

# CS Bridge, Lecture 10

## Animation

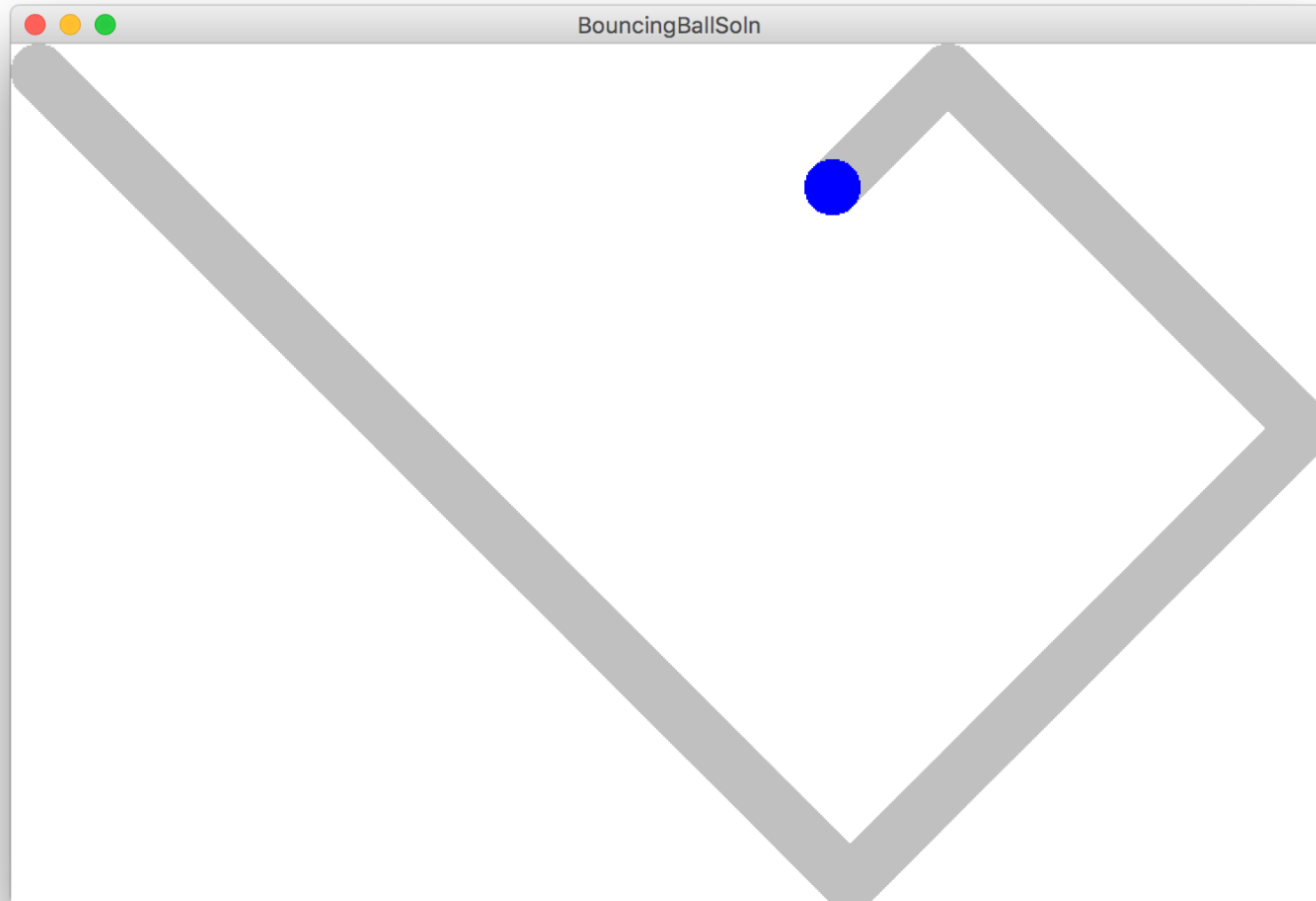


# Learning Goals

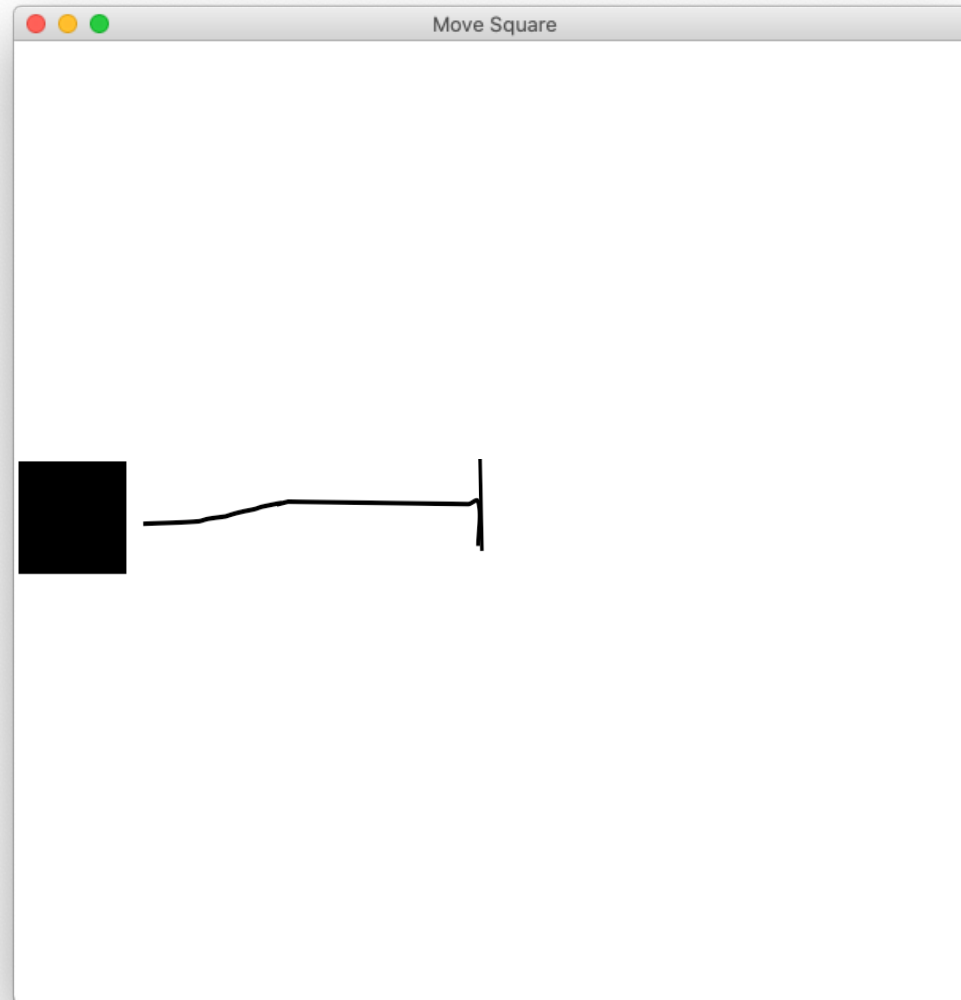
1. Get more practice writing programs with graphics
2. Understand how parameters are passed between functions
3. Write graphics programs with animation



# End Goal: Bouncing Ball!



# Checkpoint: "Move To Center"



# Lecture Plan

- **Review:** Graphics
- Animation Loop Structure
- **Example:** Move To Center
- **Practice:** Bouncing Ball
- Passing Parameters

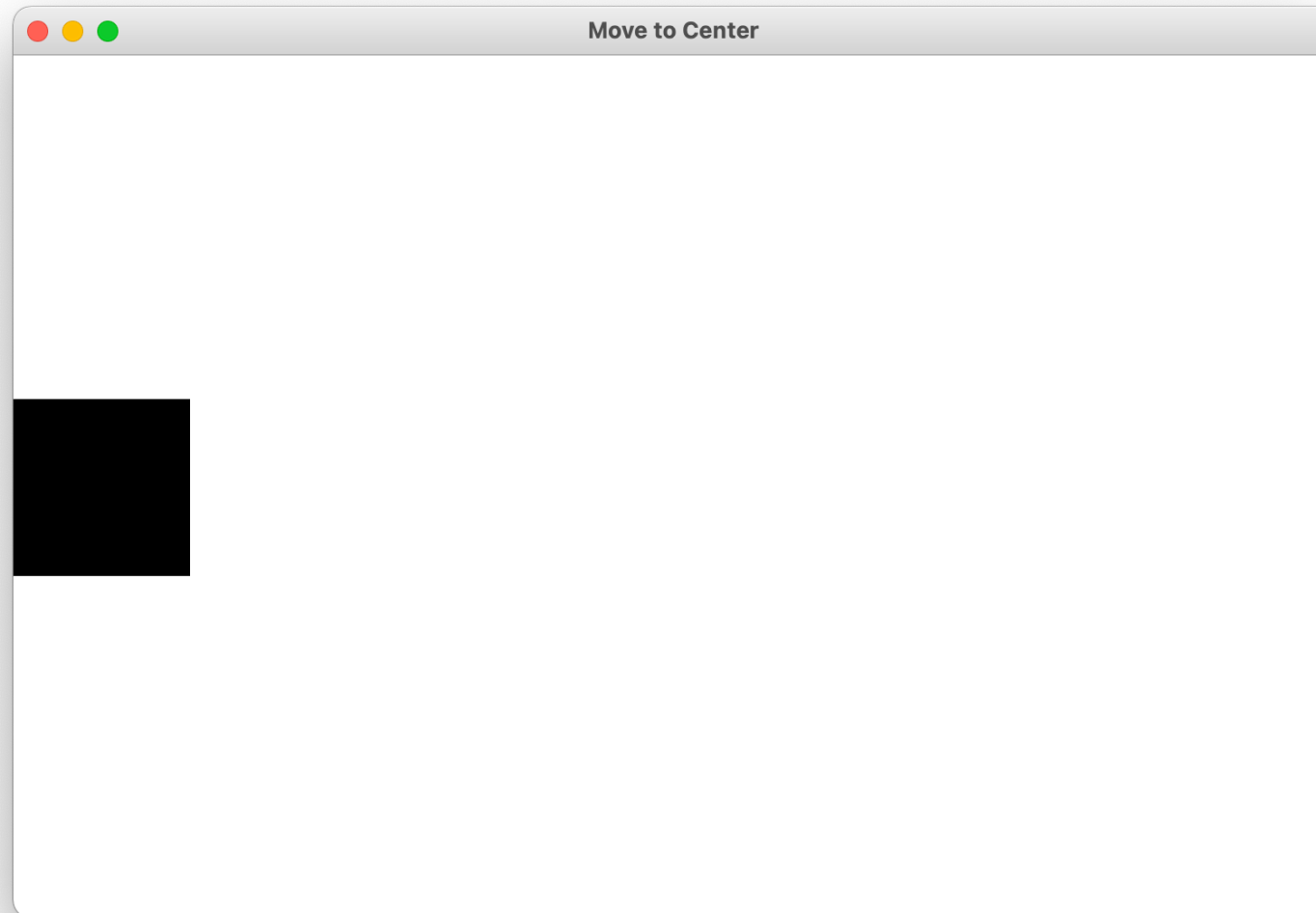
# Lecture Plan

- **Review:** Graphics
- Animation Loop Structure
- **Example:** Move To Center
- **Practice:** Bouncing Ball
- Passing Parameters

In our last episode...

