# 6 Programming with Sound



Literature:

W. McGugan, Beginning Game Development with Python and Pygame, Apress 2007

# Example: Slide Show with Background Music

- How to play back music while the program runs?
  - How to access the sound subsystem?
  - How to load a sound file?
    - » Supported file formats?
  - How to control playback?
- Sound playback always takes place in parallel to rest of program
  - Separate *thread* in program
  - Time container in *parallel* composition

# Example: Background Music - Python/Pygame (1)

```
import pygame
from pygame.locals import *
from sys import exit

background = pygame.Color(255,228,95,0)
sc_w = 356
sc_h = 356
music_file = "nancygroff_turntome.ogg"

pygame.init()
pygame.mixer.init(44100,-16,2,1024*4)

# Create program display area
screen = pygame.display.set_mode((sc_w,sc_h),0,32)
pygame.display.set_caption("Simple Slide Show")

# Set background color by drawing a rectangle
pygame.draw.rect
  (screen,background,pygame.Rect(0,0,sc_w,sc_h),0)

      ...contd.
```

# Sound Initialization

- Sound subsystem:
  - Gateway between program and operating system
    - » System specific (e.g. QuickTime library)
    - » Or abstraction layer (e.g. Pygame)
  - *Mixer* (name derived from audio mixer hardware)
    - » Common name e.g. in Pygame and Java Sound!
- Audio Format:
  - Sample rate / playback rate: samples/second
  - Sample size: bits
  - Stereo channels (mono=1, stereo=2)
  - Buffer size: number of samples buffered for playback
- Pygame mixer initialization defines playback properties:

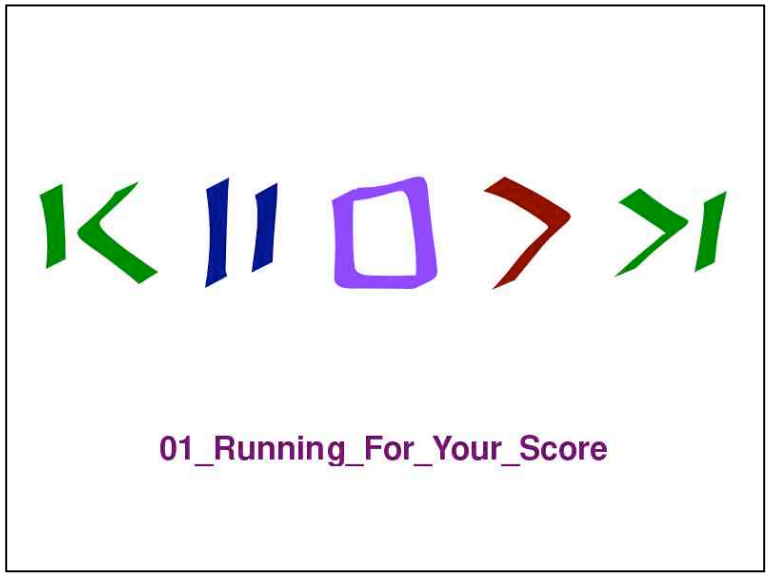
  ```
  pygame.mixer.init(44100,-16,2,1024*4)
  ```

  44100 samples/s, 16 bit samples (signed), stereo, 4k buffer

# Example: Background Music - Python/Pygame (2)

```
...(cont.)

# Load and play background music
pygame.mixer.music.load(music_file)
pygame.mixer.music.play()

# Load slide and show it on the screen
slide = pygame.image.load('pics/tiger.jpg').convert()
screen.blit(slide,(50,50))
pygame.display.update()
pygame.time.wait(4000)
...
# Load slide and show it on the screen
slide =
  pygame.image.load('pics/butterfly.jpg').convert()
...
pygame.time.wait(4000)
pygame.mixer.music.fadeout(3000)
```

# Example: Jukebox




```
...
 if button_pressed == "next":
   current_track =
      (current_track + 1) % max_tracks
   pygame.mixer.music.load(
       music_filenames[current_track] )
   if playing:
      pygame.mixer.music.play()
 elif button_pressed == "prev":
   if pygame.mixer.music.get_pos() > 3000:
      pygame.mixer.music.stop()
      pygame.mixer.music.play()
   else:
      current_track = (current_track - 1) % max_tracks
      pygame.mixer.music.load(
         music_filenames[current_track] )
      if playing:
         pygame.mixer.music.play()
 elif button_pressed == "pause":
   if paused:
      pygame.mixer.music.unpause()
      paused = False
   else:
      pygame.mixer.music.pause()
...
```

# 6 Programming with Sound

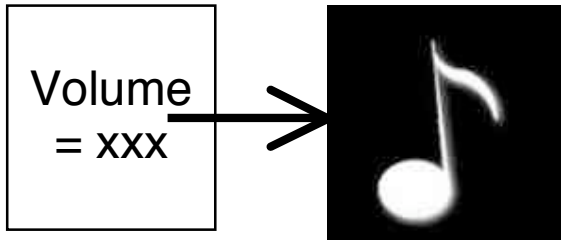


Literature:

