Multimedia-Programmierung
Übung 6

Ludwig-Maximilians-Universität München

Sommersemester 2009

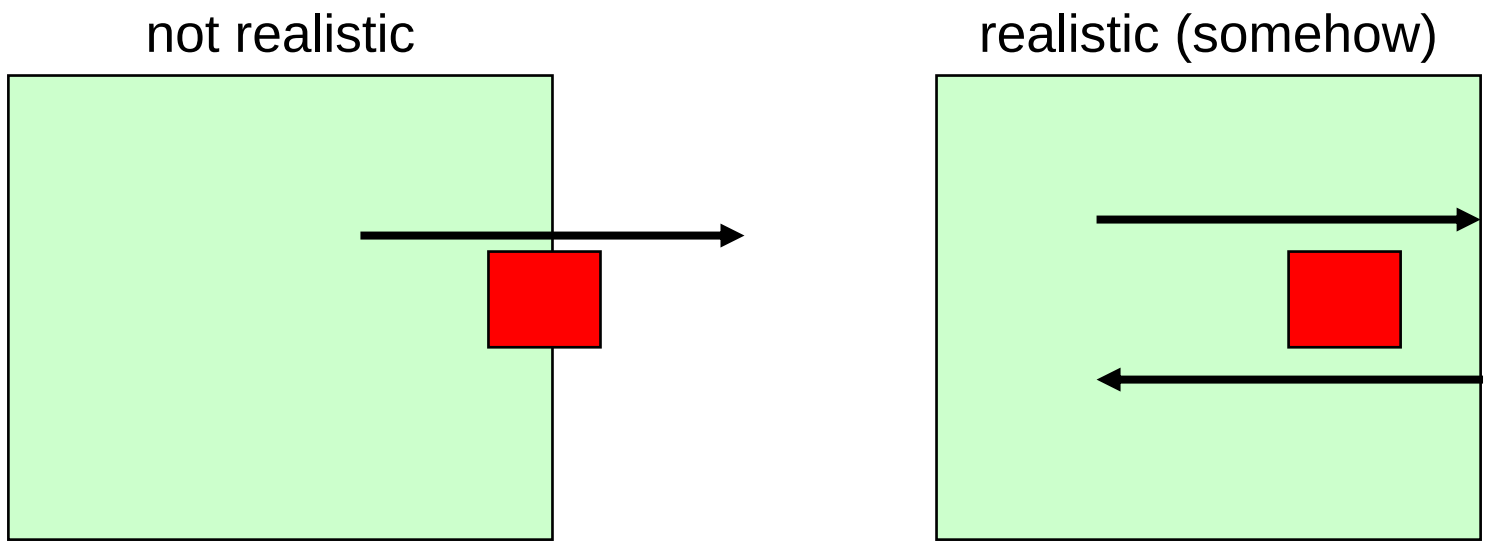
Today

- Physics and how they improve usability
- Sounds in 

Physics

How logical behaviour improves usability

- Users have specific expectations
- For example, if something hits a wall it should bounce or create some damage
- Adding physics to applications helps to improve usability



Physics

Examples I - Bumptop

- A physically enhanced Windows desktop

©bumptop.com

Physics

Examples II - Physics and Microsoft Surface

- Allows physically correct interaction with a tabletop device

Wilson, A. D., Izadi, S., Hilliges, O., Garcia-Mendoza, A., and Kirk, D. 2008. Bringing physics to the surface. In Proceedings of the 21st Annual ACM Symposium on User interface Software and Technology (Monterey, CA, USA, October 19 - 22, 2008). UIST '08. ACM, New York, NY, 67-76.

