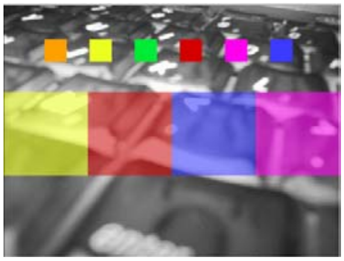
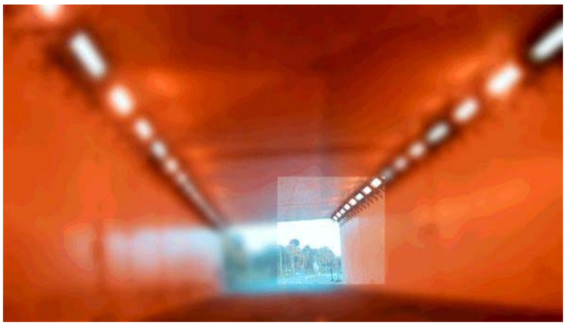
Further Literature (German)

- Ein schönes deutschsprachiges Buch mit ästhetisch ansprechenden Beispielen:

  Brendan Dawes, Flash ActionScript für Designer:
  DRAGSLIDEFADE, Markt&Technik 2002





1 Example Technology: Macromedia Flash & ActionScript

1.1 Multimedia authoring tools - Example Macromedia Flash

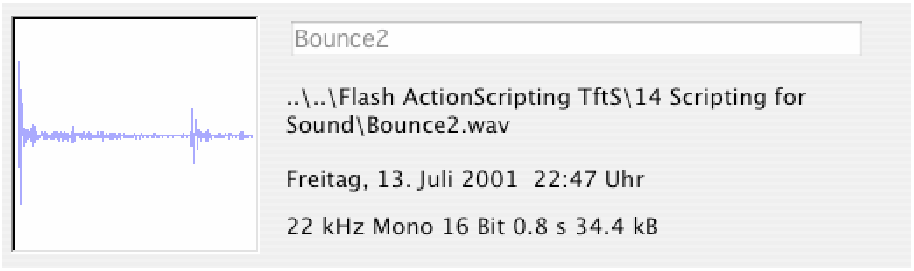
1.2 Elementary concepts of ActionScript
Scripting in General + „History" of ActionScript
Objects and Types in ActionScript
Animation with ActionScript

1.3 Interaction in ActionScript

1.4 Media classes in ActionScript

Literature:
Derek Franklin, Jobe Makar: Flash MX 2004 actionscript, Macromedia Press 2004

Sounds in the Library

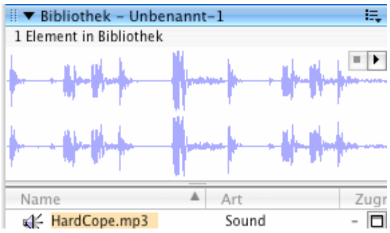

Bounce2

..\..\Flash ActionScripting TftS\14 Scripting for Sound\Bounce2.wav

Freitag, 13. Juli 2001 22:47 Uhr

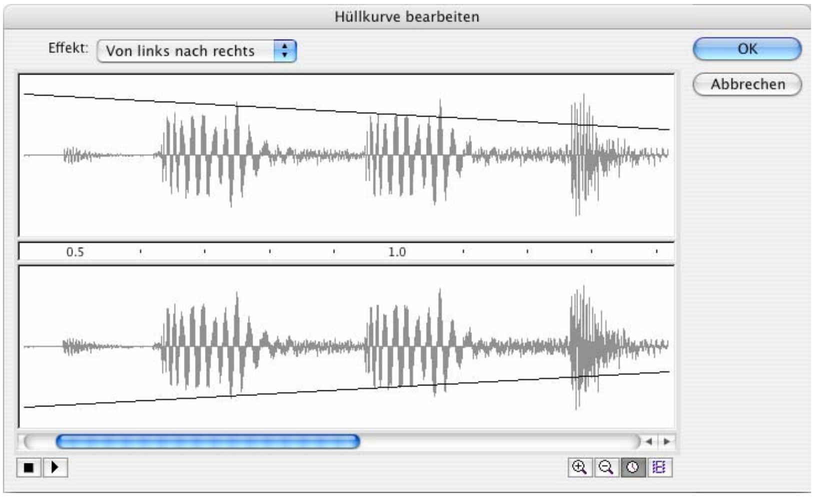
22 kHz Mono 16 Bit 0.8 s 34.4 kB

- Sounds are imported from a file (in Flash essentially WAV, MP3, AU)
  - Flash command: File -> Import -> Import into Library
- Sounds in the library are the raw material to be used in further design