W. McGugan, Beginning Game Development with Python and Pygame, Apress 2007

# Rendering Control Objects

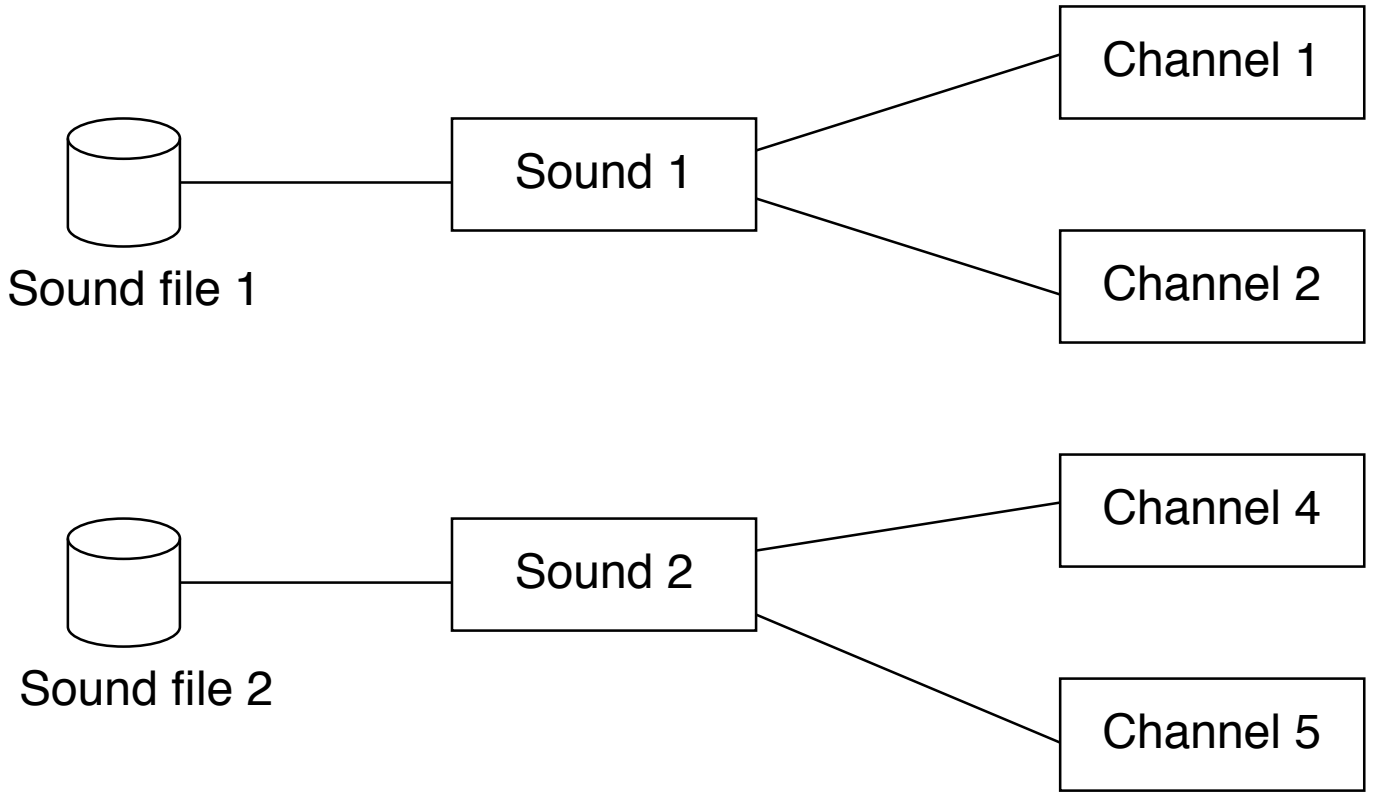
- General schema for using media objects in programs:
  - *Loading* creates an internal representation of the media content (from any source, internal or external)
  - *Rendering* creates an external representation of the media content form an internal representation
- Rendering process can be modified by parameter settings:
  - For images: e.g. compositing rules, clipping
  - For sound: e.g. volume, placement of mono source in stereo panorama
- Specific object representing rendering parameters: *rendering control*
  - Refers to media object (is a *handle* on the object)
  - Locally stores rendering parameters
- Examples:
  - *Channel* objects in Pygame
  - *SoundChannel* objects in ActionScript 3.0