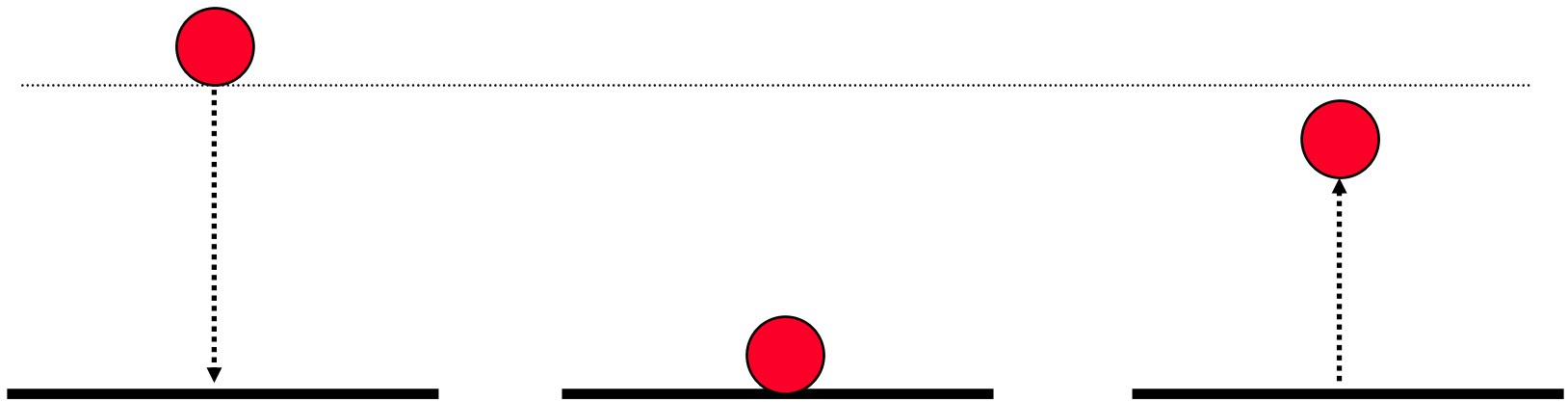
Programming Physics

- Frameworks, APIs, development tools etc. often offer physics engines (e.g. 3D game engines, Interpolators in Flash)

- In Python, **WE** do the physics!!

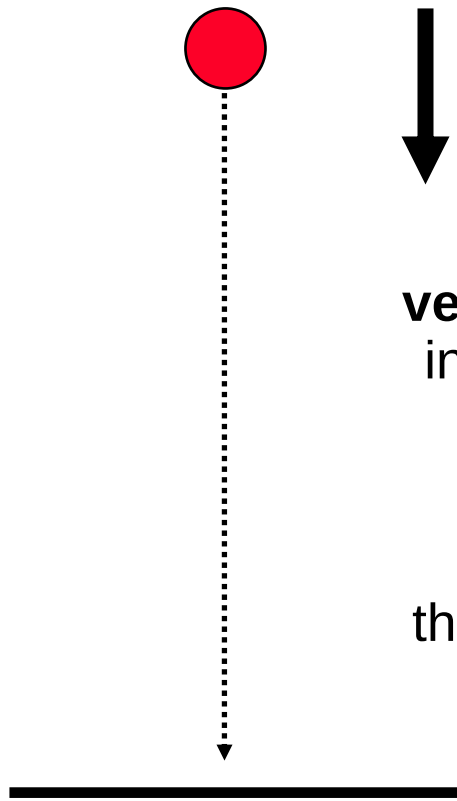
Bouncing Ball Example 1

- Let’s make a ball bounce in a realistic way
- 1. We need a concept:



Bouncing Ball Example 2

- 2. What makes the ball fall and bounce?



gravity makes the ball fall

velocity depends on gravity and increases/decreases over time

the material of the ball influences how far it will **bounce** back

Bouncing Ball Example 3

```
class Ball(pygame.sprite.Sprite):
    def __init__(self, color, initial_position):
        pygame.sprite.Sprite.__init__(self)
        size = 20
        self.gravity = 900
        self.velocity = 0
        self.bounce = 0.9

        self.image = pygame.Surface((size,size),pygame.SRCALPHA,32)
        pygame.draw.circle(self.image,color,(size/2,size/2),size/2)
        self.rect = self.image.get_rect()
        self.rect.center = initial_position

    def update(self, time_passed, size):

        self.velocity += (self.gravity * time_passed)
        self.rect.bottom += int(self.velocity * time_passed)

        if self.rect.bottom >= size[1]:
            self.rect.bottom = size[1]
            self.velocity = -self.velocity * self.bounce
```

