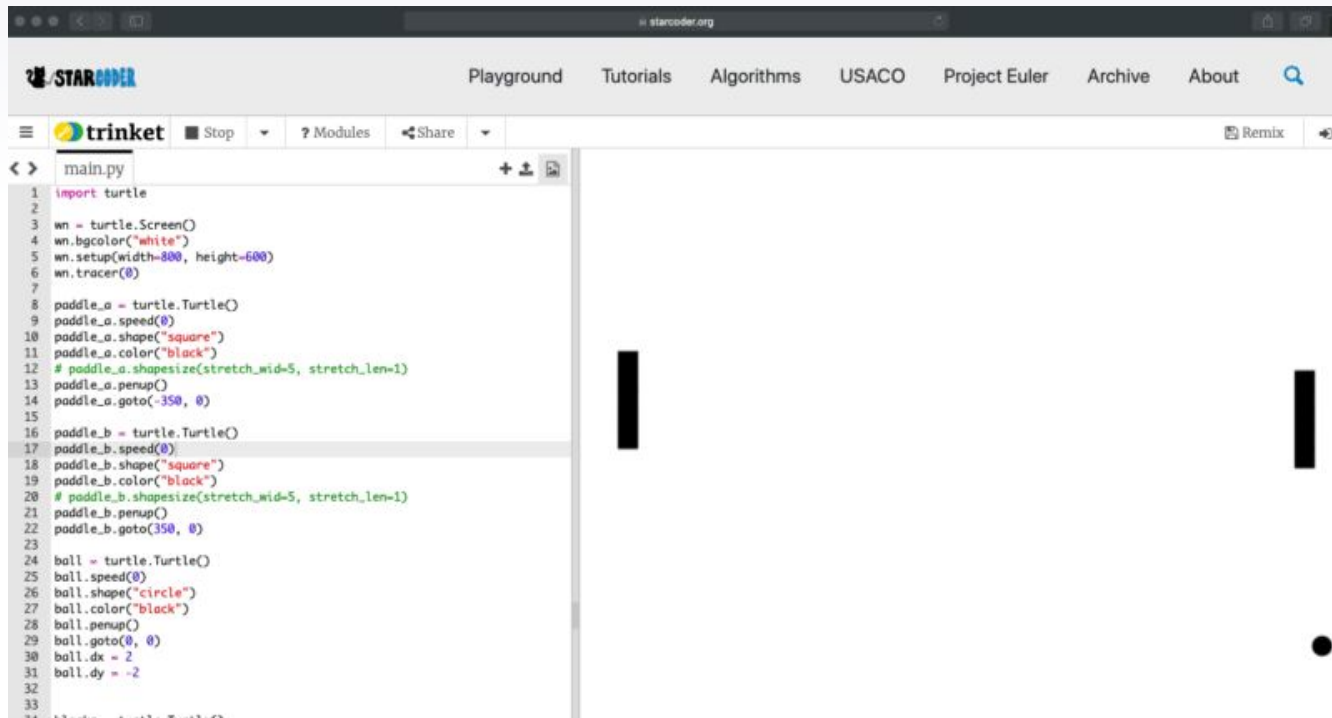




Welcome to Python in Projects: Pong

▶ Today's Lesson

▶ Pong!



```
1 import turtle
2
3 wn = turtle.Screen()
4 wn.bgcolor("white")
5 wn.setup(width=800, height=600)
6 wn.tracer(0)
7
8 paddle_a = turtle.Turtle()
9 paddle_a.speed(0)
10 paddle_a.shape("square")
11 paddle_a.color("black")
12 # paddle_a.shapesize(stretch_wid=5, stretch_len=1)
13 paddle_a.penup()
14 paddle_a.goto(-350, 0)
15
16 paddle_b = turtle.Turtle()
17 paddle_b.speed(0)
18 paddle_b.shape("square")
19 paddle_b.color("black")
20 # paddle_b.shapesize(stretch_wid=5, stretch_len=1)
21 paddle_b.penup()
22 paddle_b.goto(350, 0)
23
24 ball = turtle.Turtle()
25 ball.speed(0)
26 ball.shape("circle")
27 ball.color("black")
28 ball.penup()
29 ball.goto(0, 0)
30 ball.dx = 2
31 ball.dy = -2
32
33
```