# Channels in Pygame

- Channel:
  - One out of several sources that are mixed together by the sound card
  - **`play()`** method returns a Channel object (or **`None`** if all channels are busy)
- Limited number of channels
  - Number of channels can be set (**`pygame.mixer.set_num_channels`**
  - Channels are assigned to playing tasks automatically until maximum number is reached (all channels busy)
  - Channels for important audio information can be reserved (**`pygame.mixer.set_reserved`**)
- Typical methods for Channel objects:
  - Individual playback control (pause, play)
  - Volume control, for left and right speakers
  - Event handling for end of playing time
    - » Fire event at end of playing time
    - » Play queued sound object

# Multiple Sounds and Channels



# Asynchronous Playback

- Quiz question:
  What to we hear when this code is executed?

```
sound1 = pygame.mixer.Sound(soundfile)
channel1 = sound1.play()
channel2 = sound1.play()
channel3 = sound1.play()
```

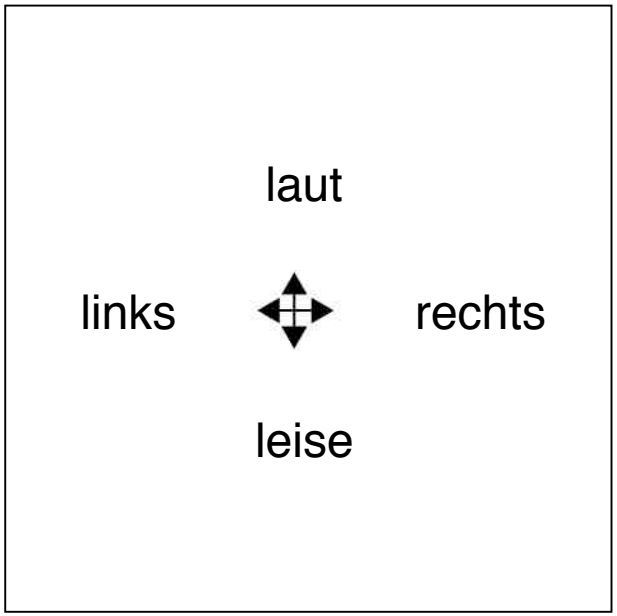
- The play() method triggers the start of playback only...

# Example: Setting Volume/Balance with Mouse (1)

```
SCREEN_SIZE = (480,480)
mouse_image_file = "arrowcursor.png"
loop_file = "GuitarLoop.wav"

import pygame
from pygame.locals import *

def run():
    pygame.mixer.init(44100,-16,2,1024*4)
    pygame.init

    screen = pygame.display.set_mode(SCREEN_SIZE, 0)
    pygame.display.set_caption("Sound Control")
    mouse_cursor =
      pygame.image.load(mouse_image_file).convert_alpha()

    loop_sound = pygame.mixer.Sound(loop_file)
    channel = loop_sound.play(-1)
```

# Example: Setting Volume/Balance with Mouse (2)

```
...
 def adjust_volume(x, y, width, height):
   vol = 1.0 -
     max(0,float(y)/(height-mouse_cursor.get_width()/2))
   pan = max(0,float(x)/(width-mouse_cursor.get_width()/2))
   right_volume = vol*pan
   left_volume = vol*(1.0 - pan)
   return (left_volume, right_volume)
...
```



# Example: Setting Volume/Balance with Mouse (1)

```
...
    while True:
       for event in pygame.event.get():
       if event.type == QUIT:
          return
       screen.fill((255,255,255))
       x, y = pygame.mouse.get_pos()
       x -= mouse_cursor.get_width()/2
       y -= mouse_cursor.get_height()/2
       screen.blit(mouse_cursor,(x,y))
       pygame.display.update()

       if channel is not None:
         left, right =
           adjust_volume(x,y,SCREEN_SIZE[0],SCREEN_SIZE[1])
         channel.set_volume(left,right)

if __name__ == "__main__":
    run()
```

# 6 Programming with Sound

6.1 Playing Music: Streaming

6.2 Controlling Sound with Objects

6.3 Sound Effects and Events 

Literature:

W. McGugan, Beginning Game Development with Python and Pygame, Apress 2007

R. van der Spuy: Foundation Game Design with Flash, Apress/Friends of ED 2009

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

# Event-Driven Sound

- In interactive programs and animations:
  - Sound as part of presentation
  - Needs to be synchronized with user interactions and animation progress
  - Several sounds may play synchronously
- Examples:
  - Sound triggered by collision detection in animation (bounce, crash)
  - Sound triggered by user input (keyboard beep)
  - Sound synchronized with animation (pitch or volume analog to movement)
- Sound triggering events may be explicit program events or just implicit (position in program code)

# Events Created by Sound System

- Specific conditions of the sound system may be made available as events to the programmer
  - Example: “End event” for playback in Pygame

    **`Channel.set_endevent(id)`**
    requests an event to be triggered when sound has finished playing. Appropriate identifier for event is given as parameter
- Examples for other events possibly created by sound system (not Pygame-specific):
  - External change of volume or other parameters
  - Playback reaching a certain intermediate position
  - Exceptional situations (e.g. too few channels)