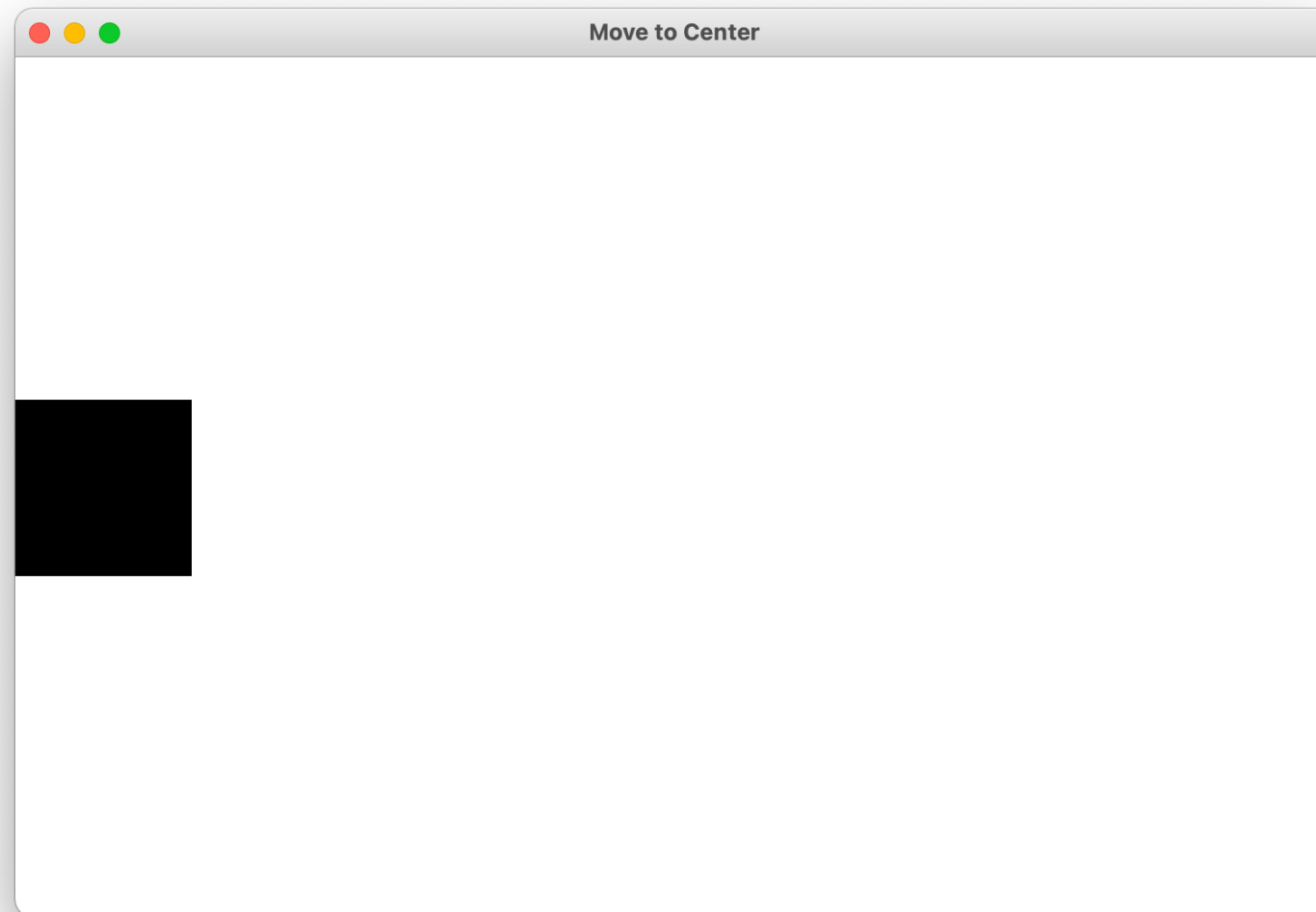
# Graphics From Tkinter

**from** graphics **import** Canvas





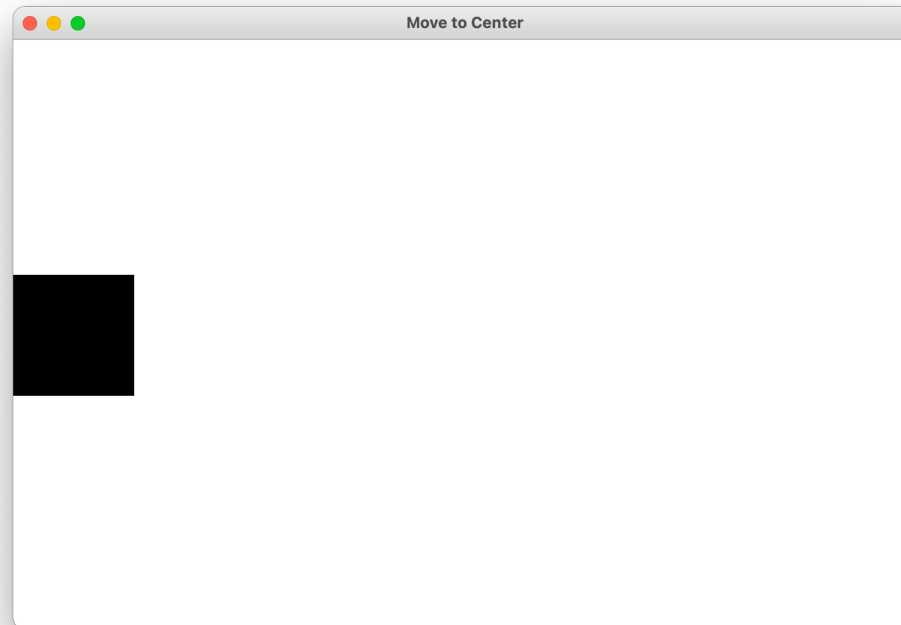
# Add Square



# Add Square

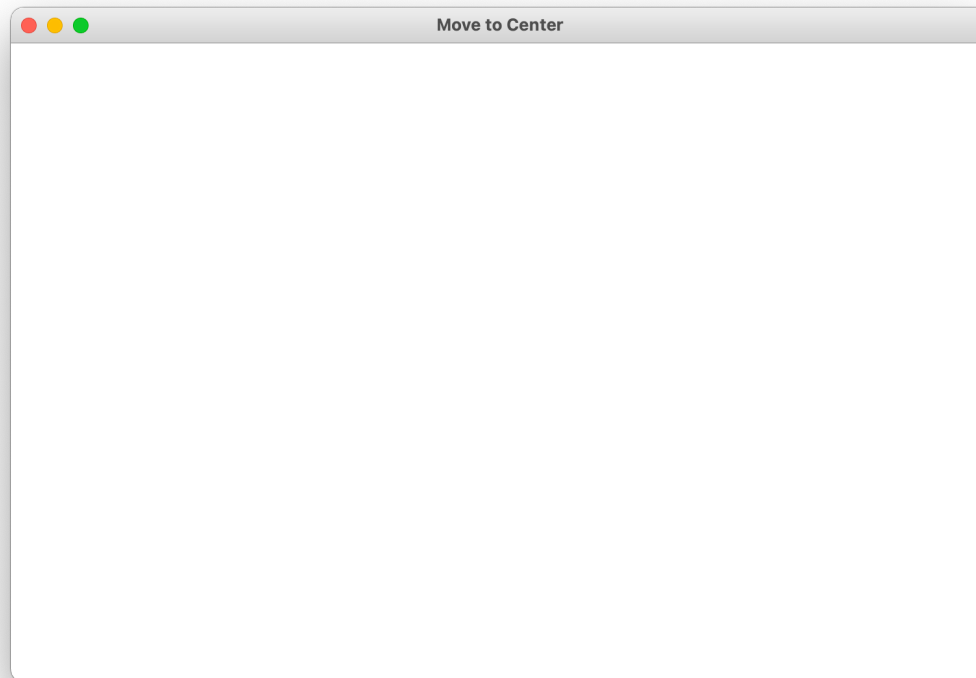
```
SQUARE_SIZE = 100
```

```
def main():  
    canvas = Canvas()  
    canvas.set_canvas_title("Move to Center")  
    square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2  
    rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)  
    canvas.set_color(rect, "black")  
    canvas.mainloop()
```



# Add Square

```
canvas = Canvas()
canvas.set_canvas_title("Move to Center")
square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2
rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)
canvas.set_color(rect, "black")
canvas.mainloop()
```



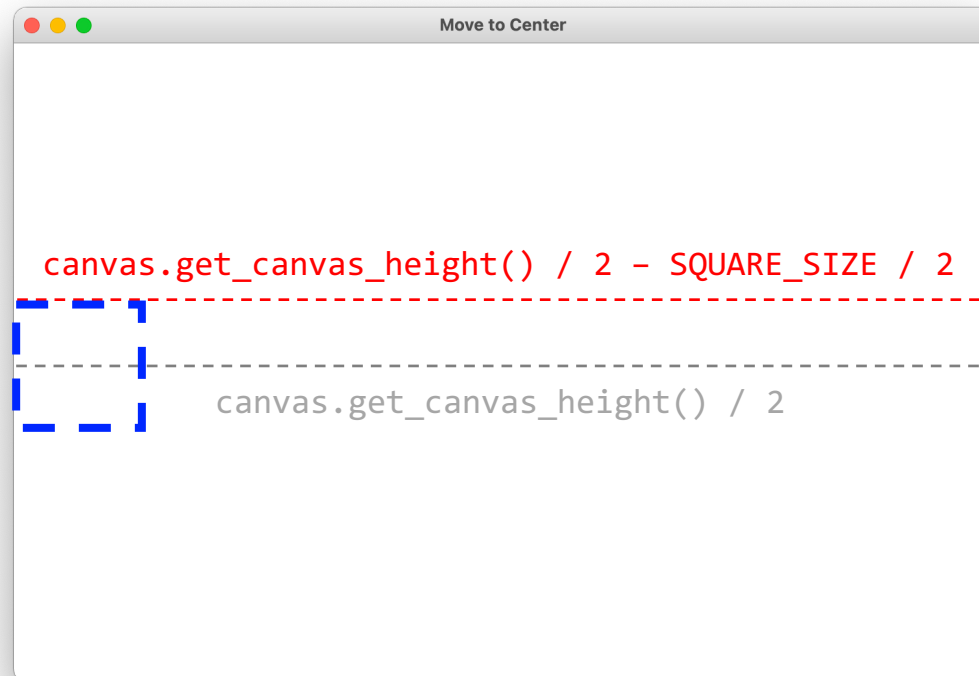
# Add Square

```
canvas = Canvas()
canvas.set_canvas_title("Move to Center")
square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2
rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)
canvas.set_color(rect, "black")
canvas.mainloop()
```



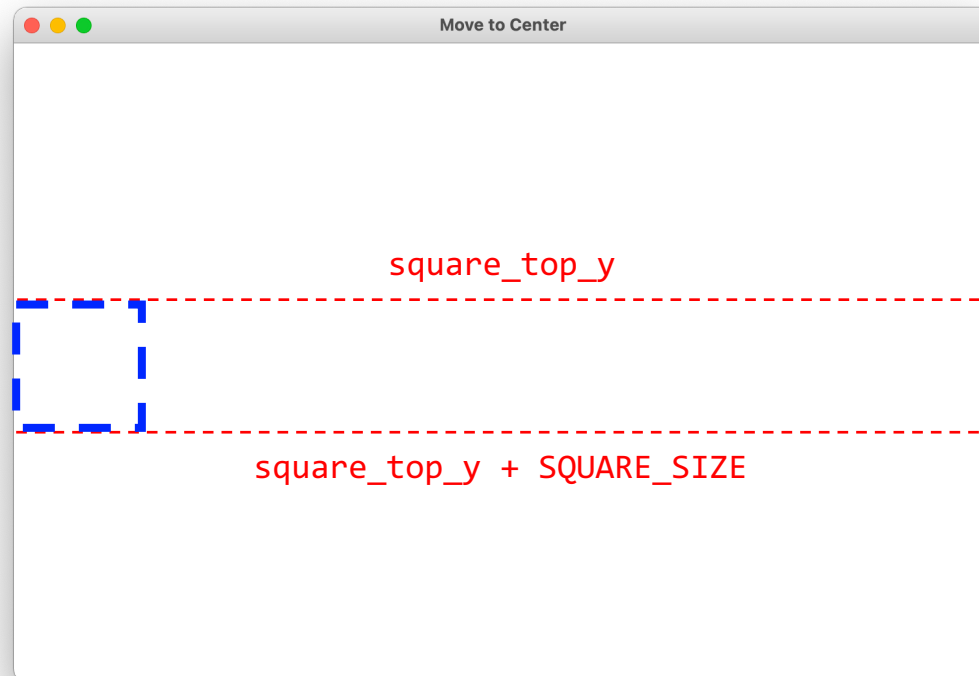
# Add Square

```
canvas = Canvas()
canvas.set_canvas_title("Move to Center")
square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2
rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)
canvas.set_color(rect, "black")
canvas.mainloop()
```



# Add Square

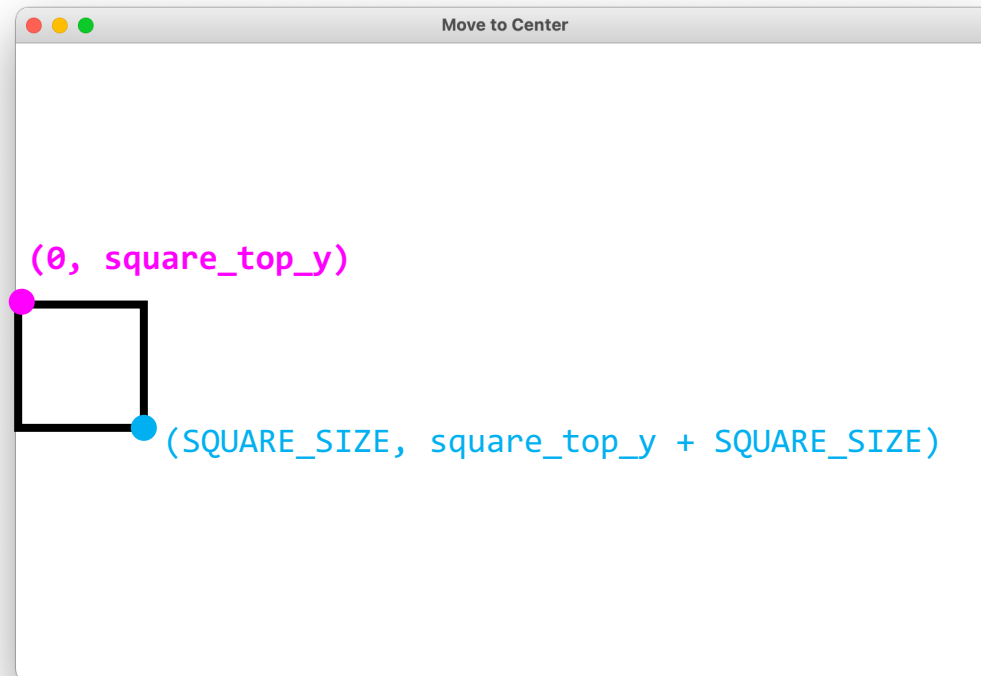
```
canvas = Canvas()
canvas.set_canvas_title("Move to Center")
square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2
rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)
canvas.set_color(rect, "black")
canvas.mainloop()
```



Some “heavy  
duty” variables  
allow you to call  
functions on them

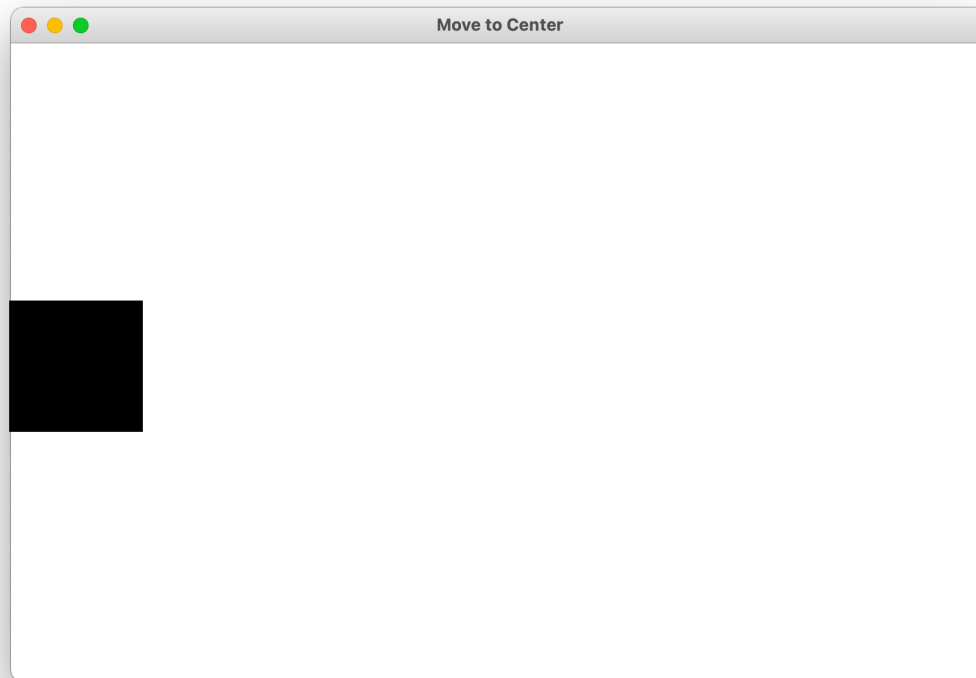
# Add Square

```
canvas = Canvas()
canvas.set_canvas_title("Move to Center")
square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2
rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)
canvas.set_color(rect, "black")
canvas.mainloop()
```