Sound Processing in Authoring Tool

- Some simple effects can be created graphically



Sound Objects in Time-based Animations

- Sound object:
  - Encapsulates a (pre-produced) sound clip
- A sound object is associated with a specific timeline
  - Sound is played as the time in the timeline progresses
  - There may be many sounds in one presentation
    - » Main timeline
    - » Individual movie clip instance timelines
  - Sounds are mixed together
- Association of sound instance (from library) to timeline
  - Either graphically (e.g. dragging sound onto frame)
  - or using ActionScript method **`attachSound()`**

ActionScript Syntax for Sound Objects

- Creating a sound object:

```
var soundObjectName:Sound = new Sound(TargetClip);
```

  Example:

```
var mySound:Sound = new Sound(myMovieClip_mc);
```

  Omitting the *TargetClip:* Definition of global sound

- A Sound object is a *handle* like the Color object
- Controlling the sound’s volume:

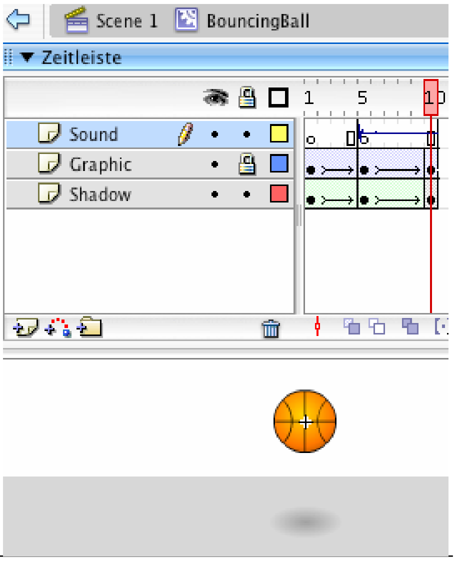
```
mySound.setVolume(50);
```

- Attaching a library sound:

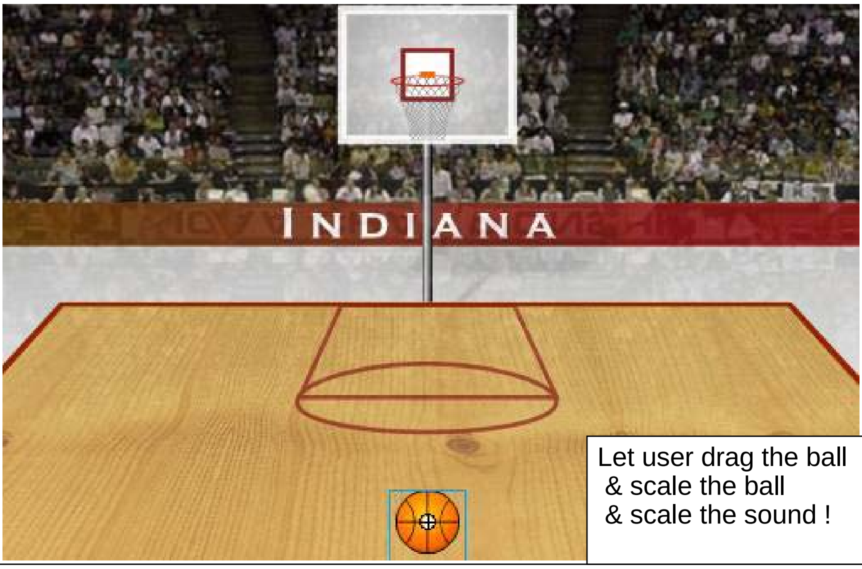
```
mySound.attachSound("rockMusic");
```