# Example: Bouncing Balls (1)

From Pygame book (excerpt):

```
class Ball(object):

    def __init__(self, position, speed, image, bounce_sound):

        self.position = Vector2(position)
        self.speed = Vector2(speed)
        self.image = image
        self.bounce_sound = bounce_sound
        self.age = 0.0

    def update(self, time_passed):

        w, h = self.image.get_size()

        screen_width, screen_height = SCREEN_SIZE

        x, y = self.position
        x -= w/2
        y -= h/2

        ...
```

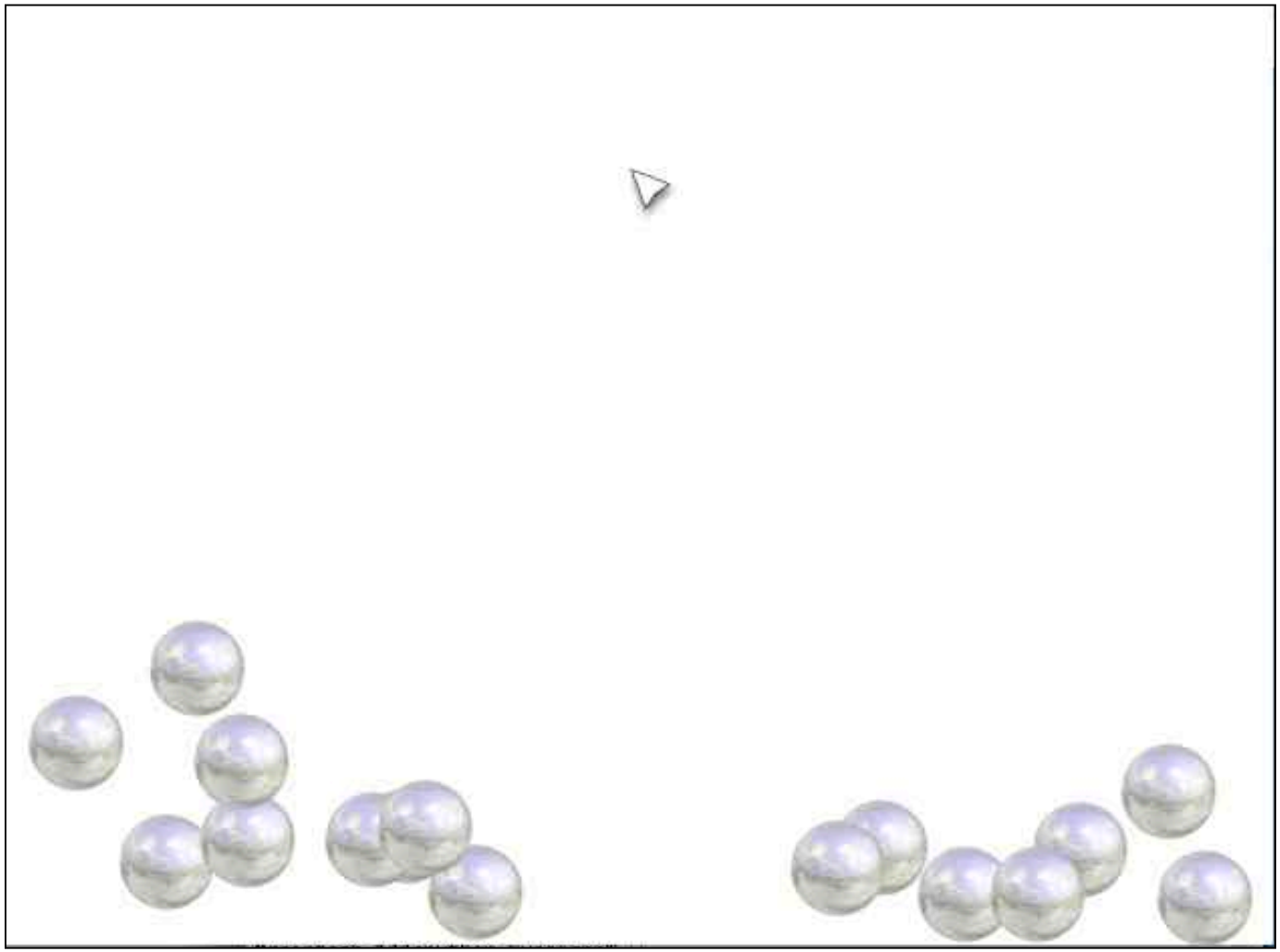
# Example: Bouncing Balls (2)

```
# Has the ball bounced?
bounce = False

# Has the ball hit the bottom of the screen?
if y + h >= screen_height:
    self.speed.y = -self.speed.y * BOUNCINESS
    self.position.y = screen_height - h / 2.0 - 1.0
    bounce = True

# Has the ball hit the left of the screen?
if x <= 0:
    self.speed.x = -self.speed.x * BOUNCINESS
    self.position.x = w / 2.0 + 1
    bounce = True

# Has the ball hit the right of the screen
elif x + w >= screen_width:
    self.speed.x = -self.speed.x * BOUNCINESS
    self.position.x = screen_width - w / 2.0 - 1
    bounce = True

# Do time based movement
self.position += self.speed * time_passed
# Add gravity
self.speed.y += time_passed * GRAVITY

if bounce:
    self.play_bounce_sound()

self.age += time_passed
```