Basic Turtle Review

```
import turtle
```

```
wn = turtle.Screen()
```

```
wn.bgcolor("white")
```

```
wn.setup(width=800, height=600)
```

```
wn.tracer(0)
```

← Turns turtle animations on



▶ Creating the First Paddle

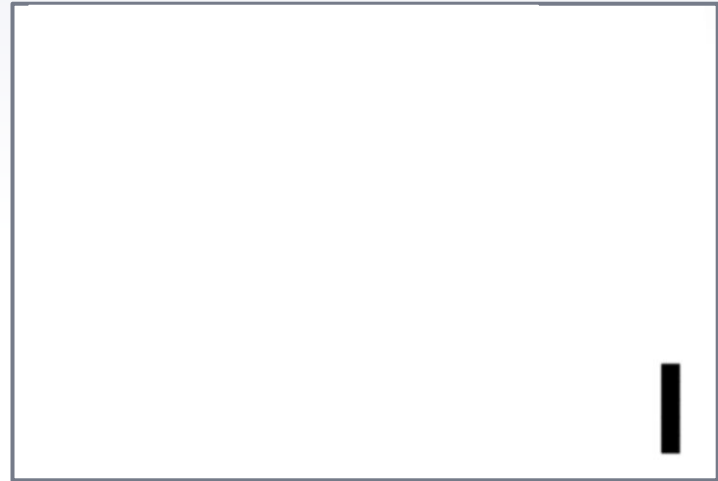
```
paddle_a = turtle.Turtle()  
paddle_a.speed(0)  
paddle_a.shape("square")  
paddle_a.color("black")  
paddle_a.penup()  
paddle_a.goto(-350, 0)
```



Still blank! Why?

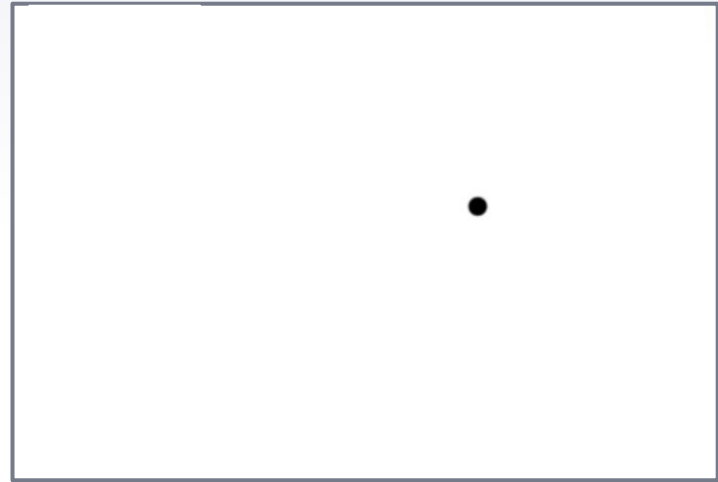
▶ Creating the Second Paddle

```
paddle_b = turtle.Turtle()  
paddle_b.speed(0)  
paddle_b.shape("square")  
paddle_b.color("black")  
paddle_b.penup()  
paddle_b.goto(350, 0)
```



▶ Creating the Ball

```
ball = turtle.Turtle()  
ball.speed(0)  
ball.shape("circle")  
ball.color("black")  
ball.penup()  
ball.goto(0, 0)  
ball.dx = 2  
ball.dy = -2
```



More Turtles

```
blocka = turtle.Turtle()
```

```
blockb = turtle.Turtle()
```

```
blockc = turtle.Turtle()
```

```
blockd = turtle.Turtle()
```

```
blocke = turtle.Turtle()
```

```
blockf = turtle.Turtle()
```

```
blockg = turtle.Turtle()
```

```
blockh = turtle.Turtle()
```

Movement For Each Block

```
def change_paddle_block(block,x, y):
```

```
~block.up()
```

```
~block.shape("square")
```

```
~block.goto(x, y)
```


▶ Right Paddle Movement

```
def draw_rec_right():  
    ~y = paddle_a.ycor()  
    ~x = paddle_a.xcor()  
    ~change_paddle_block(blocka, x, y + 20)  
    ~change_paddle_block(blockb, x, y + 40)  
    ~change_paddle_block(blockc, x, y - 20)  
    ~change_paddle_block(blockd, x, y - 40)
```