Example: A Bouncing Basketball

- Library contains the sound of the bouncing ball
- Movement of ball and coordinated change of shadow realised by tweening
- At the frame where ball touches ground (frame 5), sound is activated (e.g. through the object inspector)
- Sound is played from frame 5 till end of clip
  - Works only well with short sounds



Dragging the Ball over the Court



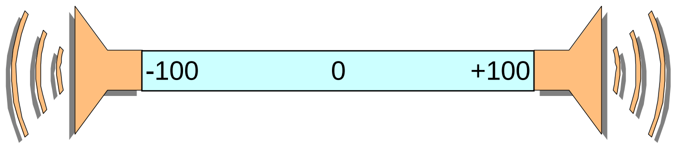
Let user drag the ball
& scale the ball
& scale the sound !

Dynamic Adjustment of Volume (and Scale)

```
var   bounce:Sound = new Sound(basketball_mc);
var   leftBoundary:Number = 60;
var   rightBoundary:Number = 490;
var   topBoundary:Number = 220;
var   bottomBoundary:Number = 360;
var   boundaryHeight:Number = bottomBoundary - topBoundary;

this.onMouseMove = function() {
  if (_xmouse > leftBoundary && _ymouse > topBoundary &&
        _xmouse < rightBoundary && _ymouse < bottomBoundary) {
      basketball_mc.startDrag(true);
      var topToBottomPercent = ((((_ymouse - topBoundary) /
        boundaryHeight) * 100) / 2) + 50;
      bounce.setVolume(topToBottomPercent);
      basketball_mc._xscale = topToBottomPercent;
      basketball_mc._yscale = topToBottomPercent;
  } else {
      stopDrag();
  }
}
```

Stereo Effect: “Panning”



- Panorama position or “balance”:
  - Relative volume of left and right stereo channel
  - Controls the perceived location of a monaural audio signal
- ActionScript (Class **`Sound`**):

  Method **`setPan(`*`relativeValue`*`)`**
  - Only left channel: –100
  - Only right channel: +100
  - Centered: 0

Example: Stereo Effect for Basketball

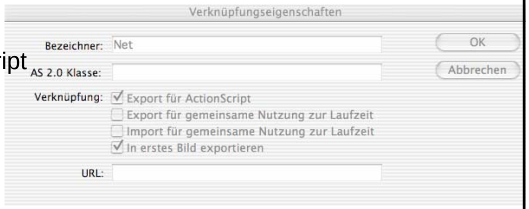
- Sound of bouncing ball draggable with mouse to left and right
  - According adjustment of sound balance

```
var   leftBoundary, rightBoundary,
      topBoundary, bottomBoundary...
var   boundaryHeight:Number = bottomBoundary - topBoundary;
var   boundaryWidth:Number = rightBoundary - leftBoundary;
var   quadrantSize:Number = boundaryWidth / 2;
var   centerPoint:Number = rightBoundary - quadrantSize;

this.onMouseMove = function() {
  if (_xmouse > leftBoundary && _ymouse > topBoundary &&
       _xmouse < rightBoundary && _ymouse < bottomBoundary) {
      ...;
      var panAmount =
       ((_xmouse - centerPoint) / quadrantSize) * 100;
      bounce.setPan(panAmount);
  }...
```

Dynamically Selected Sounds

- Sounds can be attached at runtime dynamically
  - as global sound and to movie clips
- Prerequisite in Flash:
  - Export library sound for ActionScript



Bezeichner: Net
AS 2.0 Klasse:
Verknüpfung: [✓ Export für ActionScript] Export für gemeinsame Nutzung zur Laufzeit; Import für gemeinsame Nutzung zur Laufzeit; [✓ In erstes Bild exportieren]
URL:

- Attaching a sound from library:
  Class **Sound: attachSound("*library name*");**
- Playing the sound:
  Class **Sound: start(*starttime, repetitions*); //time in secs**
  Class **Sound: stop();**

Example: Random Basketball Sounds

- On mouse click: Random number between 0 and 2
  - 0: score for “North Carolina” --> sound “boo” (Sound0)
  - 1: score for “Indiana” --> sound “cheer” (Sound1)
  - 2: no score --> sound “referee whistle” (Sound2)
  - Sound names chosen such that names can be computed from number (variable **`dynaSounds`**)
- In case of score:
  - Play “net sound”
  - Show basketball score animation (**`score_mc`**)
  - Update score fields of respective team (***team*_txt**)