# Example: Bouncing Balls (3)

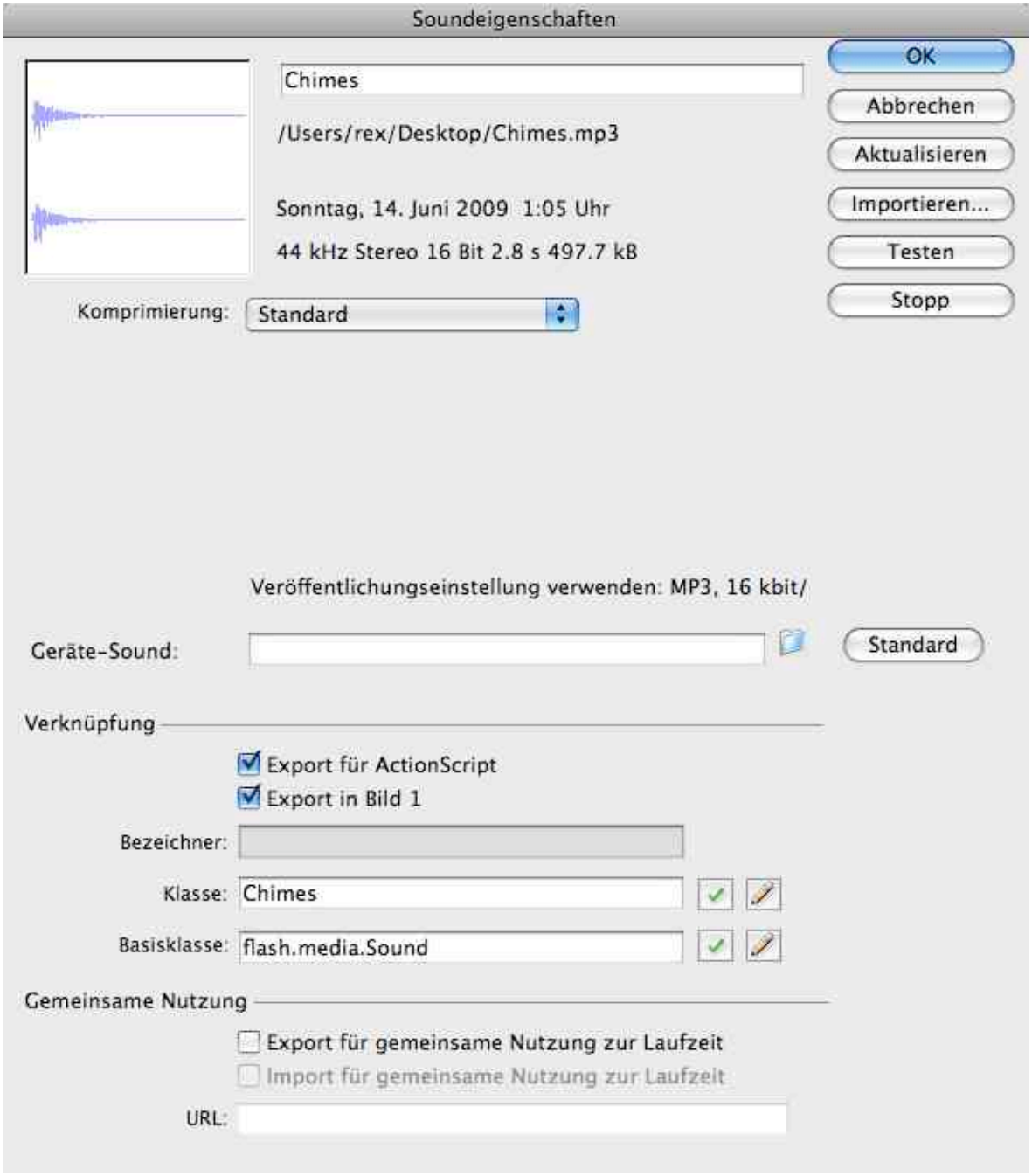
```
def stereo_pan(x_coord, screen_width):
    right_volume = float(x_coord) / screen_width
    left_volume = 1.0 - right_volume
    return (left_volume, right_volume)
...
class Ball(object):

    def play_bounce_sound(self):
       channel = self.bounce_sound.play()
       if channel is not None:
          left, right =
            stereo_pan(self.position.x, SCREEN_SIZE[0])
          channel.set_volume(left, right)
```