# Add Square

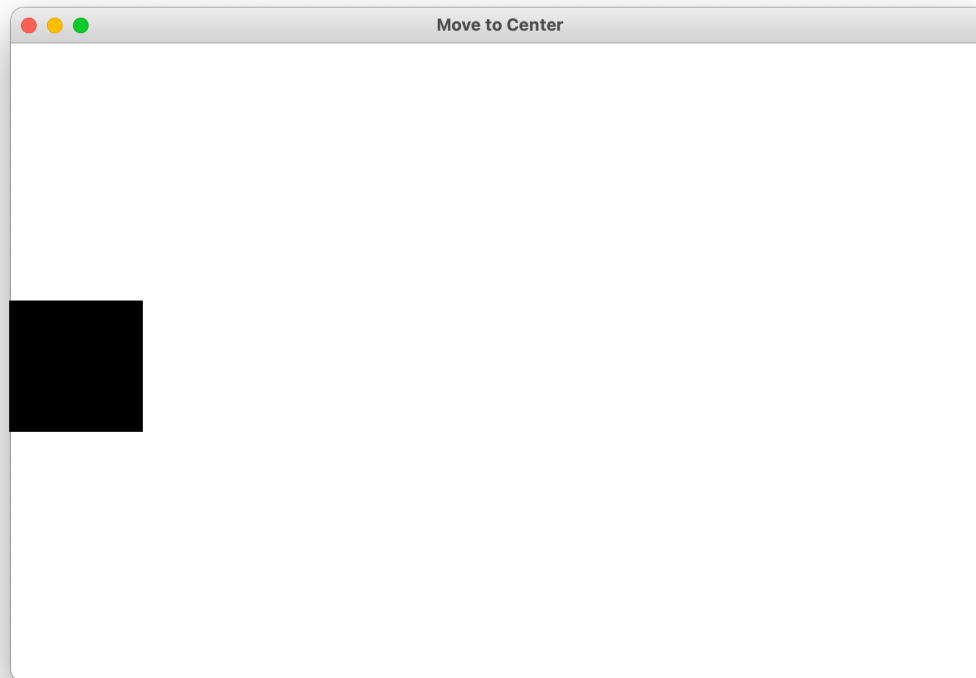
```
canvas = Canvas()
canvas.set_canvas_title("Move to Center")
square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2
rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)
canvas.set_color(rect, "black")
canvas.mainloop()
```





# Add Square

```
canvas = Canvas()  
canvas.set_canvas_title("Move to Center")  
square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2  
rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)  
canvas.set_color(rect, "black")  
canvas.mainloop()
```



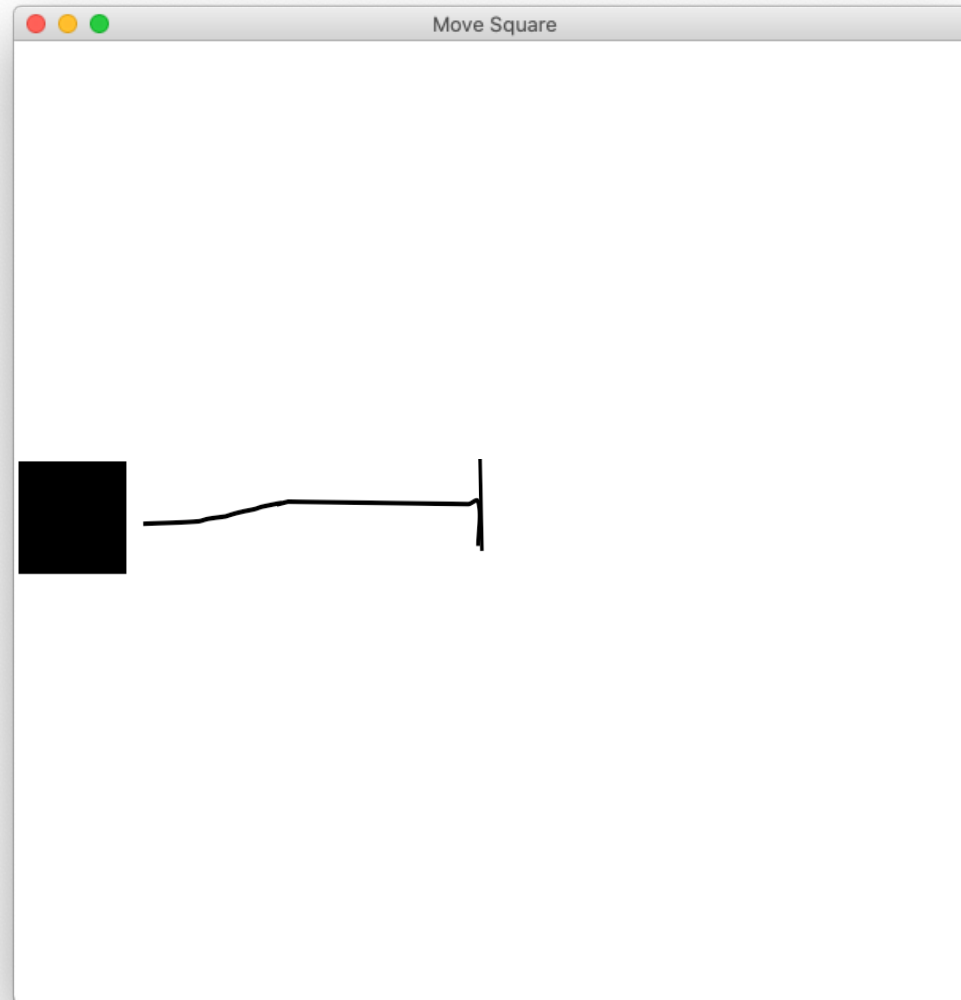
End of review!

# Lecture Plan

- **Review:** Graphics
- **Animation Loop Structure**
- **Example:** Move To Center
- **Practice:** Bouncing Ball
- Passing Parameters

How do movies or games  
animate?

# Checkpoint: "Move To Center"



# Animation Loop

```
def main():  
    # setup done once  
  
    while ???:  
        # update world  
  
        # pause  
        time.sleep(DELAY)  
  
    canvas.mainloop()
```

# Animation Loop

```
def main():  
    # setup done once  
  
    while ???:  
        # update world  
  
        # pause  
        time.sleep(DELAY)  
  
    canvas.mainloop()
```

Make all the variables  
you need.

# Animation Loop

```
def main():  
    # setup done once  
    while ???:  
        # update world  
  
        # pause  
        time.sleep(DELAY)  
  
    canvas.mainloop()
```

The animation loop is a repetition of heartbeats, either forever (while True) or until some condition is no longer true.



# Animation Loop

```
def main():  
    # setup done once  
  
    while ???:  
        # update world  
  
        # pause  
        time.sleep(DELAY)  
  
    canvas.mainloop()
```

Each heart-beat, update the world forward one frame

# Animation Loop

```
def main():  
    # setup done once  
  
    while ???:  
        # update world  
        # pause  
        time.sleep(DELAY)  
  
    canvas.mainloop()
```

If you don't pause, humans  
won't be able to see it!

# Animation Loop

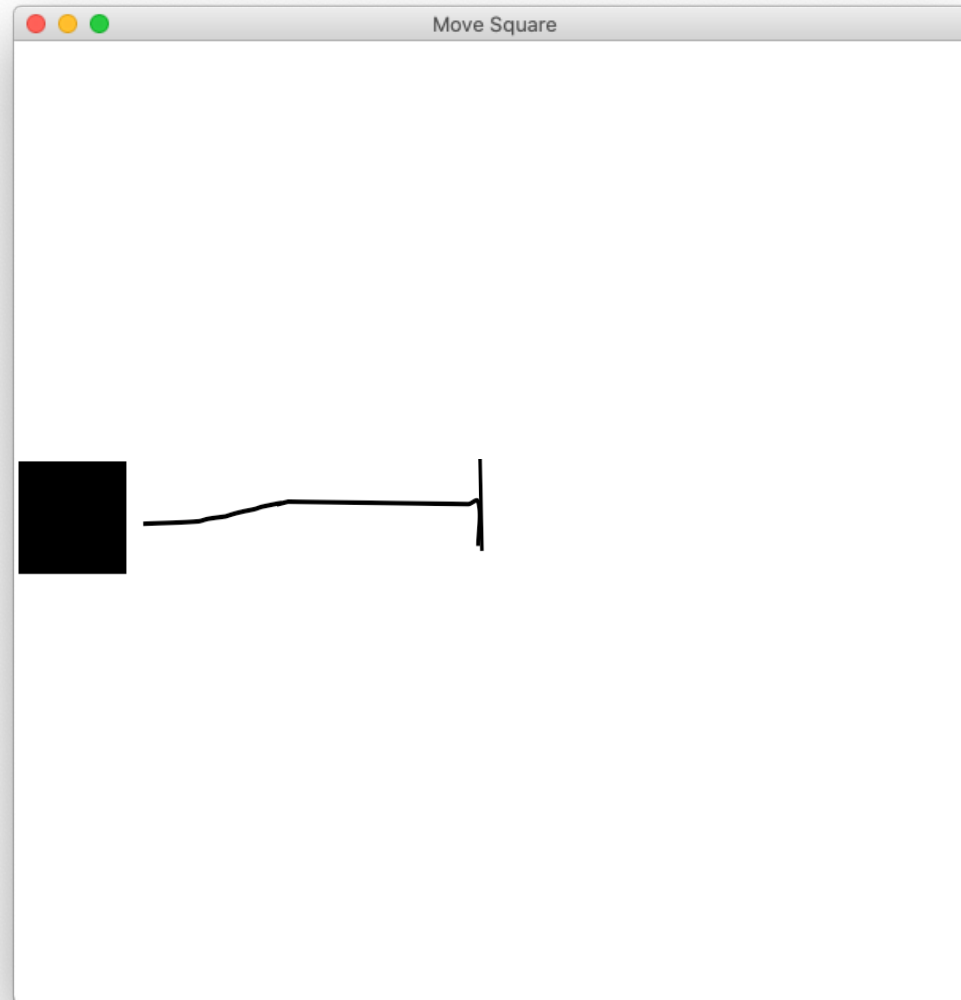
```
def main():  
    # setup done once  
  
    while ???:  
        # update world  
  
        # pause  
        time.sleep(DELAY)  
    canvas.mainloop()
```

Make sure to call  
mainloop() to make your  
program run correctly.

# Lecture Plan

- Review: Graphics
- Animation Loop Structure
- **Example: Move To Center**
- Practice: Bouncing Ball
- Passing Parameters

# Checkpoint: "Move To Center"



# Animation Loop

```
def main():  
    # setup done once  
  
    while ???:  
        # update world  
  
        # pause  
        time.sleep(DELAY)  
  
    canvas.mainloop()
```

# Animation Loop

```
def main():  
    # setup done once  
    canvas = Canvas()  
    canvas.set_canvas_title("Move to Center")  
    square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2  
    rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)  
    canvas.set_color(rect, "black")  
  
    while ???:  
        # update world  
  
        # pause  
        time.sleep(DELAY)  
  
    canvas.mainloop()
```

# Animation Loop

```
def main():
    # setup done once
    canvas = Canvas()
    canvas.set_canvas_title("Move to Center")
    square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2
    rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)
    canvas.set_color(rect, "black")

    while ???:
        # update world
        canvas.move(rect, 1, 0)
        canvas.update()

        # pause
        time.sleep(DELAY)

    canvas.mainloop()
```



# Animation Loop

```
def main():
    # setup done once
    canvas = Canvas()
    canvas.set_canvas_title("Move to Center")
    square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2
    rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)
    canvas.set_color(rect, "black")

    while ???:
        # update world
        canvas.move(rect, 1, 0)  # Move the rectangle 1 pixel to the right.
        canvas.update()

        # pause
        time.sleep(DELAY)

    canvas.mainloop()
```

# Animation Loop

```
def main():  
    # setup done once  
    canvas = Canvas()  
    canvas.set_canvas_title("Move to Center")  
    square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2  
    rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)  
    canvas.set_color(rect, "black")  
  
    while ???:  
        # update world  
        canvas.move(rect, 1, 0)  
        canvas.update()  
  
        # pause  
        time.sleep(DELAY)  
  
    canvas.mainloop()
```

Tells the canvas to update the screen. Don't forget this!  
Call it once you are finished making all canvas changes  
for now.

# Animation Loop

```
def main():  
    # setup done once  
    canvas = Canvas()  
    canvas.set_canvas_title("Move to Center")  
    square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2  
    rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)  
    canvas.set_color(rect, "black")  
  
    while ???:  
        # update world  
        canvas.move(rect, 1, 0)  
        canvas.update()  
  
        # pause  
        time.sleep(DELAY)  
  
    canvas.mainloop()
```

When do we want to stop the animation loop?

Let's live code!

# Move to Center

```
def main():
    # setup done once
    canvas = Canvas()
    canvas.set_canvas_title("Move to Center")
    square_top_y = canvas.get_canvas_height() / 2 - SQUARE_SIZE / 2
    rect = canvas.create_rectangle(0, square_top_y, SQUARE_SIZE, square_top_y + SQUARE_SIZE)
    canvas.set_color(rect, "black")

    while is_not_past_center(canvas, rect):
        # update world
        canvas.move(rect, 1, 0)
        canvas.update()

        # pause
        time.sleep(DELAY)

    canvas.mainloop()
```

# More Helpful Graphics Functions

<code>canvas.move(obj, dx, dy)</code>	Moves obj using the displacements dx and dy.
<code>canvas.moveto(obj, x, y)</code>	Sets the location of obj to the specified coordinates.

