12장 그래픽스 응용

이 장의 내용

- **12.1** 그래픽스 개요 (활용)
- **12.2 tkinter** 그래픽 모듈
- **12.3 tkinter** 그래픽 활용
- **12.4** 프랙탈 그래픽

내부라이브러리
TCL/TK라는 언어
tool command Language
/ tool kit

fractal

12.1 그래픽스 개요

그래픽스 개요

- 컴퓨터 그래픽스
  - 컴퓨터 소프트웨어와 하드웨어를 사용하여 가상의 이미지를 생성 처리 하는 과정
  - 래스터 그래픽스 raster graphics와 정점vertex 기반의 벡터 그래픽스vector graphics 시스템으로 구분
  - 물체의 형태를 설계하는 과정을 모델링(modeling) 과정과, 모델링된 물체 표면을 색으 로 칠하거나, 텍스춰 매핑(texture mapping), 광원 효과(lighting)를 계산하여 실감나는 물체를 표현하는 작업을 렌더링(rendering) 과정으로 구분

그림 12.1 원본 이미지(왼쪽)의 확대 결과 (래스터 폰트–중앙, 벡터 폰트–오른쪽)

12.2 tkinter 그래픽 모듈

- tkinter 모듈
  - 고차원의 그래픽 응용 개발을 위해서 파이썬에 내장된 그래픽 모듈
  - Tk 클래스의 객체인 tk 생성
  - tkinter 그래픽 모듈의 함수를 통한 도형은 캔버스(canvas) 공간에 그려진다
- Canvas 클래스
  - canvas 객체가 생성하여 이 객체를 통하여 모든 그래픽 관련 함수가 호출된다

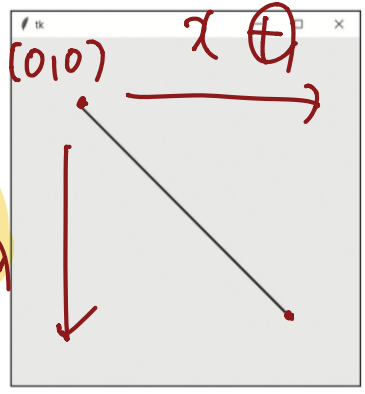
선그리기

- canvas 객체의 create_line 함수 ↗그냥두점이래는뜻
  - 4개의 인자는 좌상단(x1, y1)과 우하단(x2, y2)의 좌표

```
from tkinter import *
tk = Tk()
canvas = Canvas(tk, width = 500, height = 500)
canvas.pack()
canvas.create_line(100, 100, 400, 400)
```

