Python Turtle Graphics

Turtle Graphics ใน Python คืออะไร?

Turtle Graphics คือโมดูล Python ที่ให้วาดและสร้างแอนิเมชันได้โดยการควบคุม "เต่า" เสมือนจริงบนหน้าจอ โมดูลนี้มอบวิธีที่ใช้งานง่ายและสนุกสนานในการโต้ตอบกับโค้ด ช่วยให้สามารถสั่งเต่าได้ เช่น "เดินไปข้างหน้า" "เลี้ยวซ้าย" หรือ "วาดวงกลม" และดูคำตอบได้แบบเรียลไทม์

"เต่า" ใน Turtle Graphics คือเคอร์เซอร์หรือปากกาที่เคลื่อนที่ไปตามหน้าจอตามคำสั่งของคุณ เมื่อมันเคลื่อนที่ไปข้างหน้ามันจะทิ้งร่องรอยไว้ สร้างเส้น รูปร่าง หรือภาพวาดที่ซับซ้อนยิ่งขึ้นตามคำสั่ง แนวคิดคือ การตอบกลับด้วยภาพช่วยให้เรียนรู้หลักการเขียนโปรแกรมพื้นฐานแต่สำคัญได้ง่ายขึ้น โดยเฉพาะอย่างยิ่ง สามารถเรียนรู้เกี่ยวกับลูป (เพื่อทำซ้ำรูปร่าง) ฟังก์ชัน (เพื่อสร้างโค้ดที่นำกลับมาใช้ใหม่ได้) และเงื่อนไข (เพื่อตัดสินใจว่าเต่าควรทำอะไรต่อไป) ตัวอย่างเช่น หากเราต้องการวาดรูปดาว เราสามารถใช้ลูปเพื่อทำซ้ำการเคลื่อนไหวไปข้างหน้าและหมุนของเต่าสี่ครั้ง โปรดทราบว่าเนื่องจาก turtle graphics เป็นส่วนหนึ่งของไลบรารี Python มาตรฐาน จึงไม่จำเป็นต้องติดตั้งเพิ่มเติม

คำสั่ง Python สำหรับกราฟิกเต่าทั่วไป

มาทำความคุ้นเคยกับคำสั่งกราฟิกเต่าพื้นฐานเพื่อช่วยคุณสร้างงานออกแบบชิ้นแรกกันเถอะ เราจะเริ่มต้นด้วยการควบคุมการเคลื่อนไหวและการวาดภาพเต่าที่ง่ายที่สุด

1. เต่านำเข้า

ก่อนวาดภาพ คุณต้องนำเข้าโมดูลกราฟิกเต่าโดยใช้คำสั่งนี้ ซึ่งจะช่วยให้คุณเข้าถึงฟังก์ชันทั้งหมดของเต่าได้

```
import turtle
```

2. เต่า.ไปข้างหน้า()

คำสั่งนี้จะเคลื่อนที่เต่าไปข้างหน้าตามจำนวนหน่วยที่กำหนด ในกรณีนี้คือ 100 หน่วย เต่าจะลากเส้นขณะที่มันเคลื่อนที่

```
turtle.forward(100)
```

3. เต่า.ขวา()

เต่าจะหันตัวไปทางขวา 90 องศาแต่ไม่เคลื่อนไหว ซึ่งมีประโยชน์ในการเปลี่ยนทิศทางก่อนที่จะเคลื่อนไหวครั้งต่อไป

```
turtle.right(90)
```

4. เต่า.วงกลม()

คำสั่งนี้จะวาดวงกลมที่มีรัศมี 50 หน่วย คุณสามารถปรับรัศมีเพื่อวาดวงกลมให้ใหญ่ขึ้นหรือเล็กลงได้

```
turtle.circle(50)
```

5. เต่า.penup()

การทำเช่นนี้จะยกปากกาขึ้น หมายความว่าเต่าจะเคลื่อนไหวโดยไม่วาดอะไรเลย การเปลี่ยนตำแหน่งเต่าโดยไม่ทิ้งร่องรอยไว้ก็เป็นประโยชน์

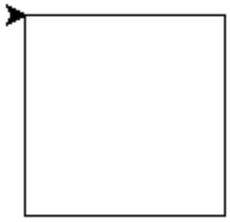
```
turtle.penup()
```

6. turtle.pendown()

การดำเนินการนี้จะลดปากกาลง ทำให้เต่าสามารถเริ่มวาดอีกครั้งหลังจากpenup()รับคำสั่ง

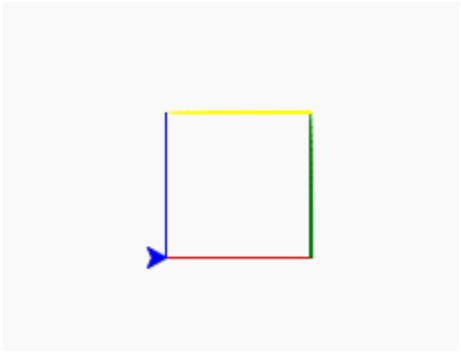
ตัวอย่างที่ 1 รูปทรงเรียบง่าย

```
import turtle
t = turtle.Turtle()
for _ in range(4):
    t.forward(100)
    t.right(90)
turtle.done()
```