# move shape to some new coordinates

```
canvas.moveto(shape, new_x, new_y)
```

# move shape by a given change\_x and change\_y

```
canvas.move(shape, change_x, change_y)
```

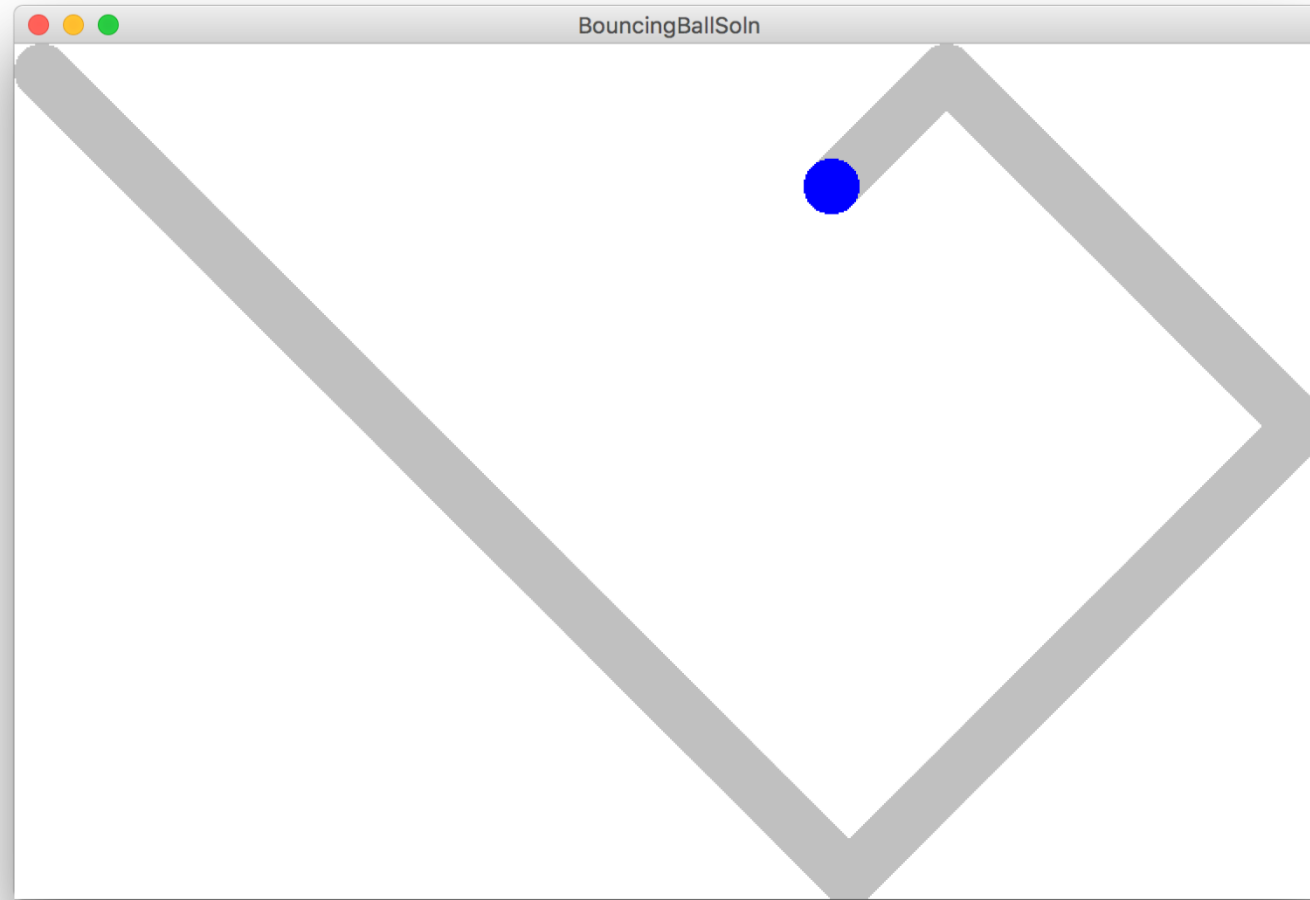
We are ready...

# Lecture Plan

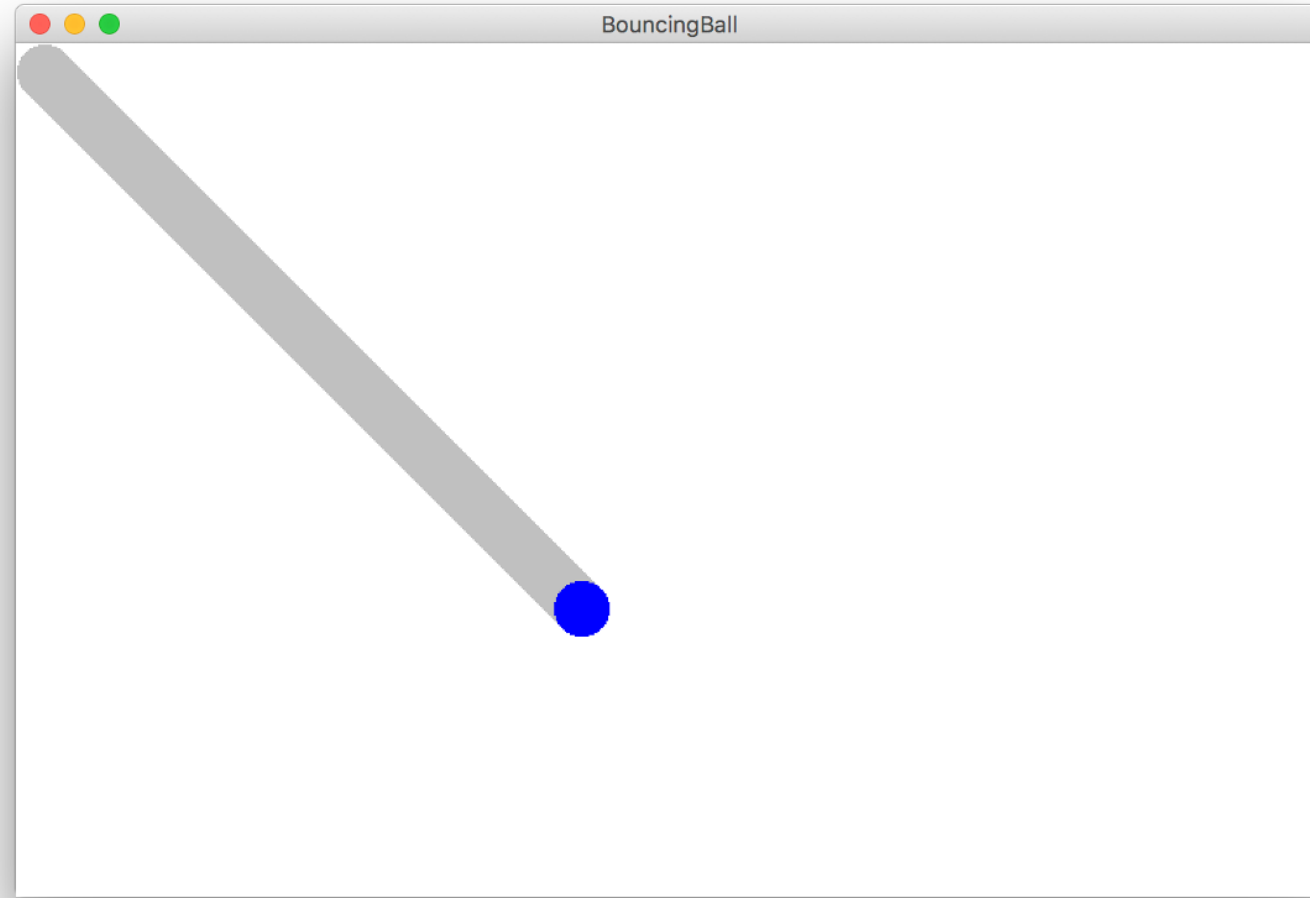
- Review: Graphics
- Animation Loop Structure
- Example: Move To Center
- **Practice: Bouncing Ball**
- Passing Parameters



# Bouncing Ball



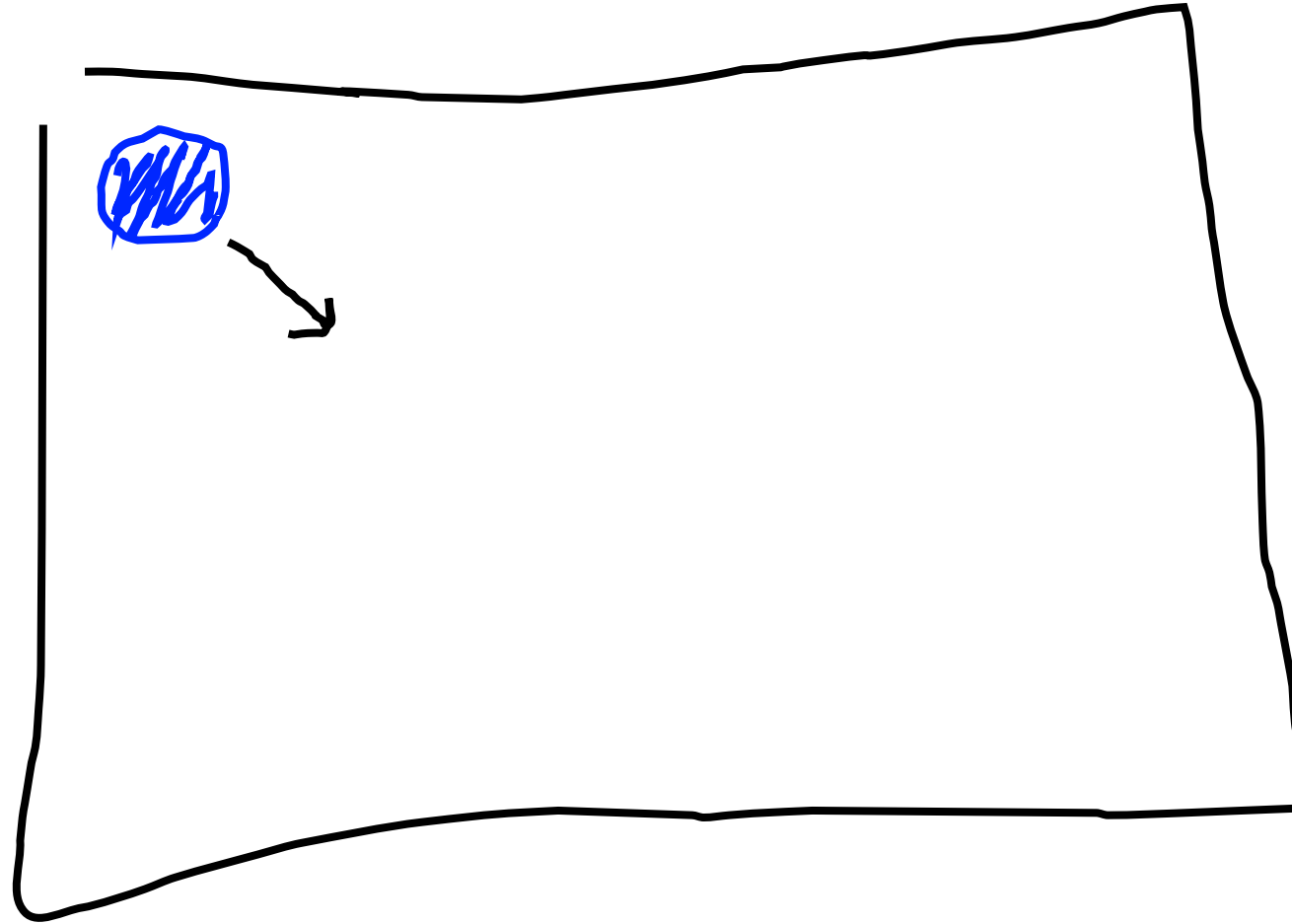
# Bouncing Ball: Milestone 1 - Movement



Let's try it!