from tkinter import * 모든것

cf) fill(색상) 으로색바꿀수있음



사각형 그리기

- create_rectangle 함수
  - 사각형이 그려질 영역의 좌상단(x1, y1), 우하단 (x2, y2) 좌표

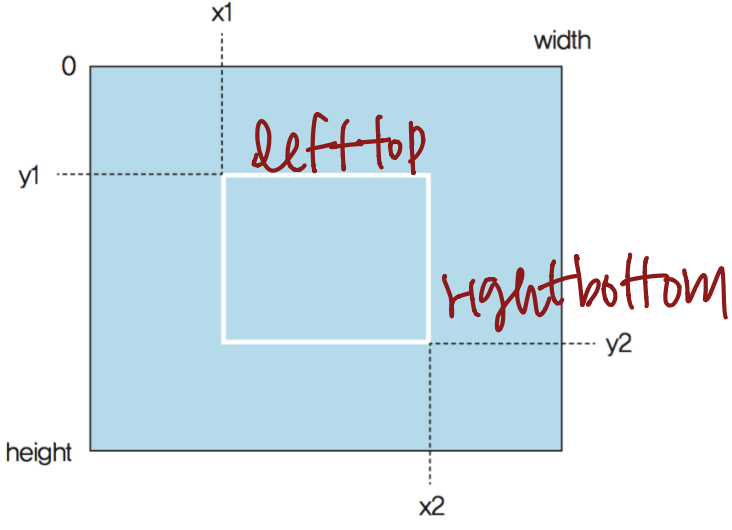


그림 12.2 사각형의 영역 지정하기

사각형 그리기

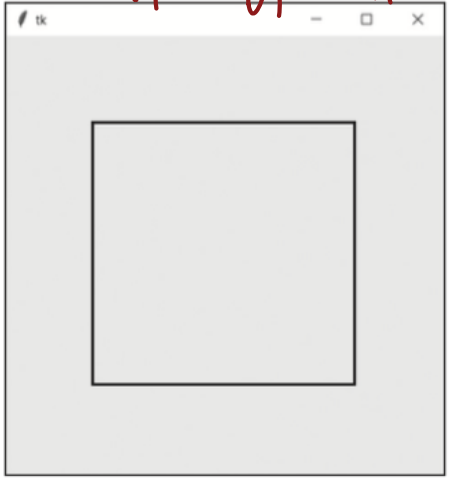
- 사각형 그리기

```
from tkinter import *
tk = Tk()
canvas = Canvas(tk, width = 500, height = 500)
canvas.pack()
canvas.create_rectangle(100, 100, 400, 400)
```

bg='yellow' : 배경색

fill='red'

x1 y1 x2 y2



무작위 직선 그리기

- x1 = randrange(400)
  - 0 ~ 399 사이의 난수를 발생시켜서 반환
- x2 = x1 + randrange(200) - 100 # x1 + (-100 ~ 99)

```
from tkinter import *
import random
tk = Tk()
canvas = Canvas(tk, width = 500, height = 500)
canvas.pack()
for i in range(200):
    x1 = random.randrange(400)
    y1 = random.randrange(400)
    x2 = x1 + random.randrange(200) - 100
    y2 = y1 + random.randrange(200) - 100
    canvas.create_line(x1, y1, x2, y2)
```

무작위 사각형 그리기

- create_rectangle 함수

```
from tkinter import *
import random
tk = Tk()
canvas = Canvas(tk, width = 500, height = 500)
canvas.pack()
for i in range(100):
    x1 = random.randrange(400)
    y1 = random.randrange(400)
    x2 = x1 + random.randrange(100)
    y2 = y1 + random.randrange(100)
    canvas.create_rectangle(x1, y1, x2, y2)
```

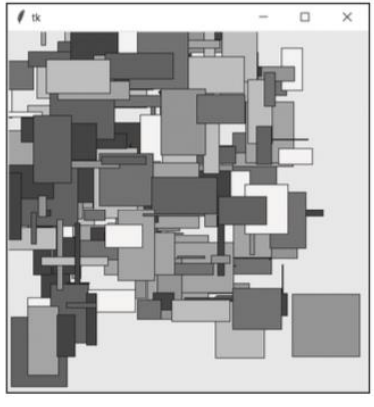
left top

rightbottom

무작위 사각형 칠하기

- fill = random.choice(colors)

```
from tkinter import *
import random
tk = Tk()
canvas = Canvas(tk, width = 500, height = 500)
canvas.pack()
colors =
['red','pink','blue','purple','violet','orange','yellow
', 'green']
for i in range(200):
    x1 = random.randrange(400)
    y1 = random.randrange(400)
    x2 = x1 + random.randrange(200) - 100
    y2 = y1 + random.randrange(200) - 100
    canvas.create_rectangle(x1, y1, x2, y2, fill =
    random.choice(colors))
```

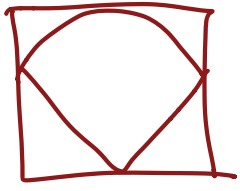


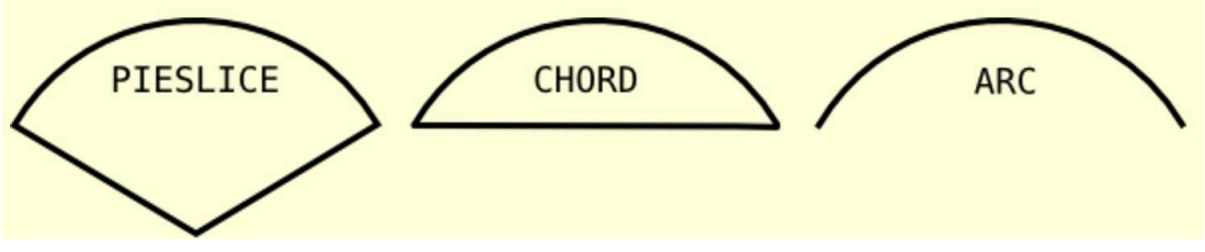
동심원 그리기

360을 넣으면 0으로 봐버리기때문에 최대한 가까운 값!