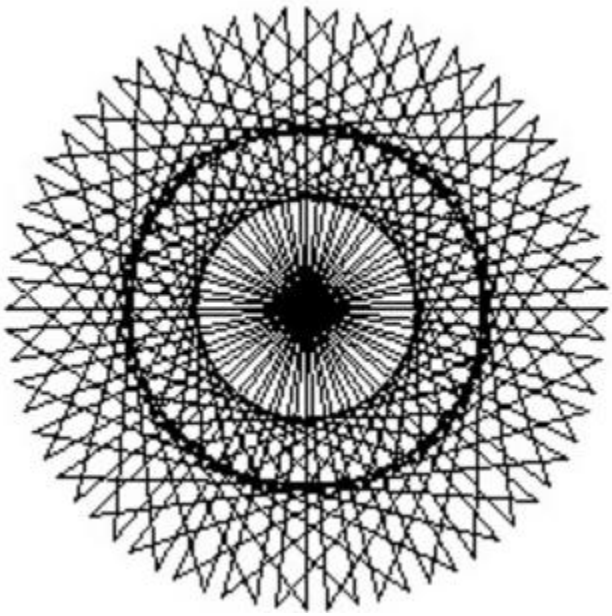
ตัวอย่างที่ 1.1 รูปทรงเรียบง่าย

```
import turtle
t = turtle.Turtle()                    # สร้างพื้นที่สำหรับเล่น turtles
for c in ['red', 'green', 'yellow', 'blue']:    # ลูปค่าตัวแปร c ตามสี
    t.color(c)                         # กำหนดสีตามค่าที่ลูปได้จากตัวแปร c
    t.forward(90)                      # ลากเส้นตรงไปข้างหน้า 90 พิกเซล
    t.left(90)                         # หันไปทางมุม 90 องศาจากมุมเดิม
turtle.mainloop()                      # ลูปค้างหน้าจอไว้
```



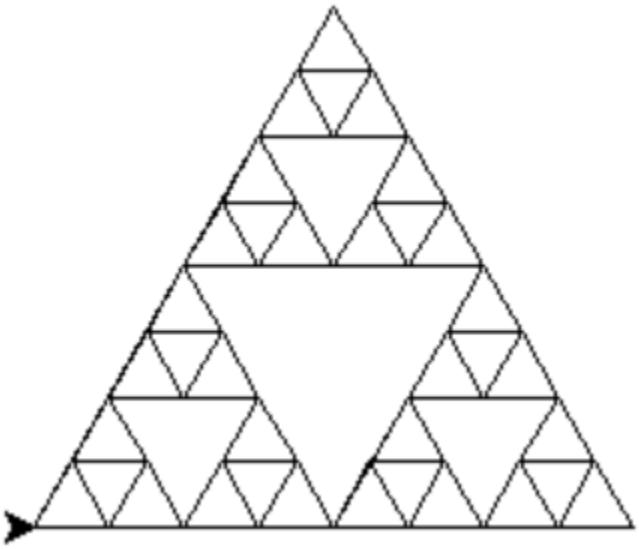
ตัวอย่างที่ 2 ลวดลายเรขาคณิต

```
import turtle
t = turtle.Turtle()
for _ in range(36):
    for _ in range(5):
        t.forward(100)
        t.right(144)
    t.right(10)
turtle.done()
```



ตัวอย่างที่ 3 แฟร็กทัลและการออกแบบแบบเรียกซ้ำ

```
 import turtle
def sierpinski(t, order, size):
    if order == 0:
        for _ in range(3):
            t.forward(size)
            t.left(120)
    else:
        sierpinski(t, order-1, size/2)
        t.forward(size/2)
        sierpinski(t, order-1, size/2)
        t.backward(size/2)
        t.left(60)
        t.forward(size/2)
        t.right(60)
        sierpinski(t, order-1, size/2)
        t.left(60)
        t.backward(size/2)
        t.right(60)
t = turtle.Turtle()
sierpinski(t, 3, 200)
turtle.done()
```



ตัวอย่างที่ 4 ภาพวาดแบบโต้ตอบ

การควบคุมการเคลื่อนไหวของเต่าได้โดยใช้คีย์บอร์ดหรือเมาส์ ซึ่งจะช่วยสร้างประสบการณ์ที่ไดนามิกยิ่งขึ้น สามารถสร้างโปรเจกต์ต่างๆ เช่น Etch-A-Sketch แบบดิจิทัล ซึ่งสามารถควบคุมเต่าด้วยปุ่มลูกศรเพื่อวาดบนหน้าจอ

```
import turtle
t = turtle.Turtle()
def move_up():
    t.setheading(90)
    t.forward(10)
def move_down():
    t.setheading(270)
    t.forward(10)
def move_left():
    t.setheading(180)
    t.forward(10)
def move_right():
    t.setheading(0)
    t.forward(10)
screen = turtle.Screen()
screen.listen()
screen.onkey(move_up, "Up")
screen.onkey(move_down, "Down")
screen.onkey(move_left, "Left")
screen.onkey(move_right, "Right")
screen.mainloop()
```