# Bouncing Ball

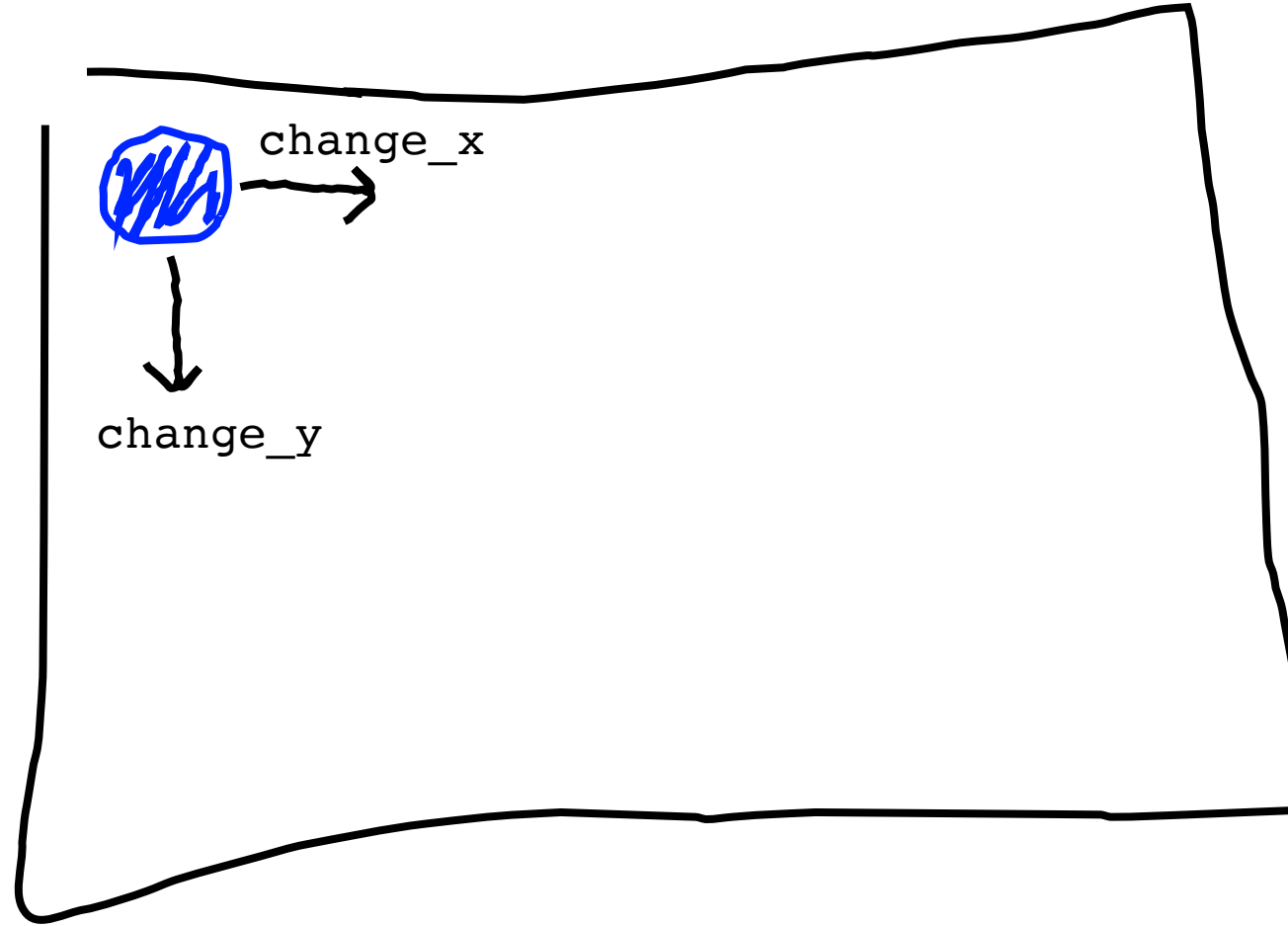
First heartbeat



**Key variable:** how much the ball position  
change each heartbeat?

# Bouncing Ball

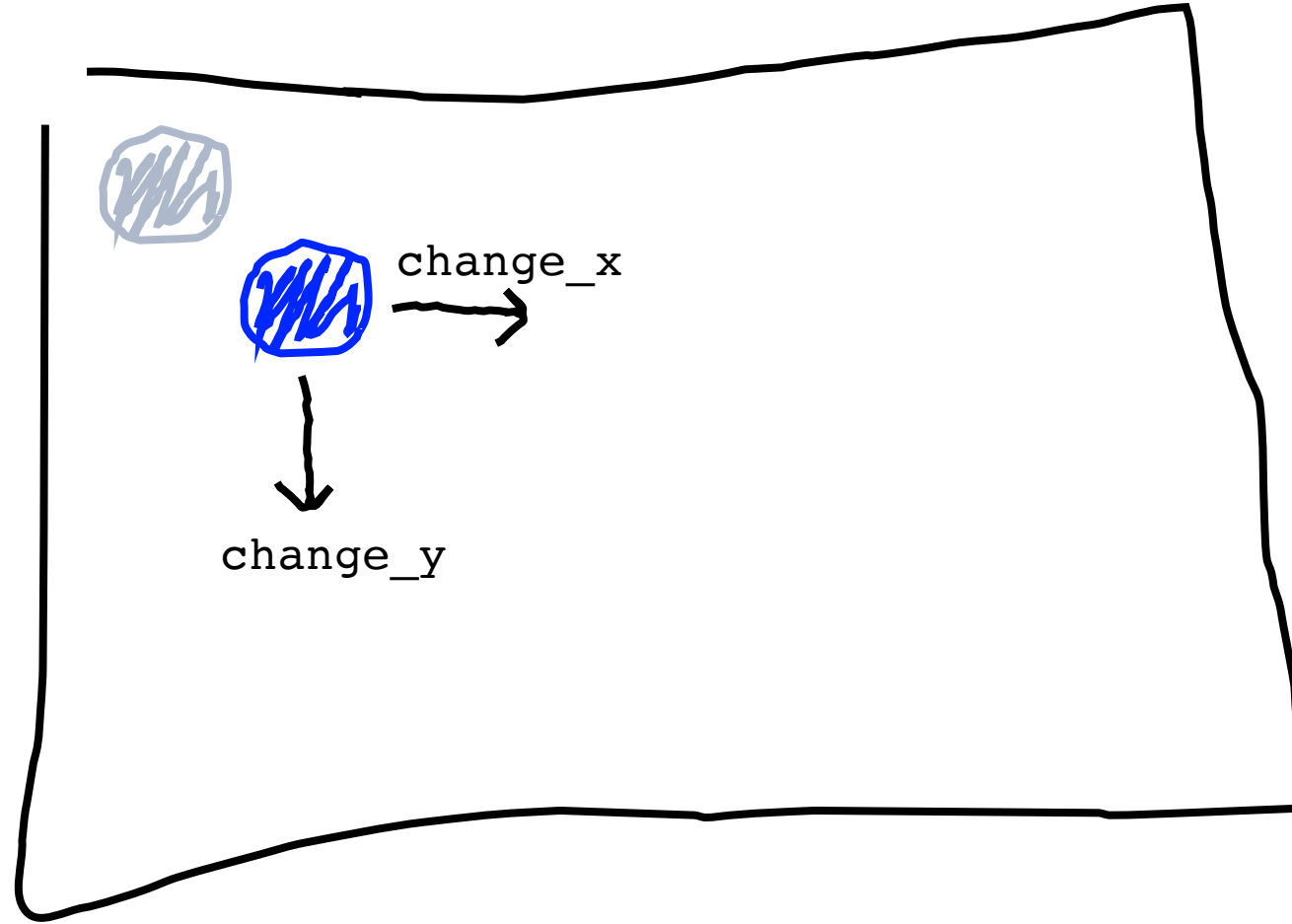
First heartbeat



The **move** function takes in  
a change in x and a change in y

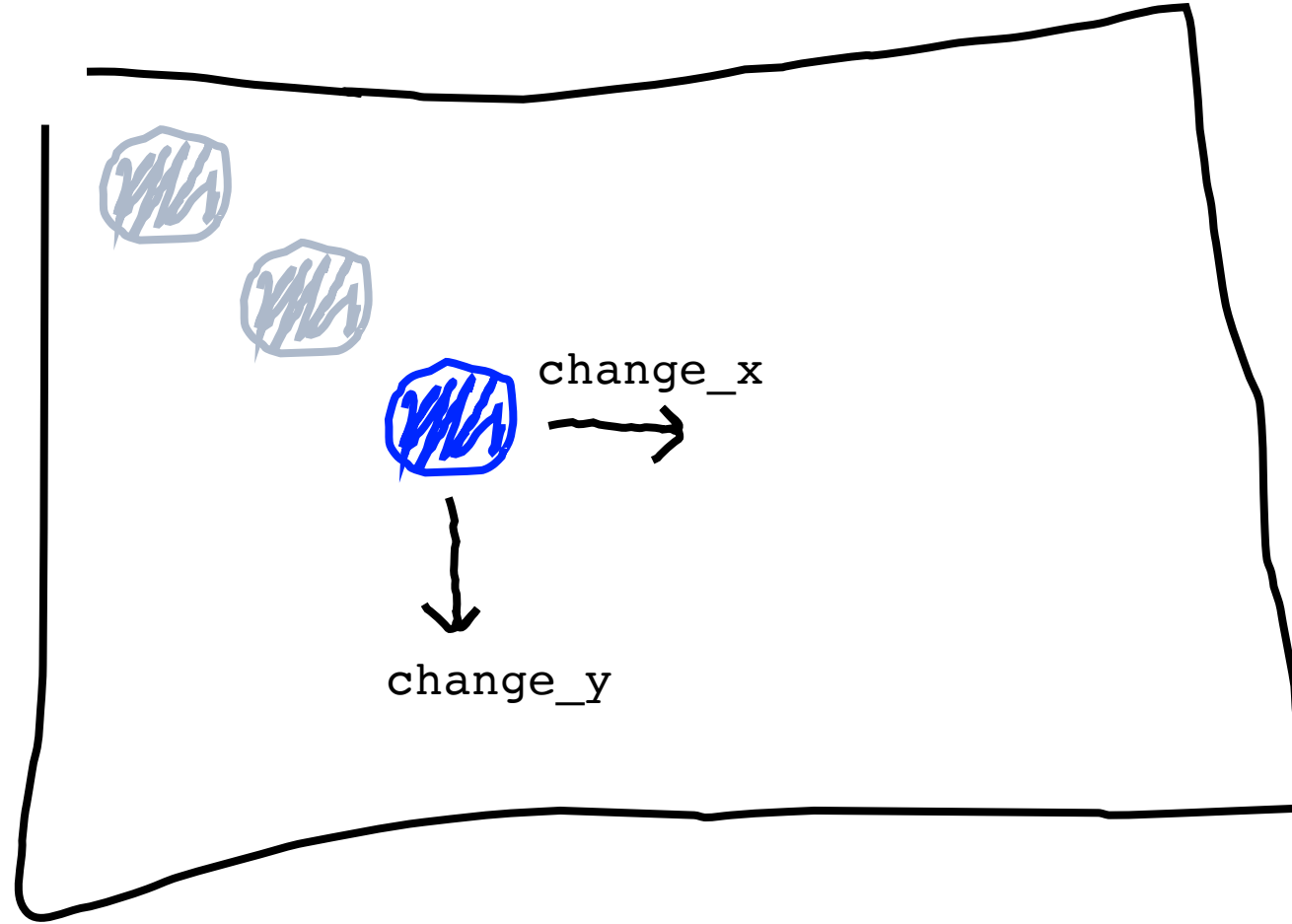
# Bouncing Ball

Second heartbeat



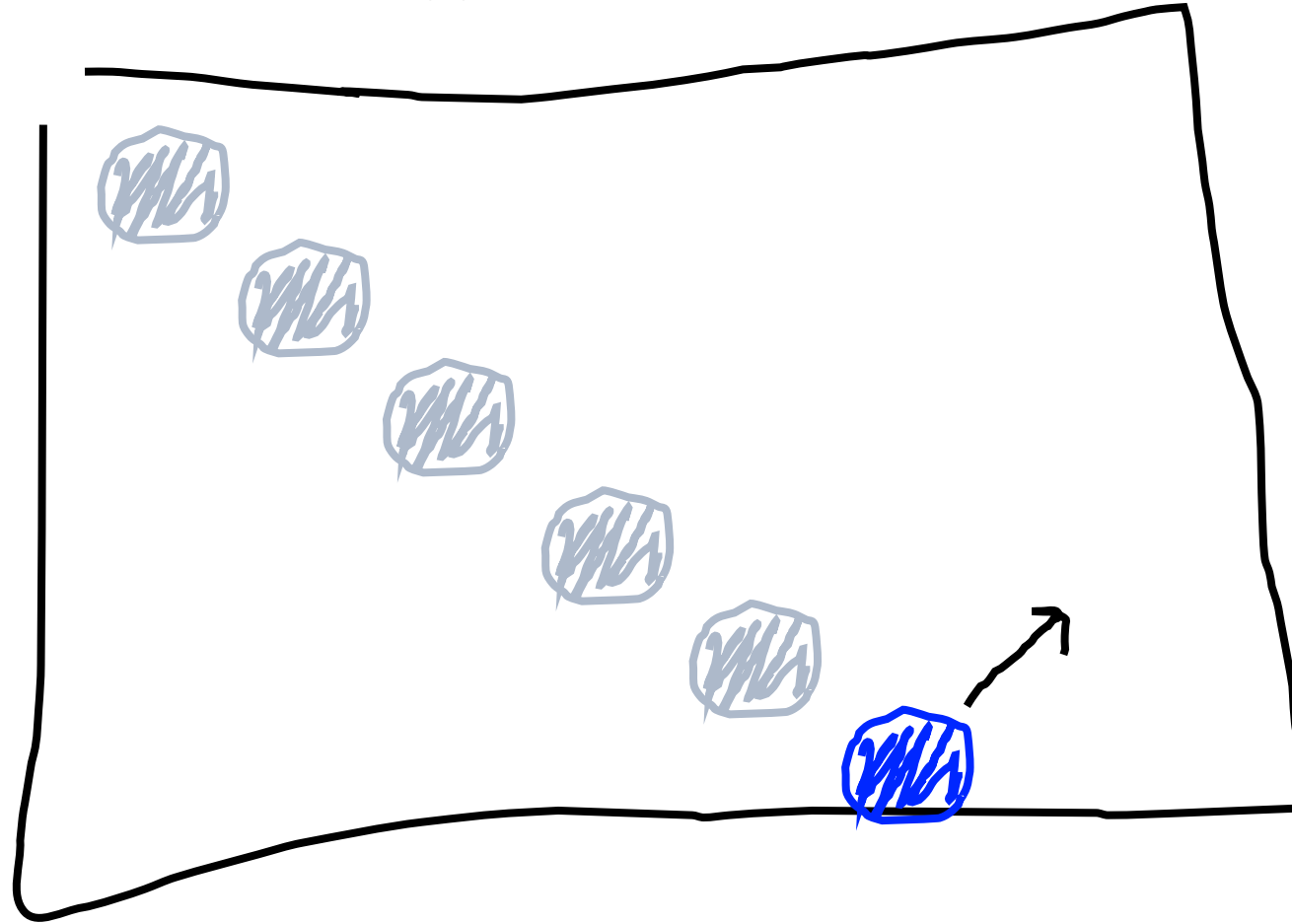
# Bouncing Ball

Third heartbeat



# Bouncing Ball

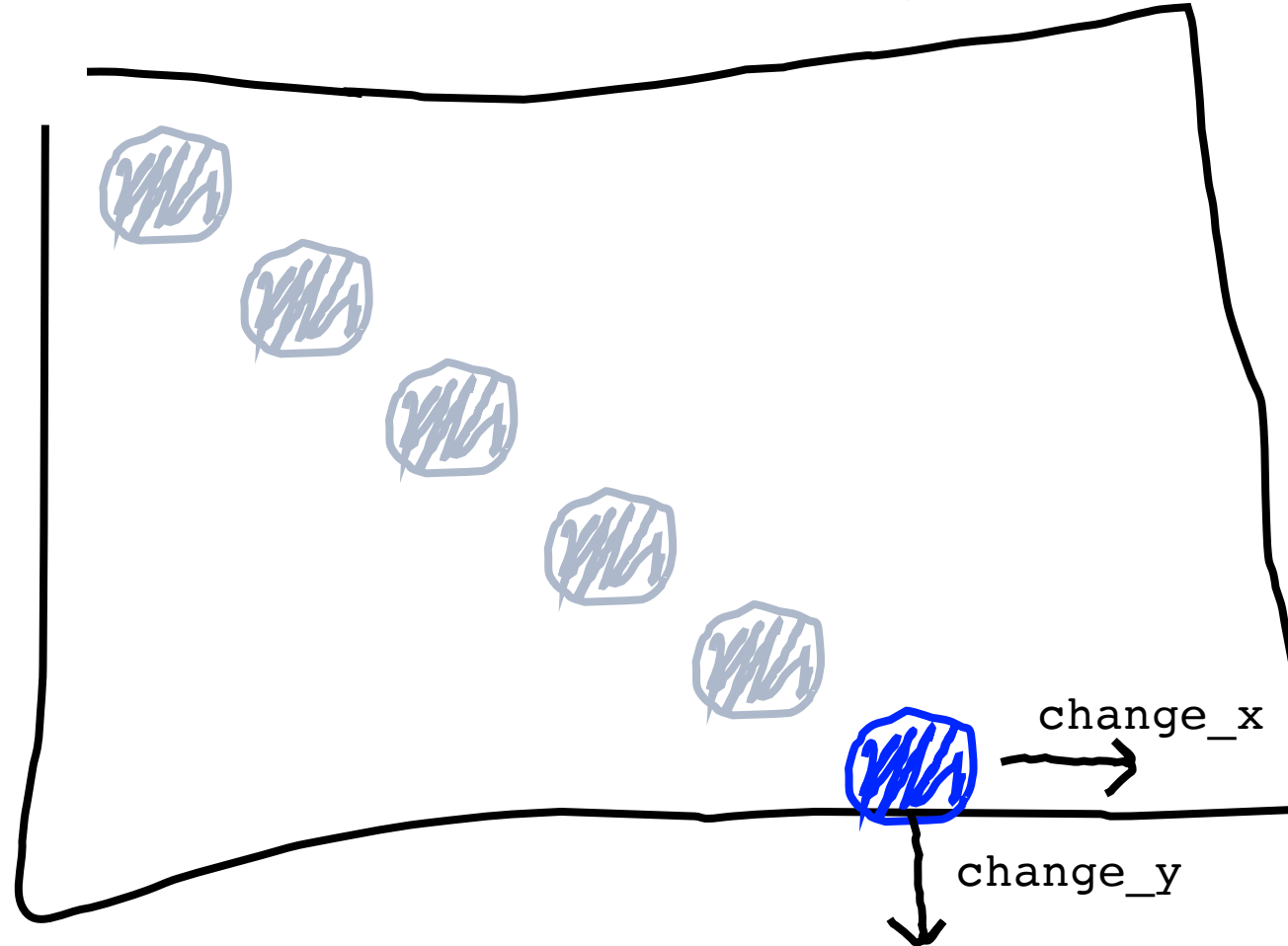
What happens when we hit a wall?