- canvas.create_arc(x1, y1, x2, y2, extent = 359, style = ARC)
  - extent는 호의 중심 각도 : 359도 = 원, cf. 360 == 0
  - style = PIESLICE, CHORD, ARC

사각형 안에 그려지는 개념



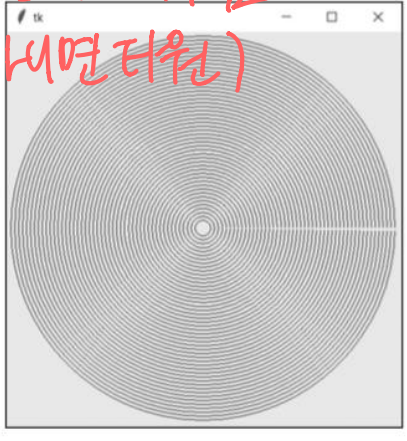


동심원 그리기

뒤에있는게 보이게하려면 도형을 겹쳐서 그려야됨.

```
from tkinter import *
import random
import time
tk = Tk()
canvas = Canvas(tk, width = 500, height = 500)
canvas.pack()
width = 500
height = 500
step = 5
for i in range(10, 250, step):
    x1 = width / 2 - i
    y1 = height / 2 - i
    x2 = width / 2 + i
    y2 = height / 2 + i
    canvas.create_arc(x1, y1, x2, y2, extent=359,style=ARC)
    tk.update()
    time.sleep(0.05)
```

create_oval 이용하면 원. (단, 정사각형 안에 그려야됨 아니면 타원)



크기조정

250-250 = 0

250+250 = 500 → 최대 크기

기다리도록

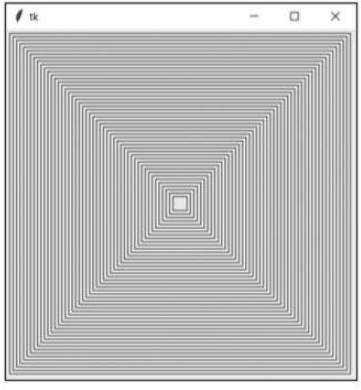
이게없으면 한번에 그려버림. "update" 되지않음

반복 사각형

```
from tkinter import *
import random
import time
tk = Tk()
canvas = Canvas(tk, width = 500, height = 500)
canvas.pack()
width = 500
height = 500
step = 5
for i in range(10, 250, step):
    x1 = width / 2 - i
    y1 = height / 2 - i
    x2 = width / 2 + i
    y2 = height / 2 + i
    canvas.create_rectangle(x1, y1, x2, y2)
    time.sleep(0.05)
```

left top

right bottom



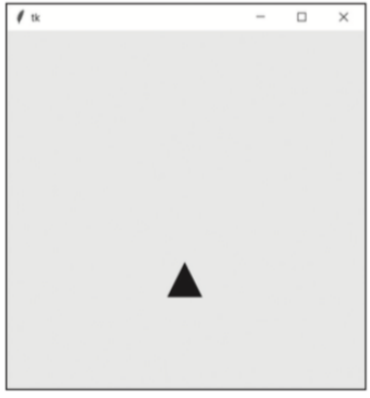
fill=random.choice(colors)

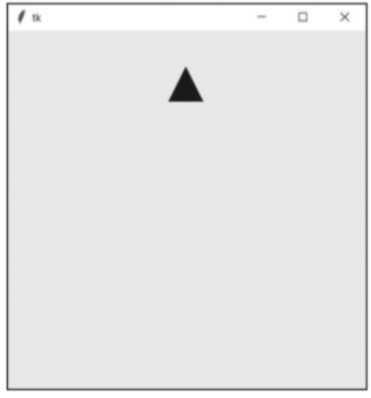
12.3 tkinter 그래픽 활용

- 애니메이션 animation
  - 정지된 그래픽 이미지를 일정 시간에 걸쳐 움직이게 만드는 일련의 작업
  - 이전 이미지에 대한 잔상 효과(afterimage effect)를 활용
- 움직이는 삼각형
  - 삼각형 도형이 아래쪽에서 위쪽으로 서서히 이동하는 애니메이션 프로그램
  - 70개의 이미지 프레임
  - sleep(0.05)는 0.05초 동안 멈추라는 의미

