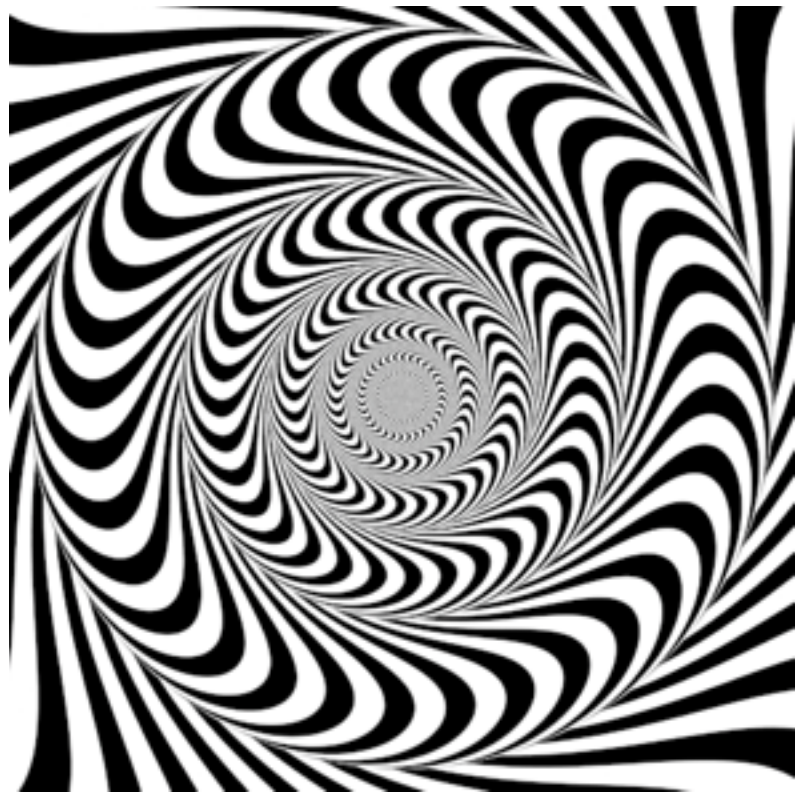
15-112
Fundamentals of Programming

Week 6 - Lecture 2:
More 2d lists. Animations.



February 18, 2016

Announcements

Connect4

bestQuiz(a)

Input:

```
                q0   q1  q2
a = [ [ 88,  80, 91 ],     student0
      [ 68, 100, -1 ],     student1
      [ 90,   78, 50]      student2
    ]
```

Output: the index of the quiz with the best average.

Interactive Graphics

Interactive graphics and animation

Animation:

a sequence of static images that change over time

Interactive graphics and animation

For now, in 15-112:

a sequence of static images that change with:

- time (a timer firing)
- mouse click
- keyboard press

] Events

Interactive graphics and animation

For now, in 15-112:

a sequence of static images that change with:

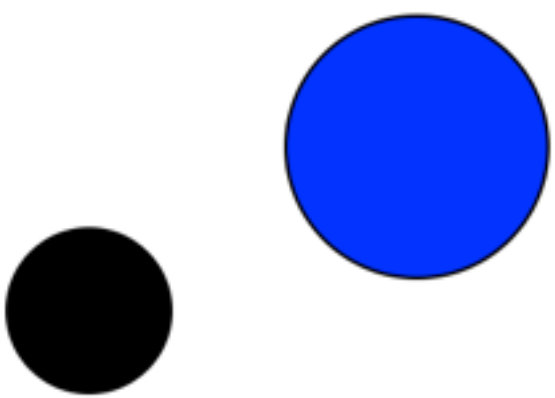
- time (a timer firing)
- mouse click
- keyboard press

Events

Interactive graphics and animation

An image is really a collection of data

(cx1, cy1) = (50, 50)
r1 = 25
color1 = “yellow”
(cx2, cy2) = (150, 50)
r2 = 25
color2 = “red”