Code for Random Basketball Sounds

```
var dynaSounds:Sound = new Sound();
var netSound:Sound = new Sound ();
...
this.onMouseDown = function() {
  var randomSound = random(3);
  dynaSounds.attachSound("Sound" + randomSound);
  dynaSounds.start(0, 1);
  if(randomSound == 0) {
      northCarolina_txt.text = Number(northCarolina_txt.text)
        + 2;
      netSound.attachSound("Net");
      netSound.start(0, 1);
      score_mc.gotoAndPlay("Score");
  } else if(randomSound == 1) {
      indiana_txt.text = Number(indiana_txt.text) + 2;
      netSound.attachSound("Net");
      netSound.start(0, 1);
      score_mc.gotoAndPlay("Score");
  }
}
```

Code for Silencing the Dynamic Sounds

- Sound to be switched off when any key is pressed:
  - *Listener* concept used
    (appropriate for events broadcasted to many recipients)

```
this.onKeyDown = function() {
   dynaSounds.stop();
}
Key.addListener(this);
```

Playing Video from Animations

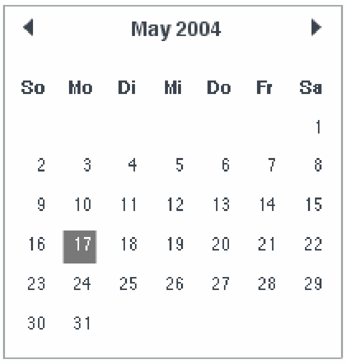
- Embedding video information into animation
  - Leads to very large files (SWF files in the case of Flash)
- External video clips:
  - Editable separately with specialized software
  - Progressive download: play during loading
  - Video played at its own frame rate, not at the rate of the animation
- Support for external video in Flash (MX 2004):
  - FLV (Flash Video) format
  - Converters from most well-known video formats to FLV exist
  - Special *Media Components* for easy integration of video
    - » MediaDisplay
    - » MediaController
    - » MediaPlayback (= MediaDisplay + MediaController)
  - Media component can also play back MP3 audio

Flash Components

- *Software component:* „A ***software component*** is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently and is subject to composition by third parties.“
  ECOOP 1996, Workshop on Component-oriented Programming
- *Flash component:* A reusable unit of Flash design and ActionScript programming with clearly specified parameters and methods. A Flash component encapsulates a ready-made solution that can be incorporated into third-party Flash applications.
- Components delivered with Flash (MX 2004, examples):
  - User Interface components:
    - » Button, CheckBox, ComboBox, DataGrid, DateChooser, Label, ProgressBar, ScrollPane. TextArea, TextInput, Window, ...
  - Data components:
    - » DataHolder, DataSet, WebServviceConnector, ...
  - Manager:
    - » PopUpManager, Depth Manager, ...
  - Media Components ...

Example Flash Component: Date Chooser

- Layout and basic behaviour pre-defined
- Component inspector allows customization, e.g.
  - Definition of string representation for days, months
  - Disabled days (not chosable)
  - Start day of week
- API allows dynamic ActionScript-based adaptation
  - E.g. setting selected date
- Components generate events


May 2004

So	Mo	Di	Mi	Do	Fr	Sa
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

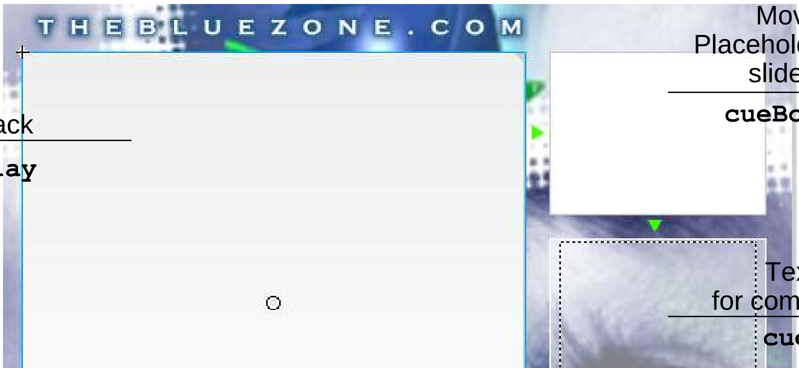
Events Generated by Media Components

- Various events are reported by Media Components to the surrounding application for flexible reaction:
  - Adjustments like change of volume
  - Media events like reaching end of media
  - User-defined events when reaching specific positions *(cue events)*