움직이는 삼각형

```
from tkinter import *
import random
import time
tk = Tk()
canvas = Canvas(tk, width = 500, height = 500)
canvas.pack()
canvas.create_polygon(250, 400, 275, 450, 225, 450)
for y in range(0, 70):
    canvas.move(1, 0, -5)
    tk.update()
    time.sleep(0.05)
```





점수 그래프 예제

옆으로 누운 막대그래프

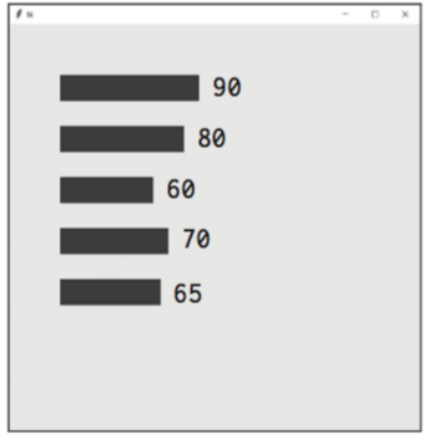
- 캔버스 화면에 텍스트를 표현할 때는 canvas.create_text를 사용

텍스트는 그림상에.

```
from tkinter import *
import time
tk = Tk()
canvas = Canvas(tk, width = 800, height = 800) canvas.pack()
scores = []
print("input 5 scores (0~100): ")
for k in range(5):
    score = int(input()) scores.append(score) i =0
for score in scores:
    x1 = 100
    y1 = 100 + i
    x2 = 100 + score * 3
    y2 = 100 + 50 + i
    canvas.create_rectangle(x1, y1, x2, y2, fill = "blue")
    canvas.create_text(x2 + 40, y1 + 20, text = str(score))
    i = i + 100
    tk.update()
    time.sleep(0.5)
```

i=0

폭(높이)가 50이다



이미지 불러오기

- PhotoImage 함수
  - 이미지 파일을 열어서 화면에 출력하는 기능
  - gif 이미지 형식만 지원
  - myimage = PhotoImage(file = img)
  - canvas.create_image(10, 10, anchor = NW, image = myimage)

NW	N	NE
W	CENTER	E
SW	S	SE

그림 12.3 Anchor 옵션

이미지 불러오기

```
from tkinter import *
import random
import time
tk = Tk()
canvas = Canvas(tk, width = 1000, height = 1000)
canvas.pack()
image_list = ['Koala.gif', 'Penguins.gif']
while True:
    for img in image_list:
    myimage = PhotoImage(file = img)
    canvas.create_image(10, 10, anchor = NW, image = myimage)
    tk.update()
    time.sleep(3)
```

NW 닻을 점

(10,10)

12.4 프랙탈 그래픽

- 프랙탈fractal
  - 작은 일부분이 전체와 비슷한 기하학적 형태를 말하며 이런 특징을 자기 유사성self-similarity 이라고 한다
- 프랙탈 그래픽 도형
  - 컴퓨터 소프트웨어를 이용하여 재귀적이거나 반복적인 작업을 통 해 만들어지는 패턴의 도형
  - 만델브로 집합Mandelbrot set, 칸토어 집합Cantor set, 시어핀스키 삼각형Sierpinski triangle, 코흐 눈송이koch snowflake 등