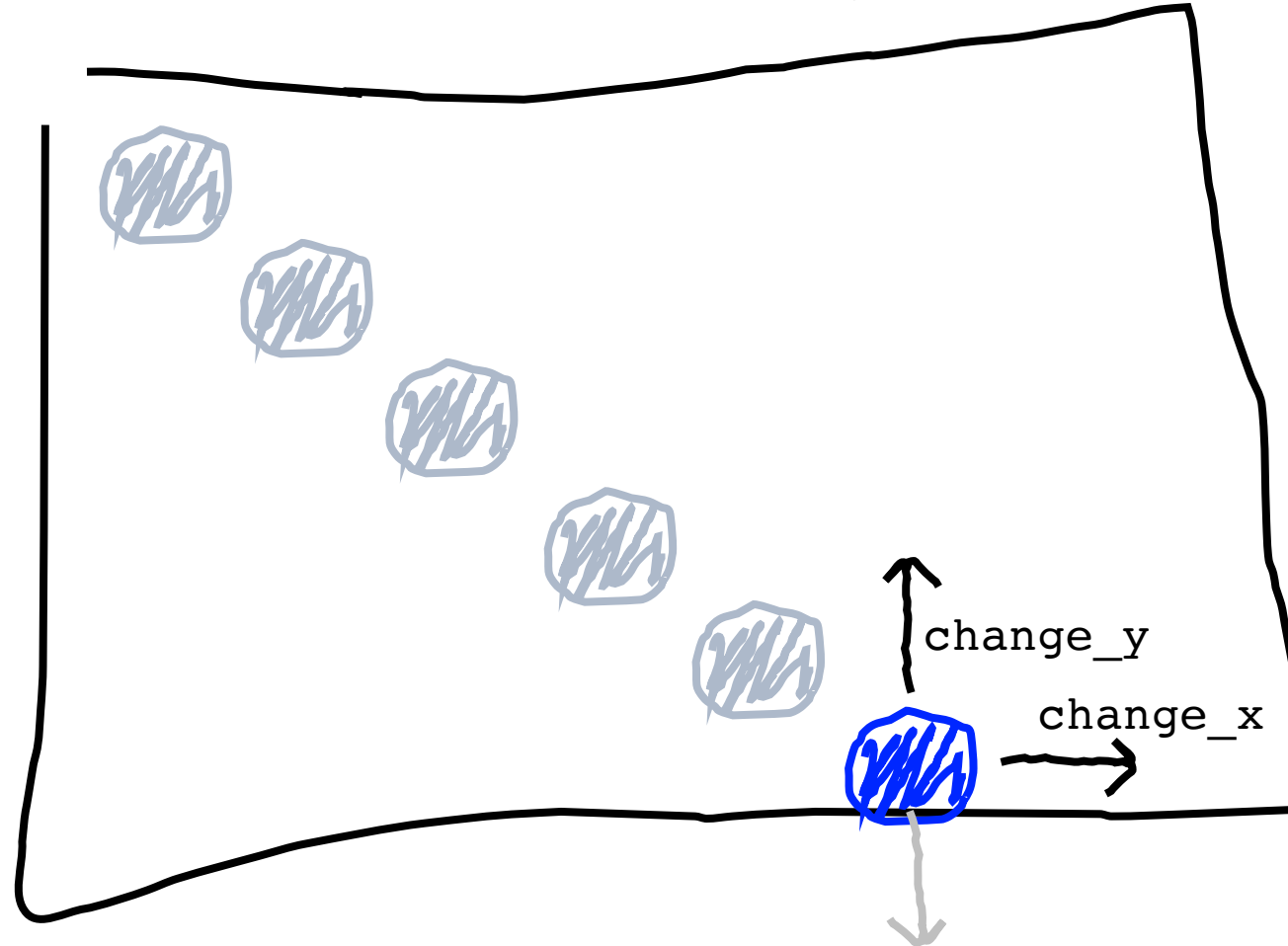
# Bouncing Ball

We have this velocity



# Bouncing Ball

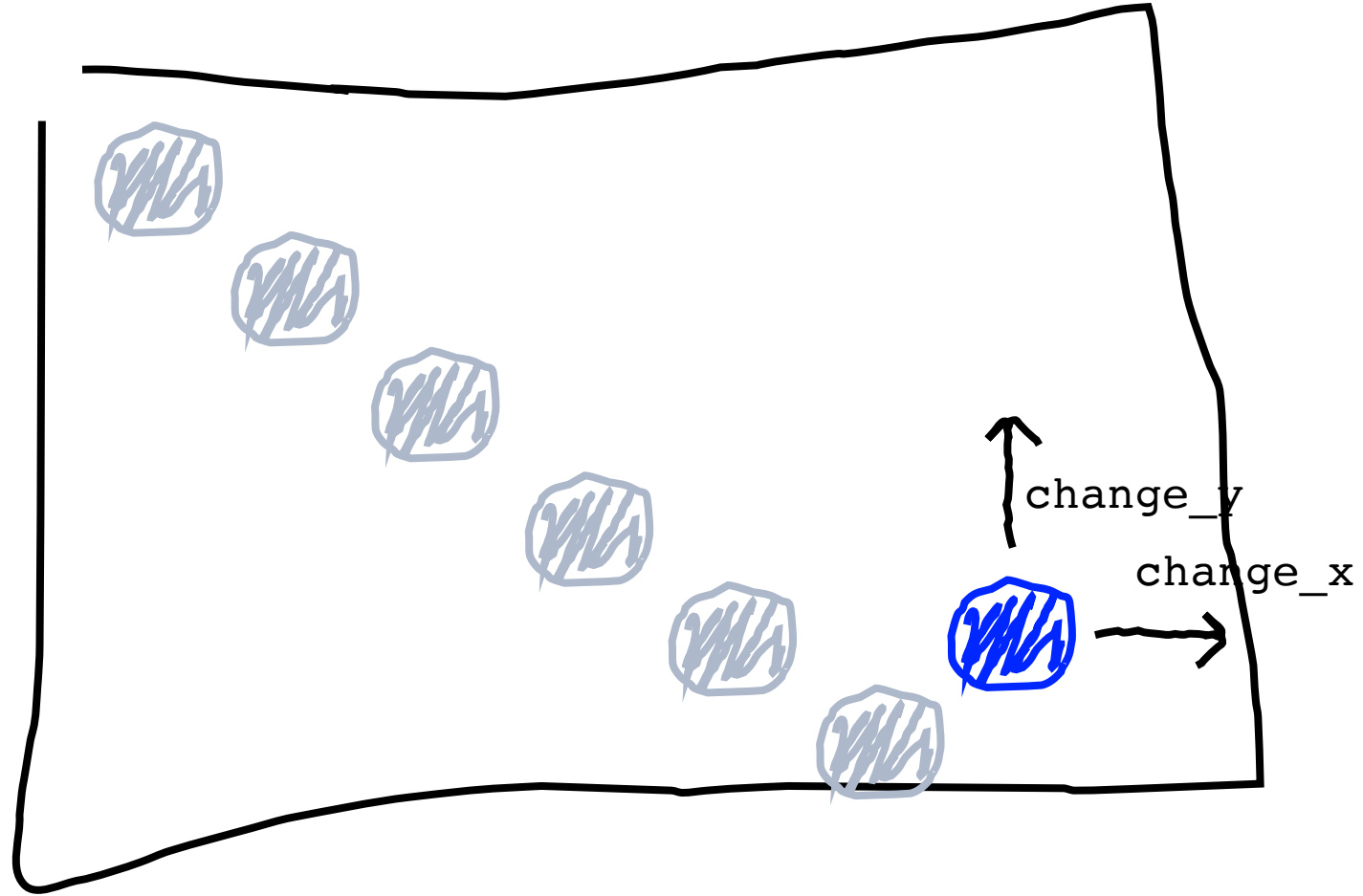
Our new velocity



When reflecting vertically:  $\text{change\_y} = -\text{change\_y}$

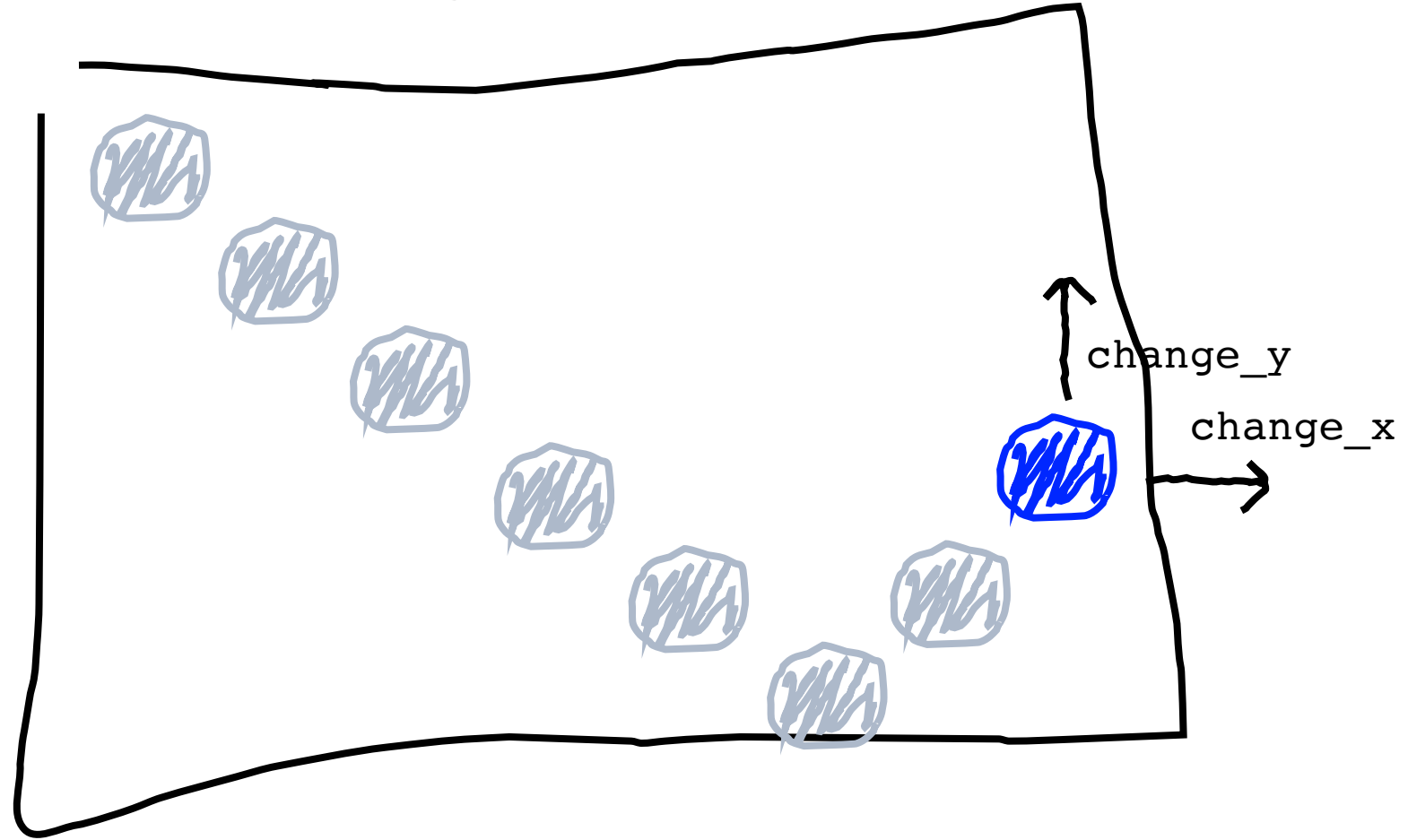