(cx1, cy1) = (50, 100)
r1 = 25
color1 = “black”
(cx2, cy2) = (150, 50)
r2 = 40
color2 = “blue”

Interactive graphics and animation

For now, in 15-112:

a sequence of static images that change with:

- time (a timer firing)
- mouse click
- keyboard press

Events

Interactive graphics and animation

In order to react to events:

Repeat:

- Wait for an event.
- Once an event happens:
  - Call a function based on the event type:
    - Set values related to what you will draw (based on the information in the event)
  - Redraw everything (erase all and draw again)

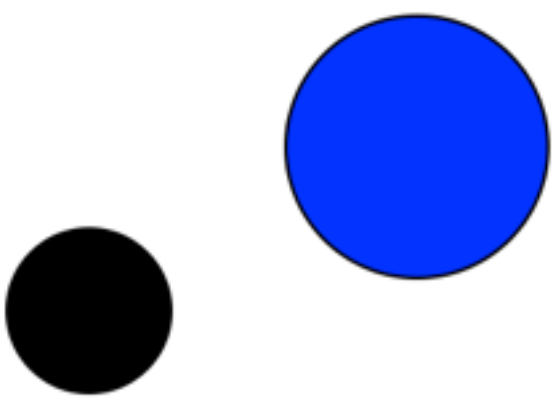
Interactive graphics and animation

Will store all values/data inside an object called data.

Interactive graphics and animation

An image is really a collection of data

(**data**.cx1, **data**.cy1) = (50, 50)
data.r1 = 25
data.color1 = “yellow”
(**data**.cx2, **data**.cy2) = (150, 50)
data.r2 = 25
data.color2 = “red”



(**data**.cx1, **data**.cy1) = (50, 100)
data.r1 = 25
data.color1 = “black”
(**data**.cx2, **data**.cy2) = (150, 50)
data.r2 = 40
data.color2 = “blue”

Interactive graphics and animation

Assuming we are only interested in mouse click events

```
def init(data):
    # Initialize the variables related to what you want to draw
    # e.g. (data.cx1, data.cx2) = (50, 50)

def mousePressed(event, data):
    # Change data according to the event
    # (event.x, event.y) contain the coordinates of the mouse click

def redrawAll(canvas, data):
    # draw in canvas using the values in data

def run(width=300, height=300)
   # Some crazy code
   # Listens for events. Calls appropriate event-handler function.

run(400, 200)
```

event handler function

Interactive graphics and animation

```
def init(data):
    # Initialize the variables related to what you want to draw
    # e.g. (data.cx1, data.cx2) = (50, 50)

def mousePressed(event, data):
    # Change data according to the event
    # (event.x, event.y) contain the coordinates of the mouse click

def keyPressed(event, data):
    # Change data according to the event
    # event.keysym contains the key symbol

def timerFired(event, data):
    # Can change data every time timer is fired

def redrawAll(canvas, data):
    # draw in canvas using the values in data

def run(width=300, height=300)
   # Some crazy code
   # Listens for events. Calls appropriate event-handler function.
```

Interactive graphics and animation

```
def init(data):
    # Initialize the variables related to what you want to draw
    # e.g. (data.cx1, data.cx2) = (50, 50)

def mousePressed(event, data):
    # Change data according to the event
    # (event.x, event.y) contain the coordinates of the mouse click

def keyPressed(event, data):
    # Change data according to the event
    # event.keysym contains the key symbol

def timerFired(event, data):
    # Can change data every time timer is fired

def redrawAll(canvas, data):
    # draw in canvas using the values in data

def run(width=300, height=300)
   # Some crazy code
   # Listens for events. Calls appropriate event-handler function.
```

event handlers

Model View Controller (MVC)

Model

- all the data you need to draw
  - data

View

- stuff you see on screen
  - redrawAll(canvas, data) and its helper functions

Controllers

- change data according to events
  - event handlers and their helper functions