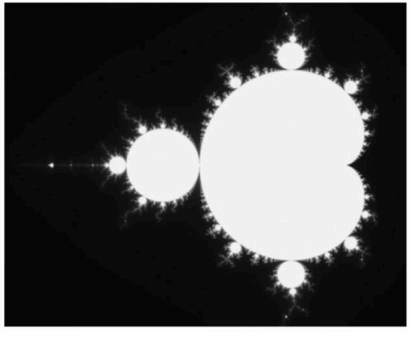


그림 12.4 멘델브로 프랙탈

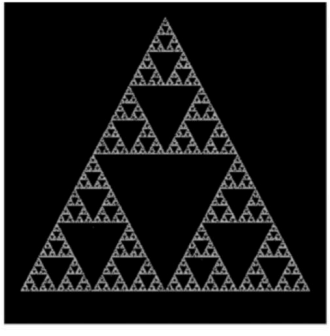


그림 12.5 프랙탈 삼각형

프랙탈 삼각형

- 프랙탈 삼각형fractal triangle
  - '불규칙한 상황을 무한히 반복하면 규칙적인 결과에 도달한다'는 카오 스 이론c haos theory의 한 예
  - 세개의 점 A, B, C와 기준점 0이 주어진다.
  - 그 다음 난수를 발생시켜 세 점 A, B, C 중 하나의 점을 선택한다.
  - 기준점 0과 선택된 점 A 사이의 중점을 칠한다.
  - 이 중점과 또다른 난수로 선택한 A, B, C 중 한 점을 선택하고 중점을 칠하고 이 과정을 반복한다
  - dice = random.randint(1, 3)

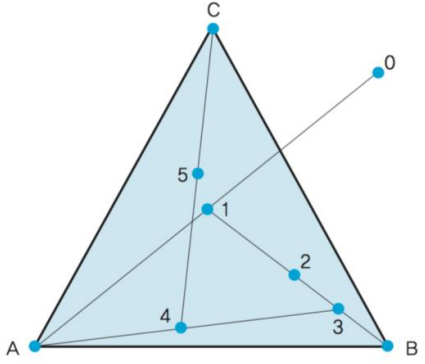
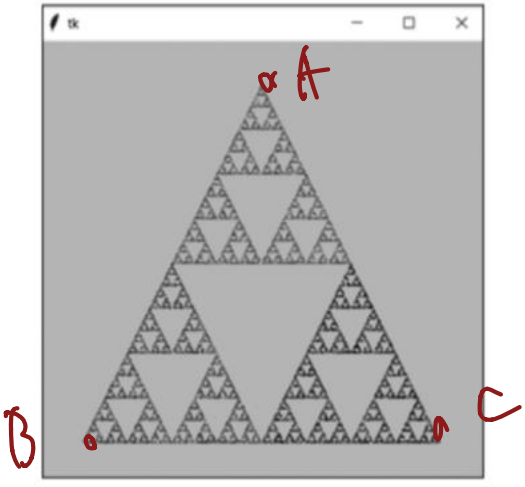


그림 12.6 프랙탈 삼각형 알고리즘

프랙탈 삼각형

원래 만들어진 걸 활용하는 것




```
from tkinter import *
import random
import time
tk = Tk()
canvas = Canvas(tk, width = 500, height = 500)
canvas.configure(background = 'light gray')
canvas.pack()
x = 250
y = 250
for i in range(50000):
    dice = random.randint(1, 3)
    if dice == 1:
        px = 250
        py = 50
        mycolor = 'red'
    elif dice == 2:
        px = 50
        py = 450
        mycolor = 'green'
    else:
        px = 450
        py = 450
        mycolor = 'blue'
    x = (x + px) / 2
    y = (y + py) / 2
    canvas.create_line(x,y,x+1, y+1,fill=mycolor)
    tk.update()
```

help-turtle demo

Key Point

Key Point

- 1과 x사이의 임의의 정수를 발생시키고 싶을 때 random 모듈의 randint(1, x) 메소드를 호출한다.
- 프랙탈 그래픽은 “불규칙한 상황을 무한히 반복하면 규칙적인 결과에 도달한다”는 카오스 이론(chaos theory)의 한 예이다.

프로그래밍 실습

▶ 프로그래밍 실습 1

1. 원이 점점 커지다가 다시 작아지는 프로그램을 작성하시오.
2. 프로그램 **12.9**의 색상 그래프를 점수 구간에 따라 다른색상으로 그리는 프로그램으로 수정하시오.

(예, 90 이상 **:** 녹색**, 80** 이상 **:** 파란색**, 60** 이상 **:** 주황색**, 60** 점 이하 **:** 빨간색**)**

▶ 프로그래밍 실습 2

1. 프랙탈 삼각형 예제에서 픽셀의 수를 **10,000**개와 **50,000**개로 각각 변경하여 그려보자**.** 삼각형이 그리지는 속도를 변경해 보자**.**