# Bouncing Ball

Seventh heartbeat



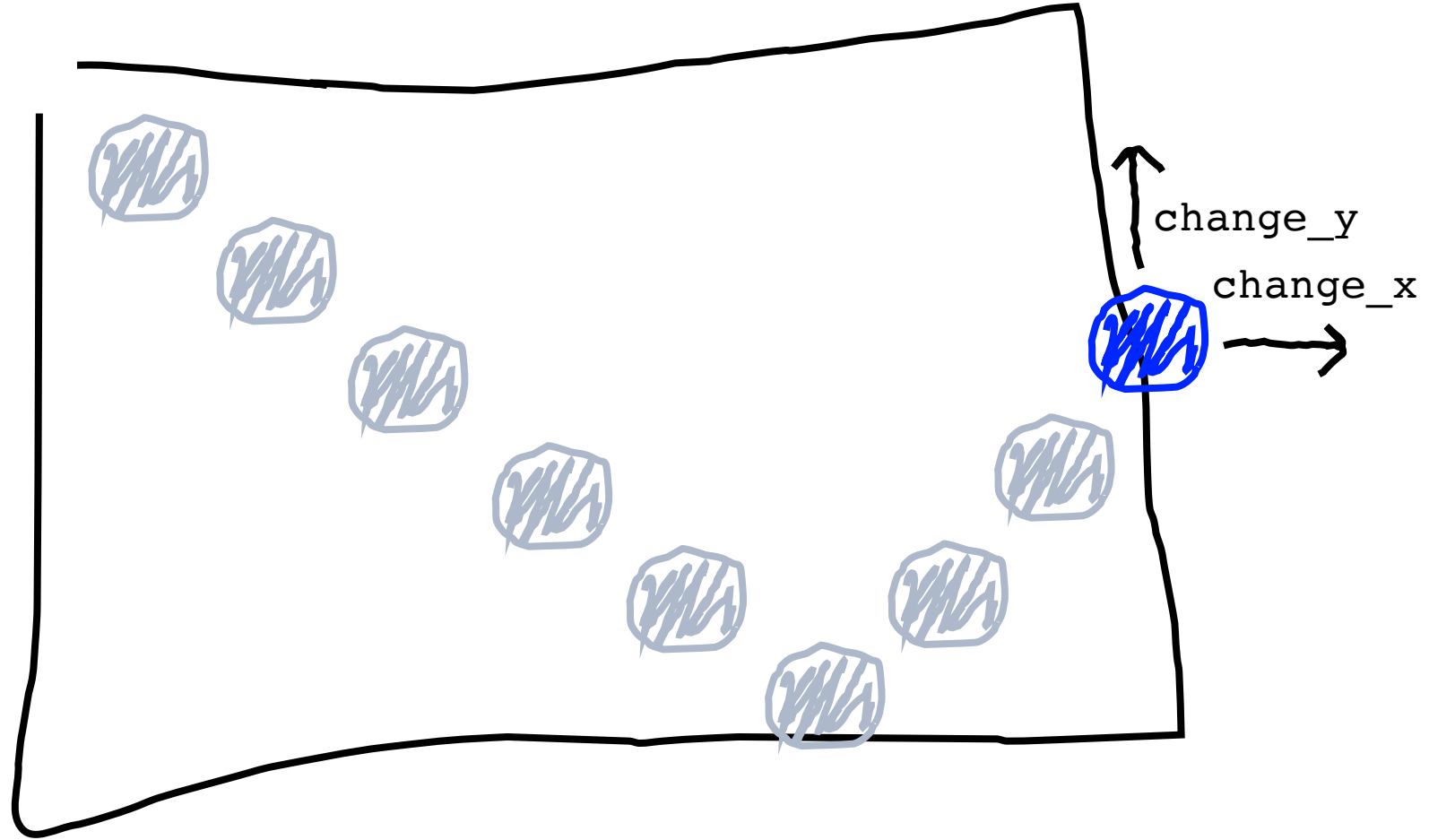
# Bouncing Ball

Eighth heartbeat



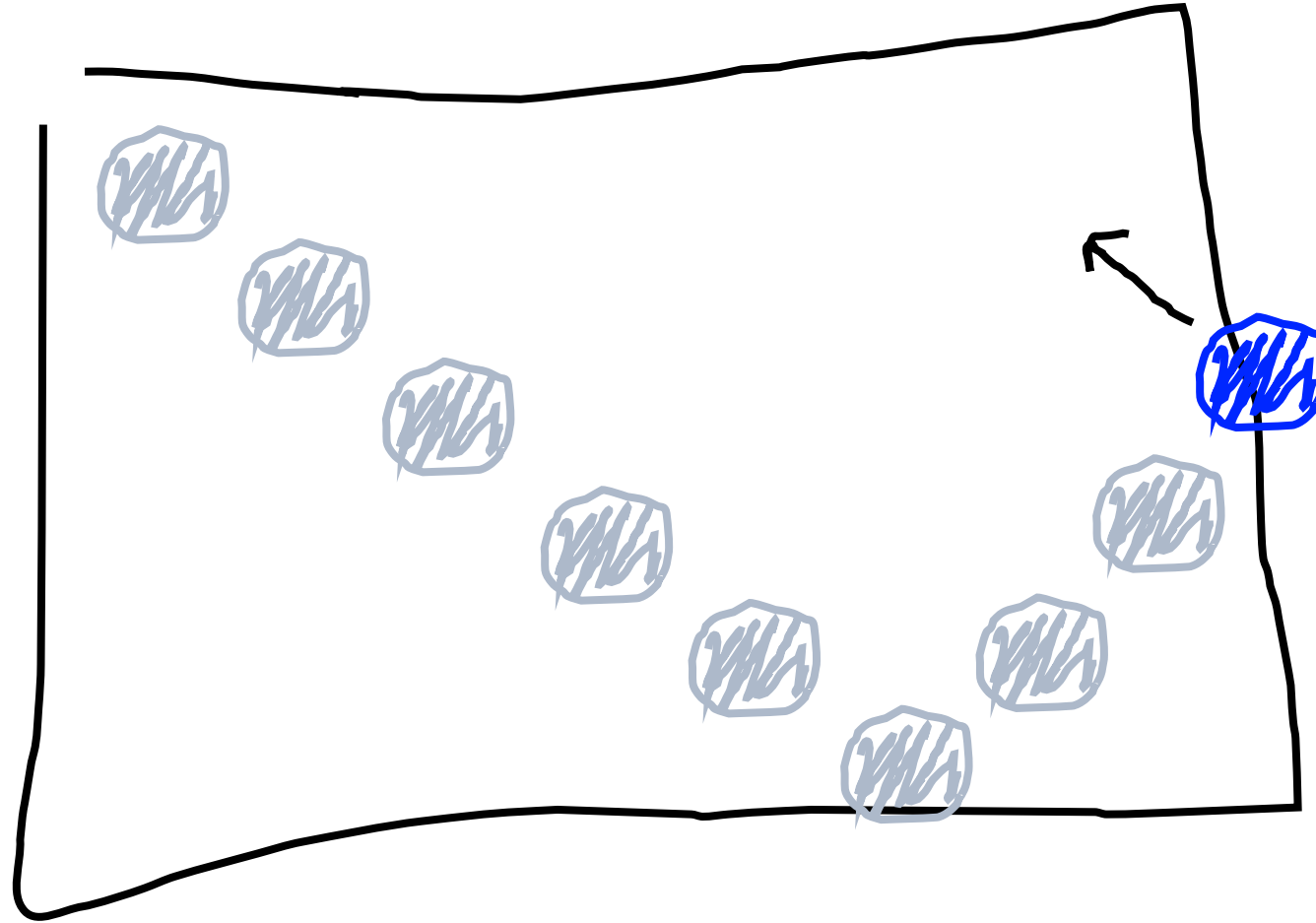
# Bouncing Ball

Ninth heartbeat



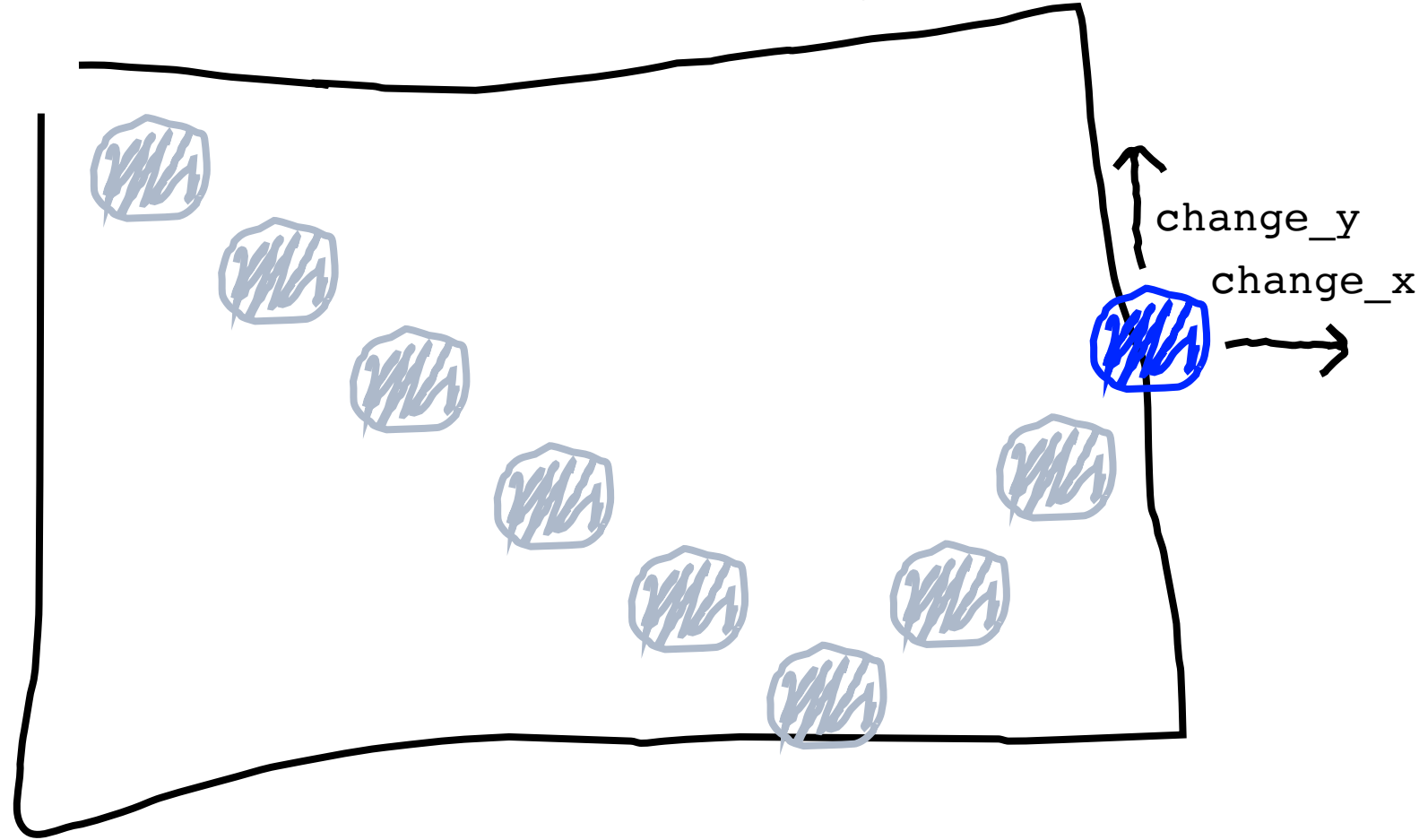
# Bouncing Ball

We want this!