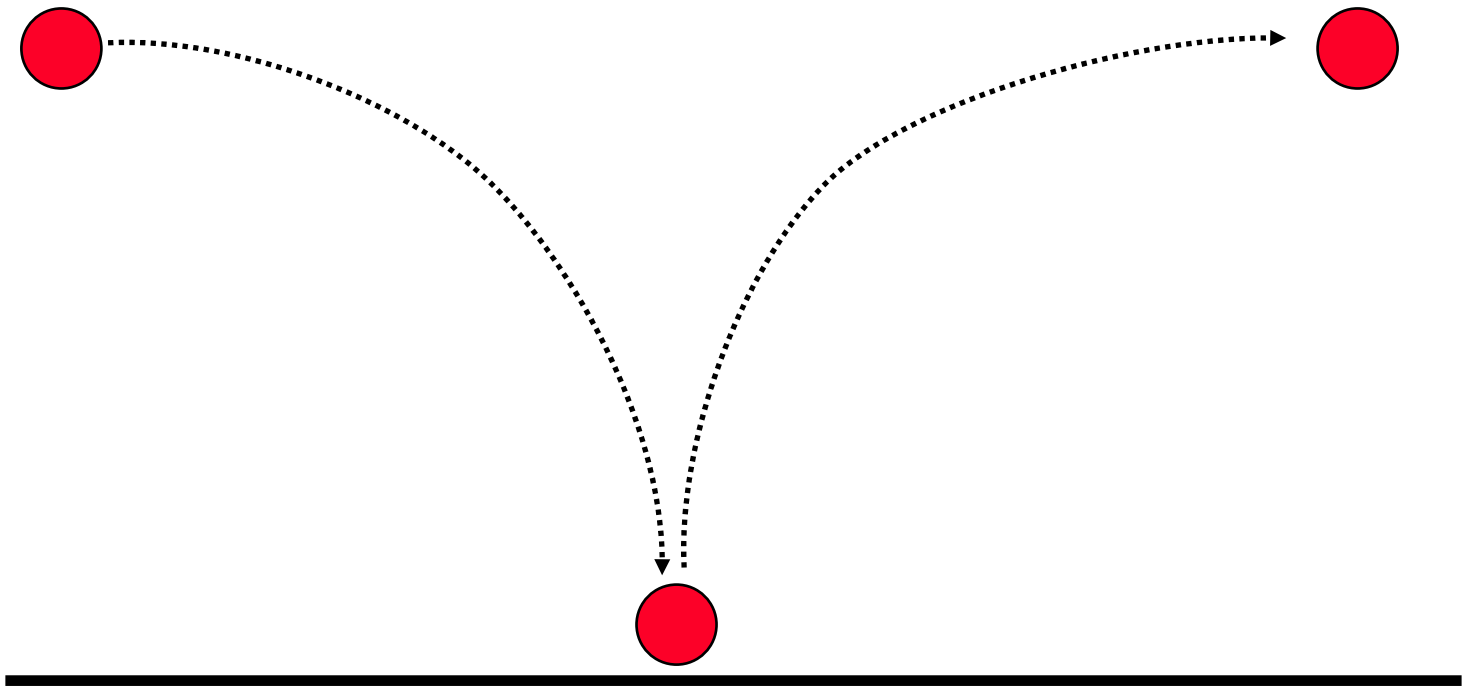
gravity per second, current velocity and bounce factor of the material

velocity is increased/decreased by the gravity

if the ball hits the ground, reduce velocity based on the bounce factor

Bouncing Ball Example 4

- Making the ball bounce and move vertically



Bouncing Ball Example 5

```
class Ball(pygame.sprite.Sprite):
    def __init__(self, color, initial_position):
        pygame.sprite.Sprite.__init__(self)
        size = 20
        self.gravity = 900
        self.vx = 0
        self.vy = 0
        self.bounce = 0.9
        …
    def update(self, time_passed, size):
        self.velocity += (self.gravity * time_passed)
        ydistance = int(self.vy * time_passed)
        self.rect.bottom += ydistance
        if ydistance == 0 and self.rect.bottom == size[1]: self.vx = 0
        self.rect.left += int(self.vx * time_passed)
        if self.rect.right >= size[0]:
            self.rect.right = size[0]
            self.vx = -self.vx
        if self.rect.left <= 0:
            self.rect.left = 0
            self.vx = -self.vx
        if self.rect.bottom >= size[1]:
            self.rect.bottom = size[1]
            self.vy = -self.vy* self.bounce
```