- Reaction to media events requires *Listener* objects, e.g.

```
var myListener:Object = new Object();
myListener.volume = function() {
  // actions to react on volume change
}
myMediaComponent.addEventListener(“volume”, myListener);
```

Example: Video with Event-Triggered Animation



Step 1: Setting Component Parameters

- Component parameters can be set
  - With the component inspector (authoring tool)
  - By script commands

```
display.autoPlay = true;
  // start playing immediately
display.activePlayControl = true;
  // display playback button as active
display.controllerPolicy = "on";
  // controls always visible
display.totalTime = 60;
  // total runtime to be used in playback progress bar
```

Step 2: Add Required Event Listeners

- Example:
  - Listener for “complete” event (i.e. end of video)
  - Automatically invokes a browser window with a given URL

```
var displayListener:Object = new Object();
displayListener.complete = function(){
  getURL("http://www.thebluezone.com");
}
display.addEventListener("complete", displayListener);
```

Step 3: Load External File

- Both filename and file extension are specified, since also MP3 files can be played
- Playback started
  - Automatically via auto-play parameter setting (as in the example)
  - When user presses “play” button in controller
  - Controlled by script

```
display.setMedia("bluezone.flv", "FLV");
```

Cue Points

- A *cue point* marks a specific point in time during media playback.
  - Cue points can be defined independently of the movie (in ActionScript)
  - When reaching a cue point, an event is fired which can be handled by ActionScript.

```
display.addCuePoint("0", 1);
display.addCuePoint("1", 8);
display.addCuePoint("2", 14);
display.addCuePoint("3", 31);
display.addCuePoint("4", 35);
display.addCuePoint("5", 53);
display.addCuePoint("6", 56);
display.addEventListener("cuePoint", displayListener);
displayListener.cuePoint = function(eventObj:Object){
  var index = Number(eventObj.target.name);
  loadMovie("cue" + index + ".jpg", "cueBox_mc");
  cue_txt.text = cueTextArray[index];
}
```

Cue Points in the Example

- Names of cue points chosen in a way such that conversion to number gives an index
- Two arrays of information to be displayed in the two extra windows
  - Still pictures
  - Text information

cue2.jpg

"Fluffy is crammed into dial-up pipe"

cueTextArray[2]

Flash Pattern: Names and Numbers

- **Problem:** Indexing and computing an index requires numbers to identify information instances. Storage in files and symbol identifiers require strings to identify information instances.
- **Solution:**
  - When a string is required to be used as an index: Choose a string representing a number and convert to number when required with function **`Number()`**
  - When a number is required to be used as a string: Compute an appropriate String by concatenating a base string with the number. Choose file names and identifiers appropriately.
- **Known Uses:**
  - String-to-Number: Cue point names in above example
  - Number-to-String: File names for Cue*X* pictures in above example; Sound names in Basketball example

Loading Variables

- Method loadVariables() to load data from external source
  - Load can take place from local file or from server over network (http)
- Special class **`LoadVars`** to maintain name/value pairs loaded from external source
  - Signals event when loaded completely
  - Example:

```
var container:LoadVars = new LoadVars();
container.load(...);
```

- String (URL) representation of loaded data (“form url-encoded”)
  - Example:

```
name=michael&age=23&phone=113344
```

XML Files in Flash

- A standard way for storing semi-structured data is XML
  - Built-in support in Flash
- Class **`XML`** for objects representing XML information
  - API for reading and manipulating tree representation: **`attributes(), childNodes(), hasChildNodes(), removeNode(), createElement(), insertBefore(), ...`**
- Typical methods for loading data:
  - **`load()   //load from a URL`**
  - **`send()   //send to a URL`**
  - **`sendAndLoad()`**