# Multiple Bouncing Balls



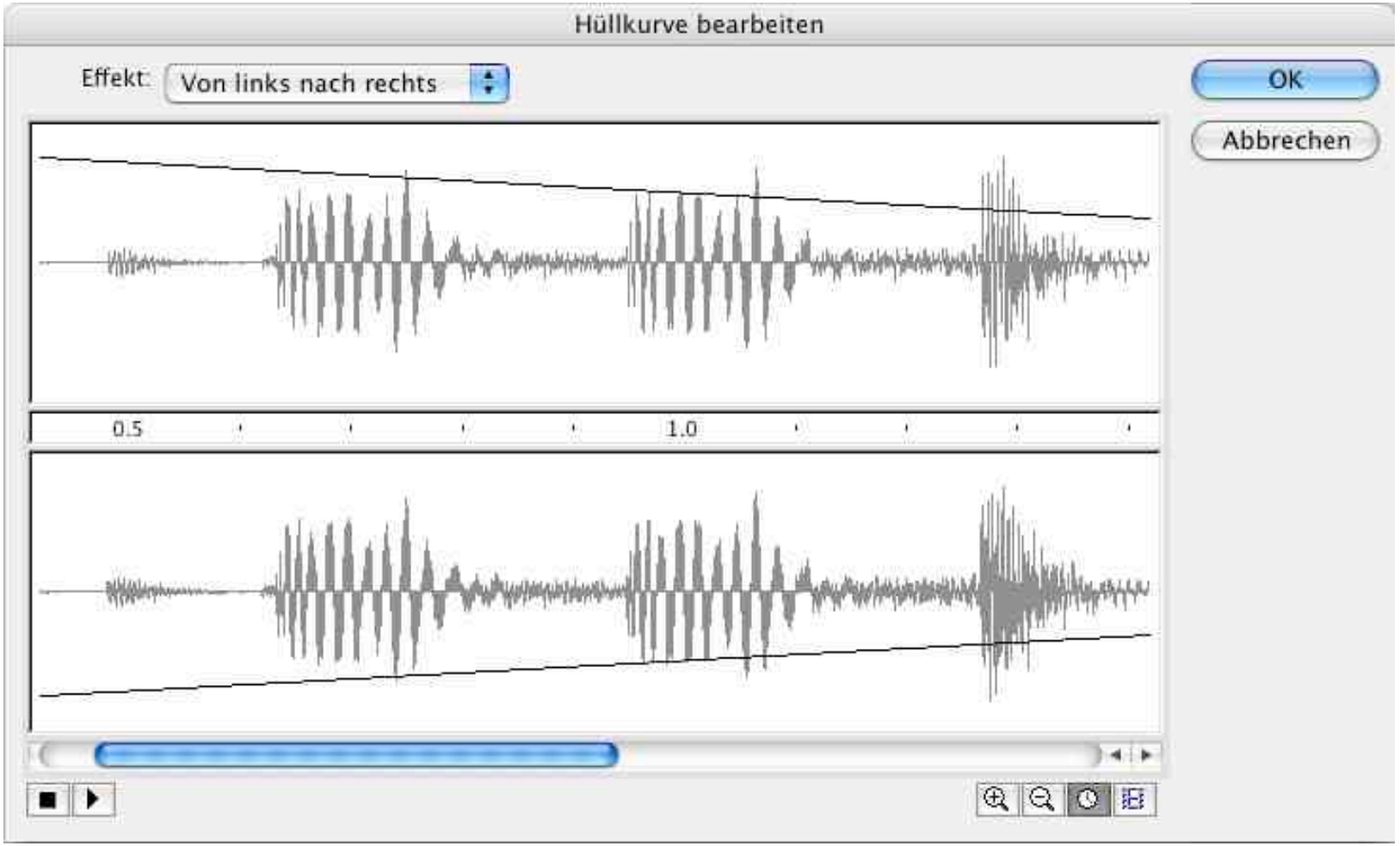
# Adobe Flash: Sounds in Library

- 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 in ActionScript 3.0

- First step:
  - Declare and create a sound object
  - Type referring to a sound in the library

```
private var _chimes:Chimes;
_chimes = new Chimes();
```