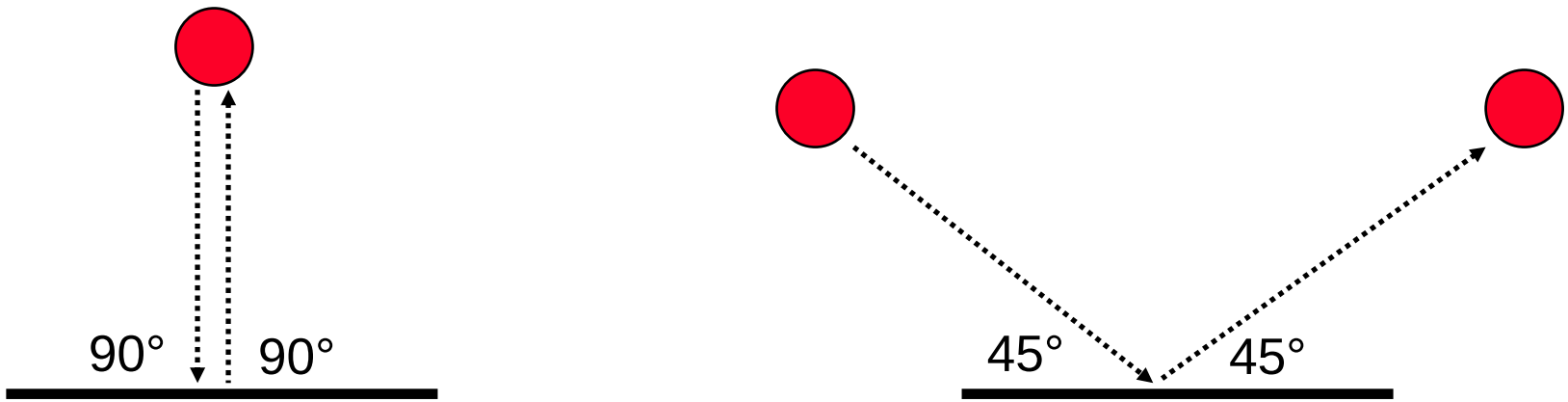
x and y velocity

clumsy way to make the ball stop

if the ball hits the sidewalls, make it change the direction

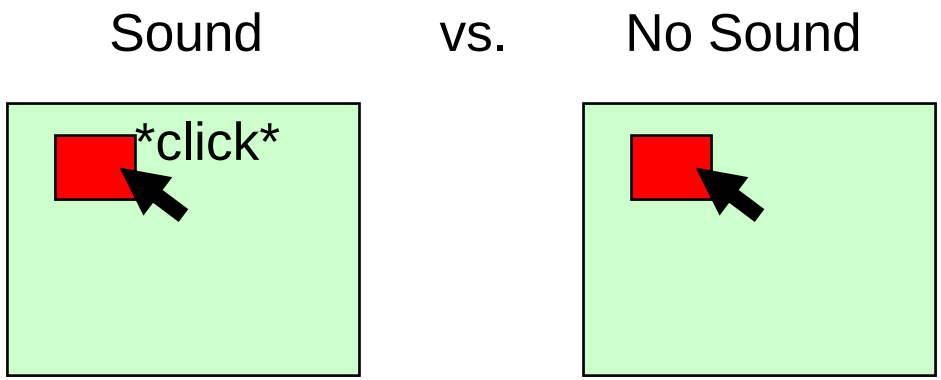
Arrival Angle = Angle of Reflection

- What if the Ball doesn't drop perfectly vertically?



Sound

- Sound is an essential part of multimedia applications
- Provides immediate feedback about an action
- Supports realism (e.g. games)
- Provides accessibility (e.g. for blind people)
- ...



