "lyssnar" efter händelser som sker. I parentesen anger du **MouseEvent.CLICK** som talar om att "mushändelsen" i det här fallet sker vid "klick" med musen. Sist anger du funktionen, i mitt fall **boll**. Nu är du färdig med den första knappen. Testkör med **ctrl + enter**. Du skall nu kunna klicka på din första knapp och komma vidare till din animation

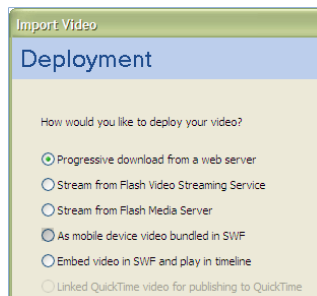
Lägga till och streama video

1. Fortsätt att arbeta i samma flashfil, skapa ett nytt lager som du exempelvis döper till movie. Om din animation som du skapade ovan slutar på frame 40 så ställ dig på frame 41 i ditt lager movie, skapa



en keyframe och gå till **file → import → import movie** för att importera din film. Leta dig fram till din film genom browse.

2. Välj det första alternativet: Progressive download from a webserver.



3. I detta steg

anger du hur spelaren skall se ut. Det finns massor

med alternativ, ange skinunderAll.swf, du kan även bestämma vilken färg du skall ha.

När du fixat till spelaren som du vill att den ska se ut och klickar på nästa så renderas filmen. Det kan ta ett bra tag beroende på hur stor filen är. När det är färdigt finns din importerade fil på din keyframe i movielagret. För att koppla filmen till knapp två som du skapade i föregående övning upprepa steg 7. Observera att istället för gotoAndPlay(); används här **gotoAndStop(81)**; Anledningen är att vid gotoAndPlay(); hoppar man fram i tidslinjen till angiven frame och sen spelar den vidare frame för frame. Videon du importerat ligger endast i en frame och för att filmen skall spelas måste du hoppa fram och stanna på angiven frame för att filmen skall gå i gång. Koden som skall läggas till



```
function video(event:MouseEvent):void
```

```
    {  
        gotoAndStop(81);  
    }
```

```
btn_video.addEventListener(MouseEvent.CLICK, video);
```

4. Din fullständiga Action Script kod bör nu se ut som följer:
stop();

```
function boll(event:MouseEvent):void
```

```
    {  
        gotoAndPlay(2);  
    }
```

```
btn_boll.addEventListener(MouseEvent.CLICK, boll);
```

```
function video(event:MouseEvent):void
```

```
    {  
        gotoAndStop(81);  
    }
```

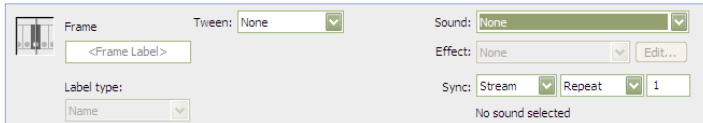
```
btn_video.addEventListener(MouseEvent.CLICK, video);
```

5. Testkör din fil, du ska nu kunna klicka på knappen för filmen och se den.

Lägga till ljud

Om du vill att ljud skall spela när du klickar på en knapp går du tillväga på samma sätt som i tidigare övningar. För att lägga till ljud till en animering så går du tillväga på följande vis.

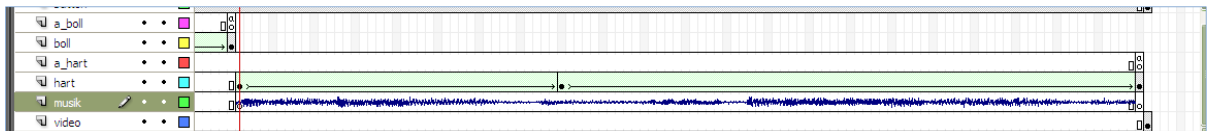
1. Antingen lägger du till ljud till din första animering som du skapade på knapp 1 (boll) eller skapar du ett nytt lager och gör en ny animering i det lagret. (Då måste du koppla ihop animeringen med en knapp som i första övningen.)
2. Skapa ett nytt lager som ljudet skall ligga i. Gör en keyframe på samma frame som din animering startar på. En ljudfil kan endast läggas i keyframes.
3. Välj **file** → **import** → **import to library** leta upp din ljudfil och importera den. Filen ligger nu i biblioteket.
4. Ställ dig på keyframen där ljudfilen skall läggas och gå ner till properties. I dropdownmenyn för sound kan du nu välja den ljudfil du lagt in i biblioteket. Synk skall vara inställd på stream så att



ljudet synkas med din animering även om användaren sitter på en långsamt dator/uppkoppling. Under effekts kan du göra olika inställningar för hur

ljudfilen skall uppföra sig. Om du exempelvis inte vill att ljudet skall spela från början kan du ställa in var ljudfilen skall spela ifrån.

5. I ljudfilens lager ställ dig på den sista framen som stämmer överens med framen där animeringen slutar och tryck F6. Detta gör att ljudfilen spelar under hela animeringen. Det bör se ut som nedan:



Du ser tydligt hur hela ljudfilen löper över hela animeringen.

Nu borde allt vara klappat och klart... Näe inte riktigt, det är en liten detalj kvar. Om du testkör din film nu så kommer du att märka när du trycker på den första knappen du skapade att när animeringen kört klart så stoppar den inte utan kör vidare med nästföljande knappas animering. Detta måste vi fixa till och det gör vi med **stop();**

6. Ställ dig på ditt första lager med animering som du skapade, mitt var boll. Lägg till ett nytt lager och döp det till ex, a_boll.
7. Lägg till en keyframe på lager: a_boll i den sista framen som stämmer överens med framen där animeringen slutar Tryck F9 och skriv in **stop();**

Nu kommer animeringen boll att stanna när den har kört färdigt och du måste trycka på en ny knapp för att starta en ny animering. Steg sju görs på de lager som har en animering, när det gäller ljudfilen så slutar den på samma keyframen som animeringen så du behöver inte lägga två lager med stopfunktion. Videolagret behöver inte ha ett stop eftersom du där använde **gotoAndStop()** vilket gör att spelaren stannar på framen som angavs.

Har du tid över öva på att fixa till den/de sista knappen/knapparna du skapade!

Lycka till!

/Ylva