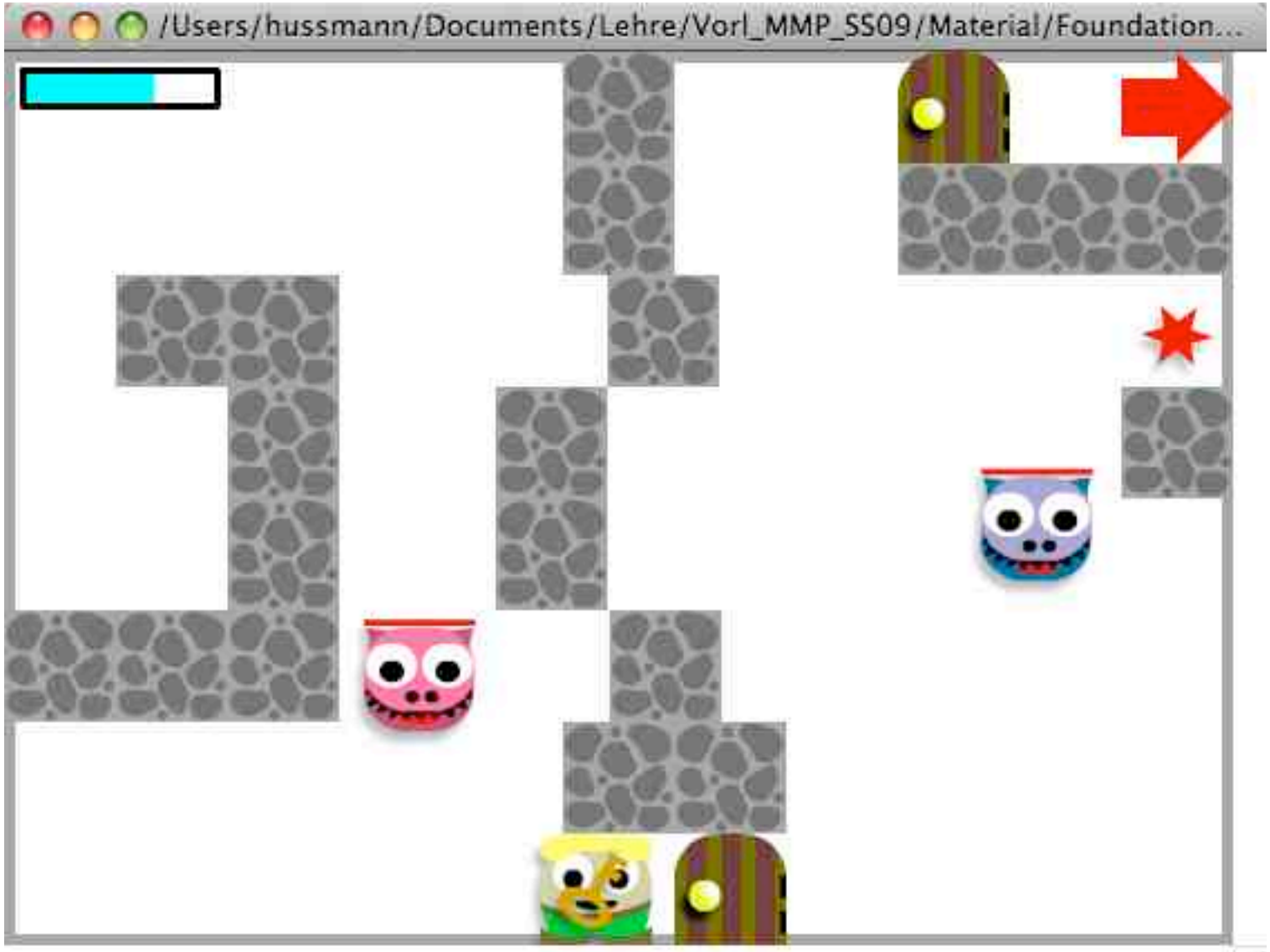
- Second step:
  - Declare and create a sound channel object
  - Independent of any specific sounds

```
private var _soundChannel:SoundChannel;
_soundChannel = new SoundChannel();
```

- Third step:
  - Play sound on a specific channel
  - Assign a sound object to a sound channel object

```
_soundChannel = _chimes.play();
```

# Example: Maze Game



# Example: Door Class for Maze Game (1)

```
package
{
  import flash.display.MovieClip;
  import flash.media.Sound;
  import flash.media.SoundChannel;
  import flash.events.Event;

  public class Door extends MovieClip
  {
     private var _isOpen:Boolean;
     private var _chimes:Chimes;
     private var _soundChannel:SoundChannel

     public function Door()
     {
          addEventListener
            (Event.ADDED_TO_STAGE, onAddedToStage);
     }
...
```

# Example: Door Class for Maze Game (2)

```
...
      private function onAddedToStage(event:Event):void
      {
            _isOpen = false;
            _chimes = new Chimes();
            _soundChannel = new SoundChannel();
            visible = true;
            addEventListener
             (Event.REMOVED_FROM_STAGE, onRemovedFromStage);
      }

      private function onRemovedFromStage(event:Event):void
      {
            removeEventListener
             (Event.ADDED_TO_STAGE, onAddedToStage);
            removeEventListener
             (Event.REMOVED_FROM_STAGE, onRemovedFromStage);
            trace("door removed");
      }
...
```