▶ Left Paddle Movement

```
def draw_rec_left():  
    ~y = paddle_b.ycor()  
    ~x = paddle_b.xcor()  
    ~change_paddle_block(blocke, x, y + 20)  
    ~change_paddle_block(blockf, x, y + 40)  
    ~change_paddle_block(blockg, x, y - 20)  
    ~change_paddle_block(blockh, x, y - 40)
```

► Paddle A Up

```
def paddle_a_up():  
    ~y = paddle_a.ycor()  
    ~y += 20  
    ~paddle_a.sety(y)
```

```
wn.listen()  
wn.onkey(paddle_a_up, "w")
```

► Paddle A Down

```
def paddle_a_down():
```

```
    ~y = paddle_a.ycor()
```

```
    ~y -= 20
```

```
    ~paddle_a.sety(y)
```

```
wn.onkey(paddle_a_down, "s")
```

Paddle B Up/Down

```
def paddle_b_up():
```

```
    ~y = paddle_b.ycor()
```

```
    ~y += 20
```

```
    ~paddle_b.sety(y)
```

```
wn.onkey(paddle_b_up,  
"Up")
```

```
def paddle_b_down():
```

```
    ~y = paddle_b.ycor()
```

```
    ~y -= 20
```

```
    ~paddle_b.sety(y)
```

```
wn.onkey(paddle_b_down,  
"Down")
```

► Block Clear

```
def block_clear():  
    ~for x in [blocka, blockb, blockc, blockd, blocke, blockf, blockg, blockh]:  
        ~~x.clear()
```

▶ Creating Main Loop

while True:

~wn.update() ← Update the screen

~draw_rec_right()

~draw_rec_left()

► Moving the Ball

```
~ball.setx(ball.xcor() + ball.dx)
```

```
~ball.sety(ball.ycor() + ball.dy)
```

Why does the ball not come back?

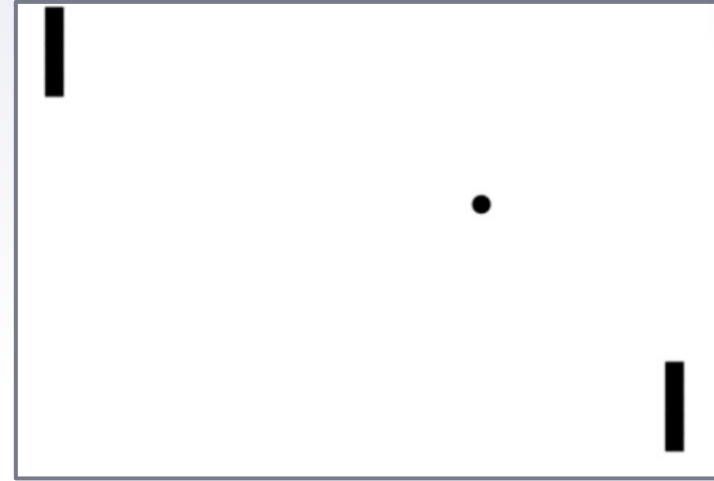
Edges

```
~if ball.ycor() > 290:  
~~ball.sety(290)  
~~ball.dy *= -1
```

```
~if ball.ycor() < -290:  
~~ball.sety(-290)  
~~ball.dy *= -1
```

```
~if ball.xcor() > 390:  
~~ball.goto(0, 0)  
~~ball.dx *= -1
```

```
~if ball.xcor() < -390:  
~~ball.goto(0, 0)  
~~ball.dx *= -1
```



Collisions

```
~if (ball.xcor() > 340 and ball.xcor() < 350) and (  
    ball.ycor() < paddle_b.ycor() + 40 and ball.ycor() > paddle_b.ycor() - 40):
```

```
~~ball.setx(340)
```

```
~~ball.dx *= -1
```

```
~if (ball.xcor() < -340 and ball.xcor() > -350) and (  
    ball.ycor() < paddle_a.ycor() + 40 and ball.ycor() > paddle_a.ycor() - 40):
```

```
~~ball.setx(-340)
```

```
~~ball.dx *= -1
```

► Final Piece!

`~block_clear()`