Sound in Pygame

Mixer

- Sounds are controlled using the pygame.mixer interface
- Mixer must be initialized pygame.mixer.init(frequency,size,channels,buffer)
- Automatically initialized with pygame.init() using the default values
- Default values can be changed using pygame.mixer.pre_init()
- The mixer “mixes” the sounds in background threads
  - Sounds are not blocking the rest of the application logic

Sound in Pygame

Sound Object

- pygame.mixer.Sound provides a class to load and control sound files (OGG and uncompressed WAV)
- Sound.play(loops=0, maxtime=0, fade_ms=0) plays the sound file
- Other methods: stop(), fadeout(time), set_volume(value) etc.

playing a sound file

```
click_sound = pygame.mixer.Sound("click.wav")
click_sound.play()
```

playing a sound file in a loop 4(!) times

```
click_sound = pygame.mixer.Sound("click.wav")
click_sound.play(3)
```

Sound in Pygame

Channels

- A channel represents one of the channels that are mixed by the soundcard
- Sound.play() returns a Channel object (or None if all channels are blocked)
- Provides methods to manipulate teh sound and create useful effects (e.g. Channel.set_volume(left, right))

playing a sound file from the right speaker only

```
channel = click_sound.play()
channel.set_volume(0.0,1.0)
```

Sound in Pygame

Stereo Panning

- Create the illusion that sound is coming from a specific point at the screen
- Manipulate the volume of the different speakers
- Can be used to make a sound “move” over the screen

stereo panning function

```
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)
```

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

Music in Pygame

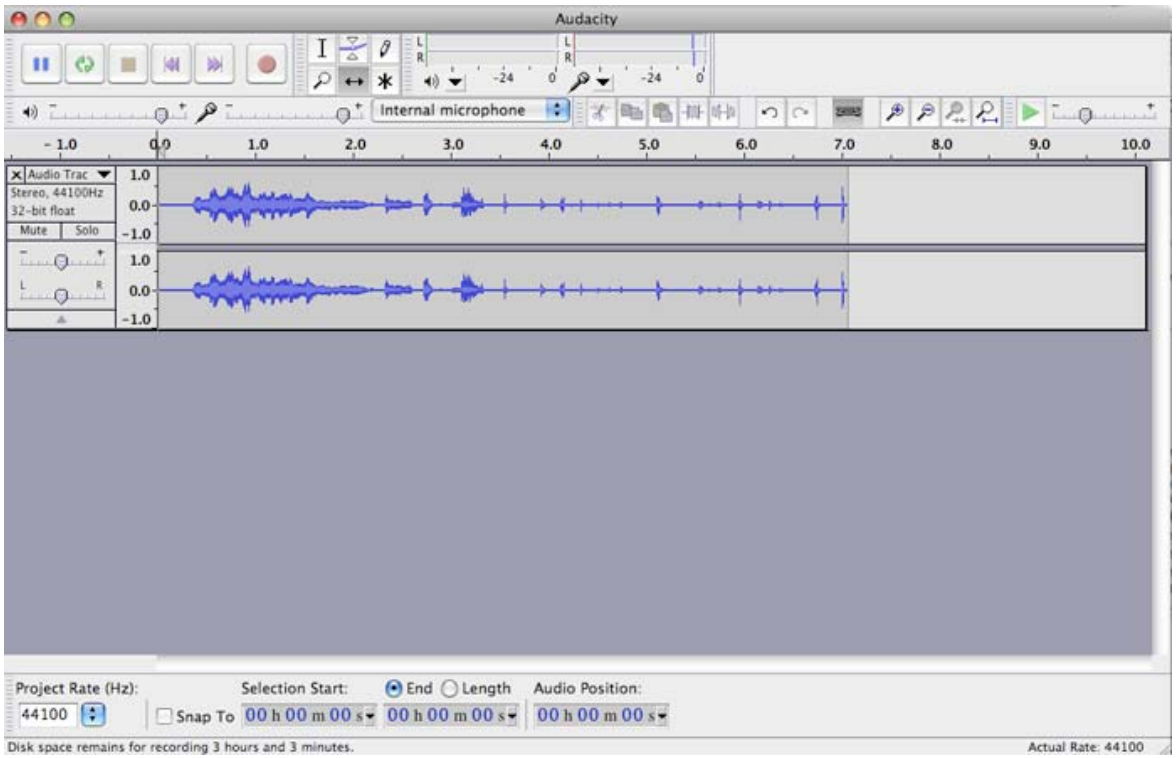
- Don’t use pygame.mixer but pygame.mixer.music
- It enables **streaming** music which means that the file will be read in small chunks
- Supports MP3 and OGG files (OGG better supported across platforms)
- Other Methods include stop(), pause(), rewind() etc.
- Attention: only one song can be streamed at the same time

playing a song using pygame

```
pygame.mixer.music.load("music.ogg")
pygame.mixer.music.play()
```