# Example: Door Class for Maze Game (3)

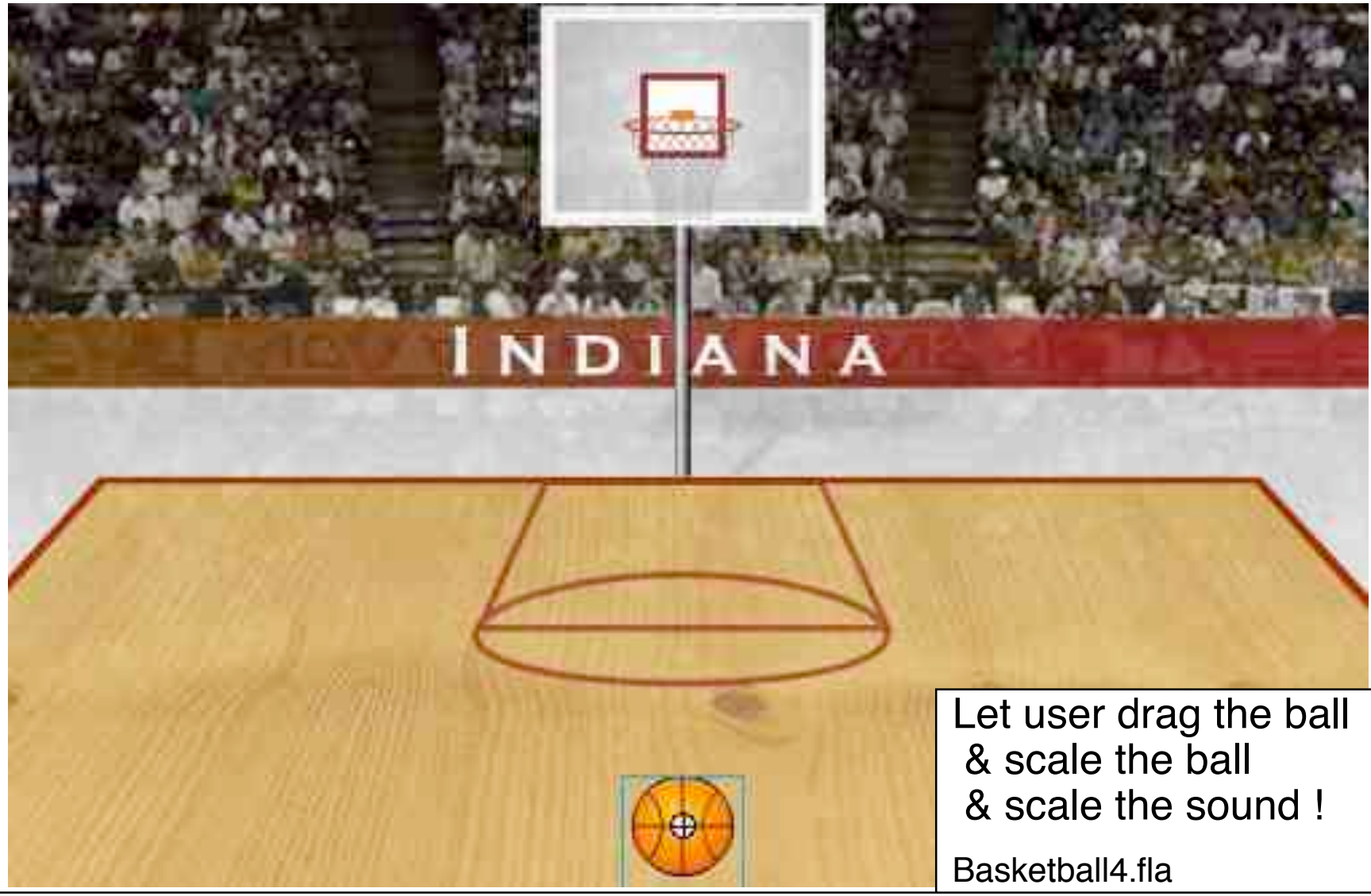
```
        //Getters and setters
        public function get isOpen():Boolean
        {
              return _isOpen;
        }

        public function set isOpen(doorState:Boolean)
        {
              _isOpen = doorState;
             if(_isOpen)
             {
                   _soundChannel = _chimes.play();
                  visible = false;
             }
             else
             {
                   visible = true;
             }
        }
   }
}
```

# 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 well only with short sounds


Basketball1.fla

# Dragging the Ball over the Court




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

Basketball4.fla

# Dynamic Scale & Volume & Stereo Panorama

(Note: Old ActionScript 2.0 example...)

In method for dragging the ball:

```
bounce.setVolume(topToBottomPercent);
...
var panAmount =
 ((_xmouse - centerPoint) / quadrantSize) * 100;
bounce.setPan(panAmount);
```

# 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

- In case of score:
  - Play “net sound”
  - Show basketball score animation (**`score_mc`**)
  - Update score fields of respective team (***`team`*`_txt`**)