Creating your own Sound

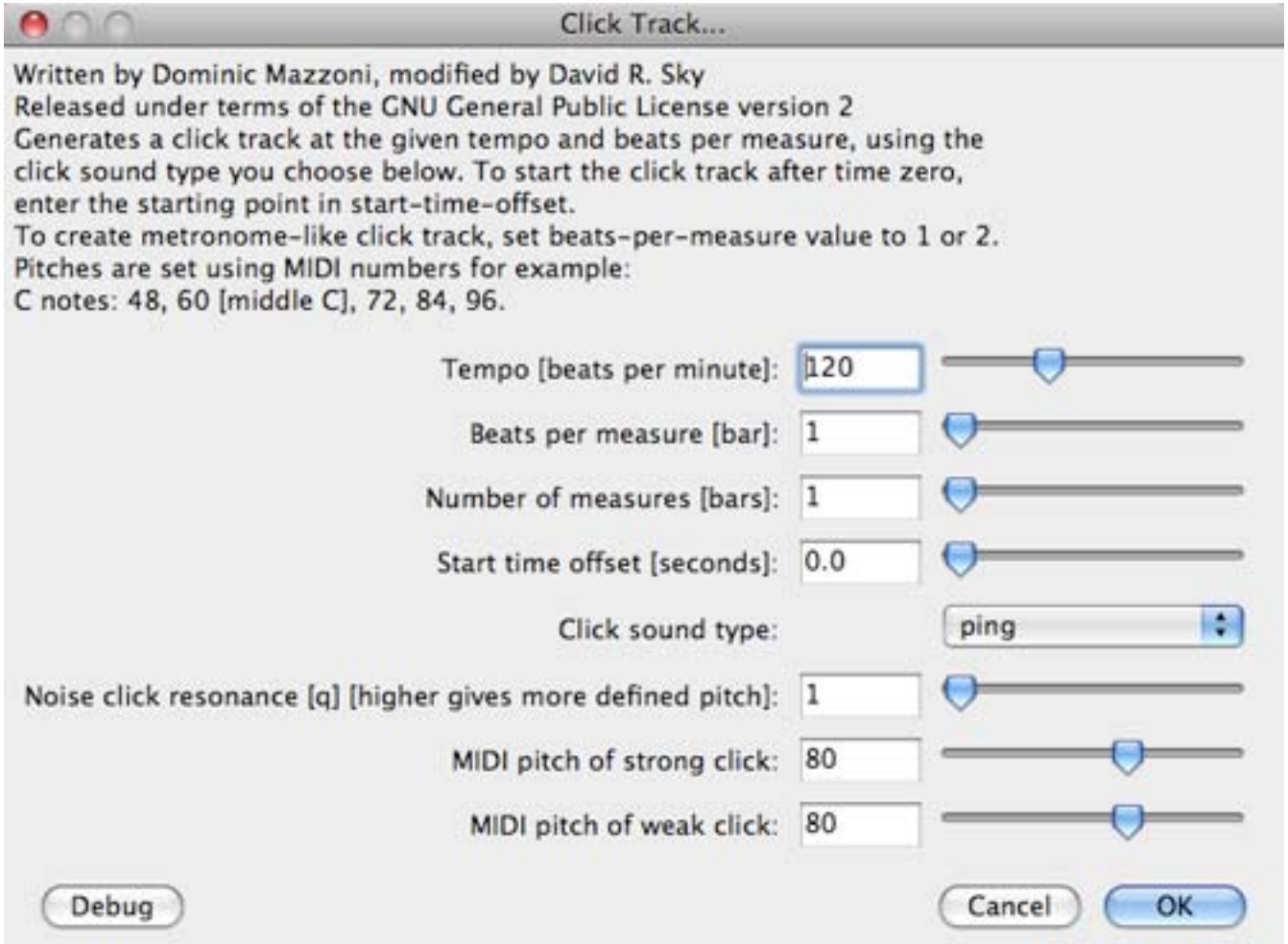
- Record real sounds and edit them
- Free sound editor Audacity (http://audacity.sourceforge.net/?lang=de)



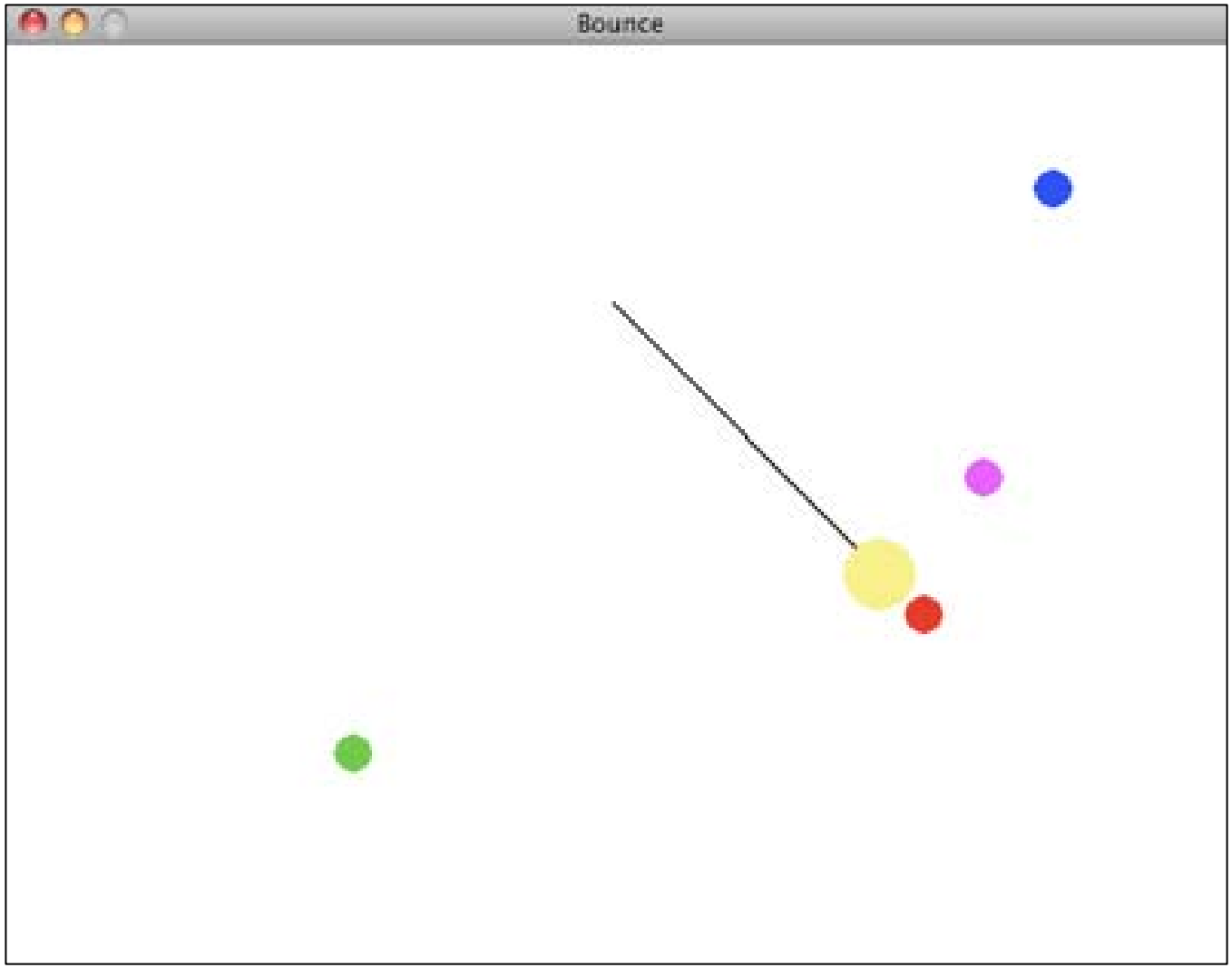
Creating your own Sound

generate a click

- Menu: Generate > Click Track



Homework



Useful Links

- Audacity
  http://audacity.sourceforge.net/?lang=de
- Pygame Mixer API
  http://www.pygame.org/docs/ref/mixer.html
- Pygame Music API
  http://www.pygame.org/docs/ref/music.html
- Pygame API !!!!
  http://www.pygame.org/ctypes/pygame-api/