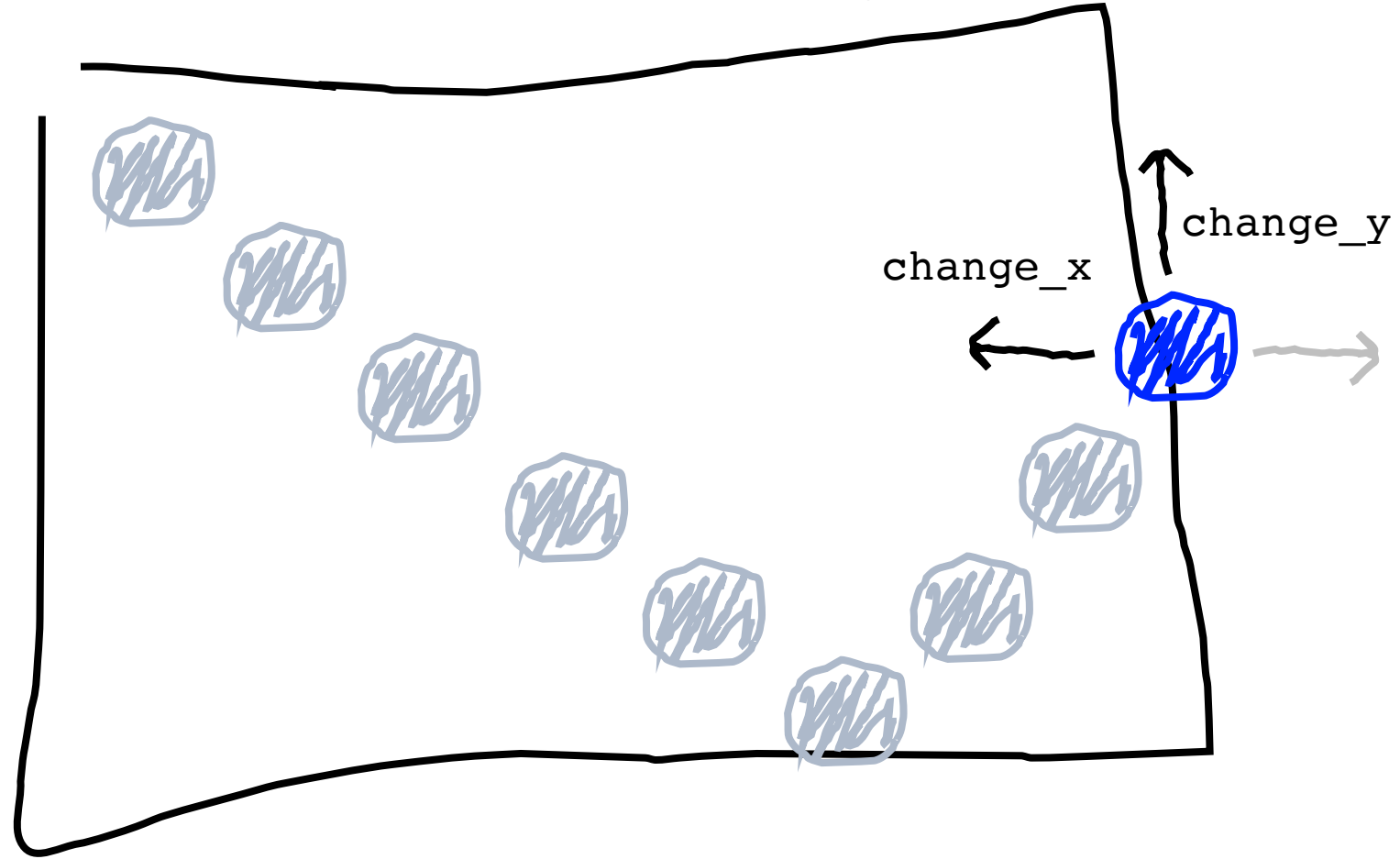
# Bouncing Ball

This was our old velocity



# Bouncing Ball

This is our new velocity

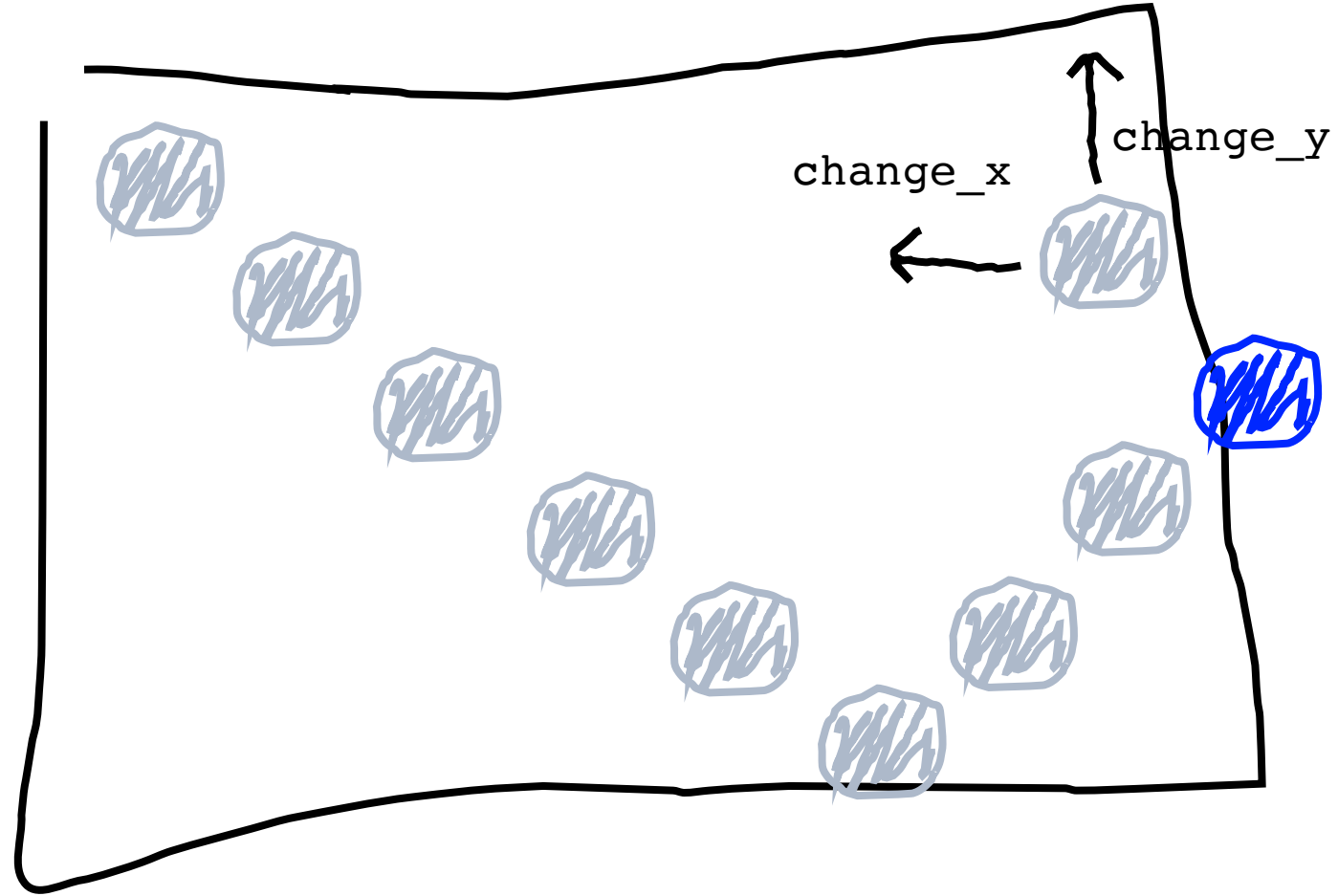


When reflecting horizontally:  $\text{change\_x} = -\text{change\_x}$



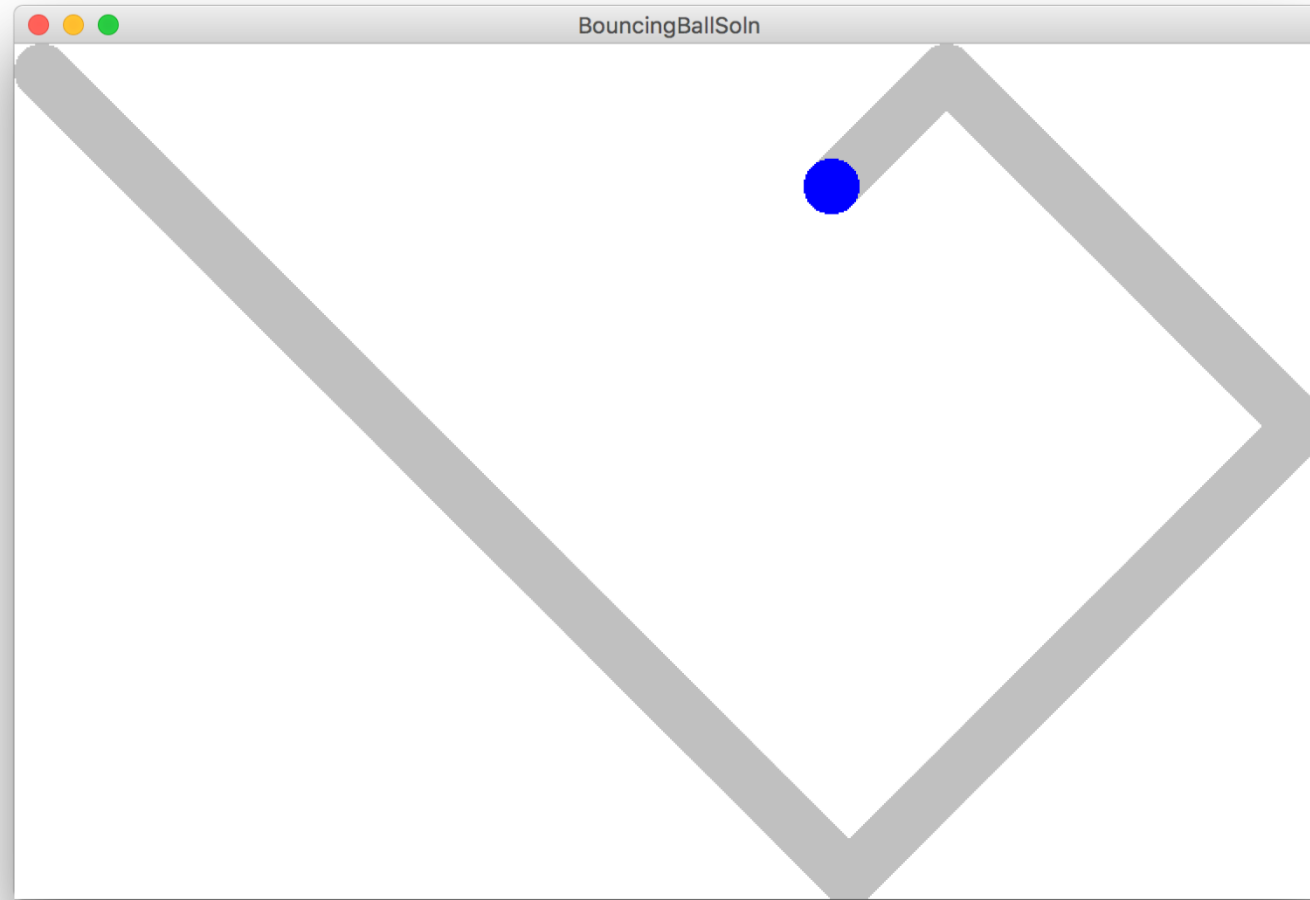
# Bouncing Ball

Tenth heartbeat



When reflecting horizontally:  $\text{change\_x} = -\text{change\_x}$

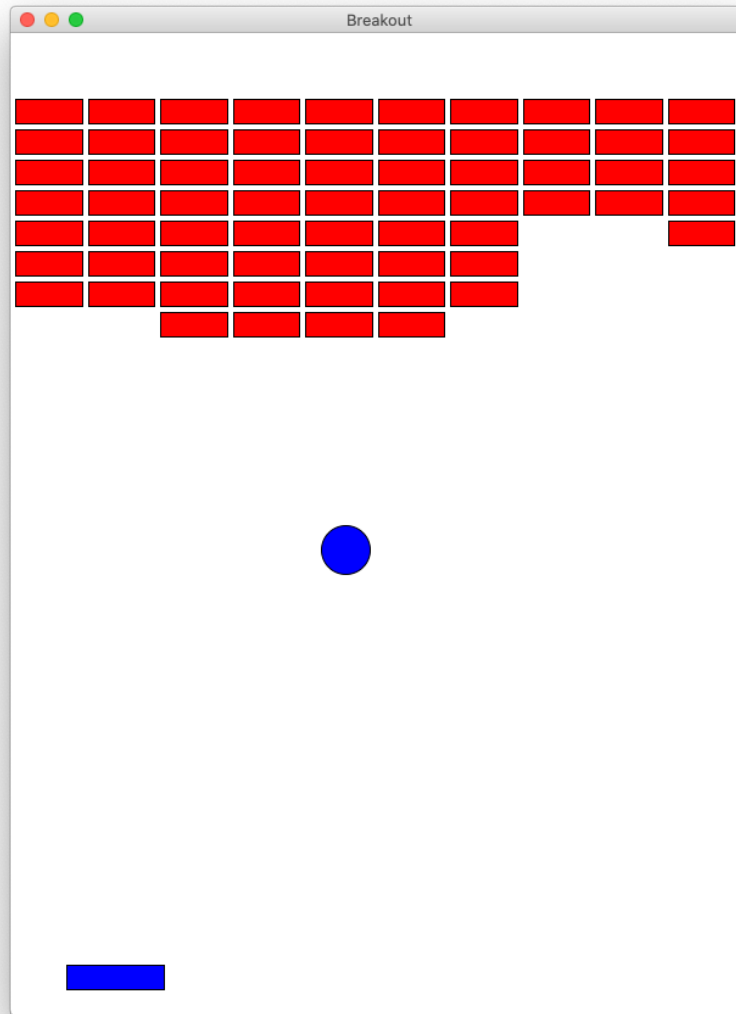
# Bouncing Ball



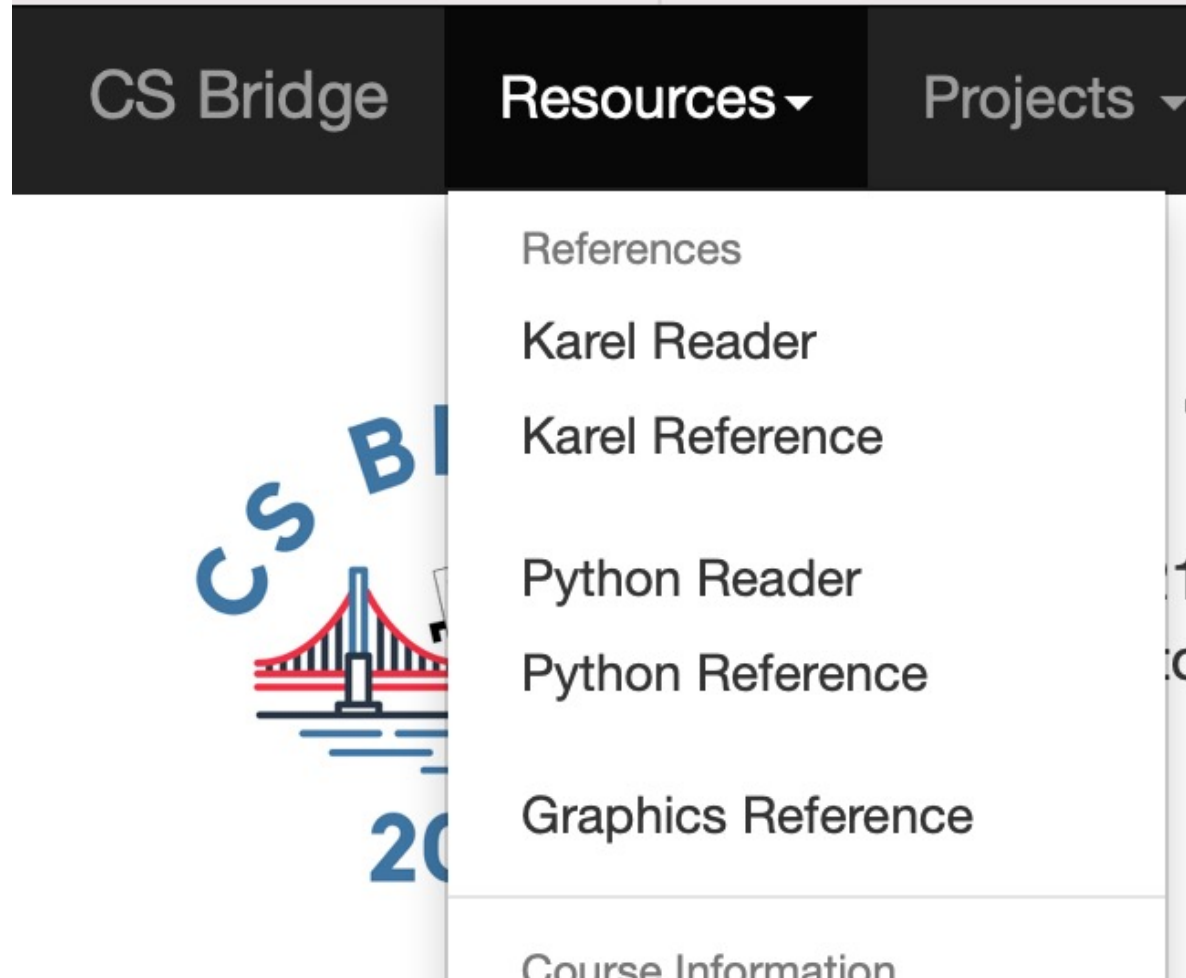
# Lecture Plan

- **Review:** Graphics
- Animation Loop Structure
- **Example:** Move To Center
- **Practice:** Bouncing Ball
- Passing Parameters

# Coming soon...

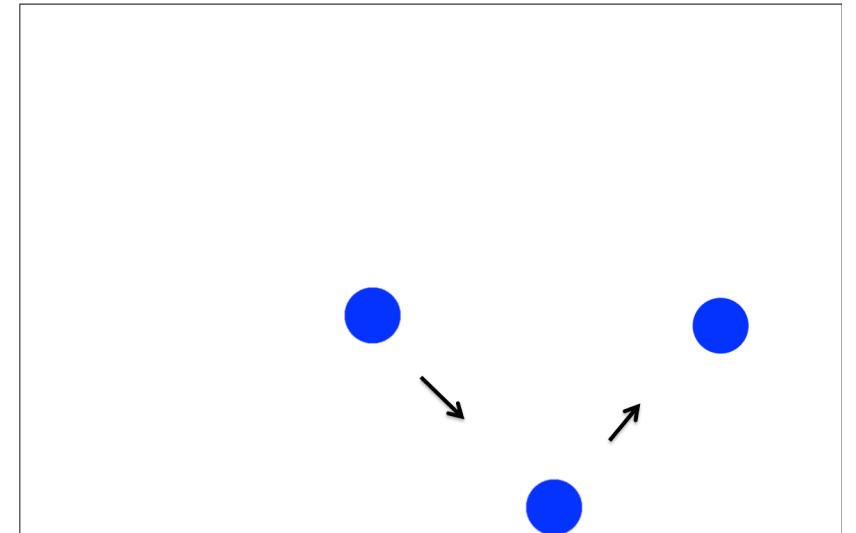
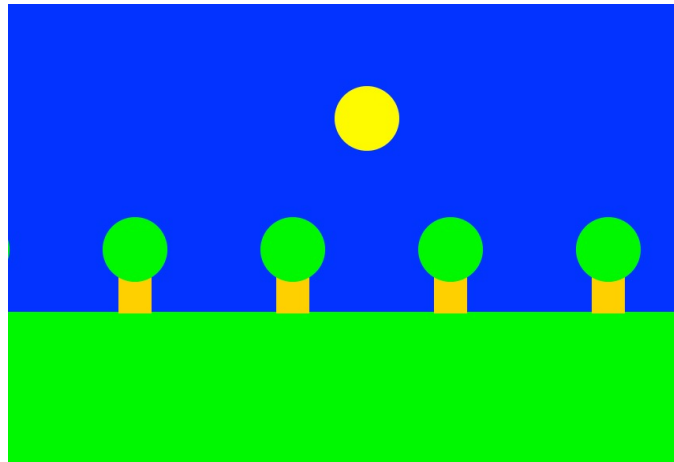
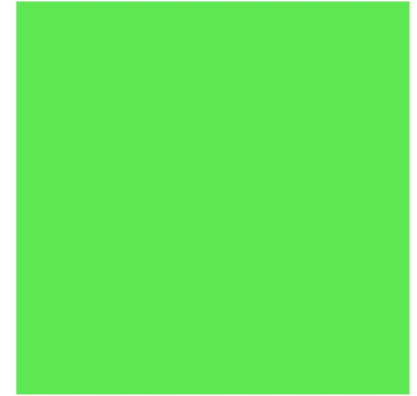


# Graphics Resources



# Rest Of Today

- **Quickstart:** Program a mystery square... (???)
- **Section:** Complete the implementation of the bouncing ball program
- **Project:** Use animation to create your own short film!



# What's Next?

- Time for your section's quickstart time!
- Check your section's Ed group for more information

# Extra Slides



# Wait a minute....

```
def make_ball(canvas):
```

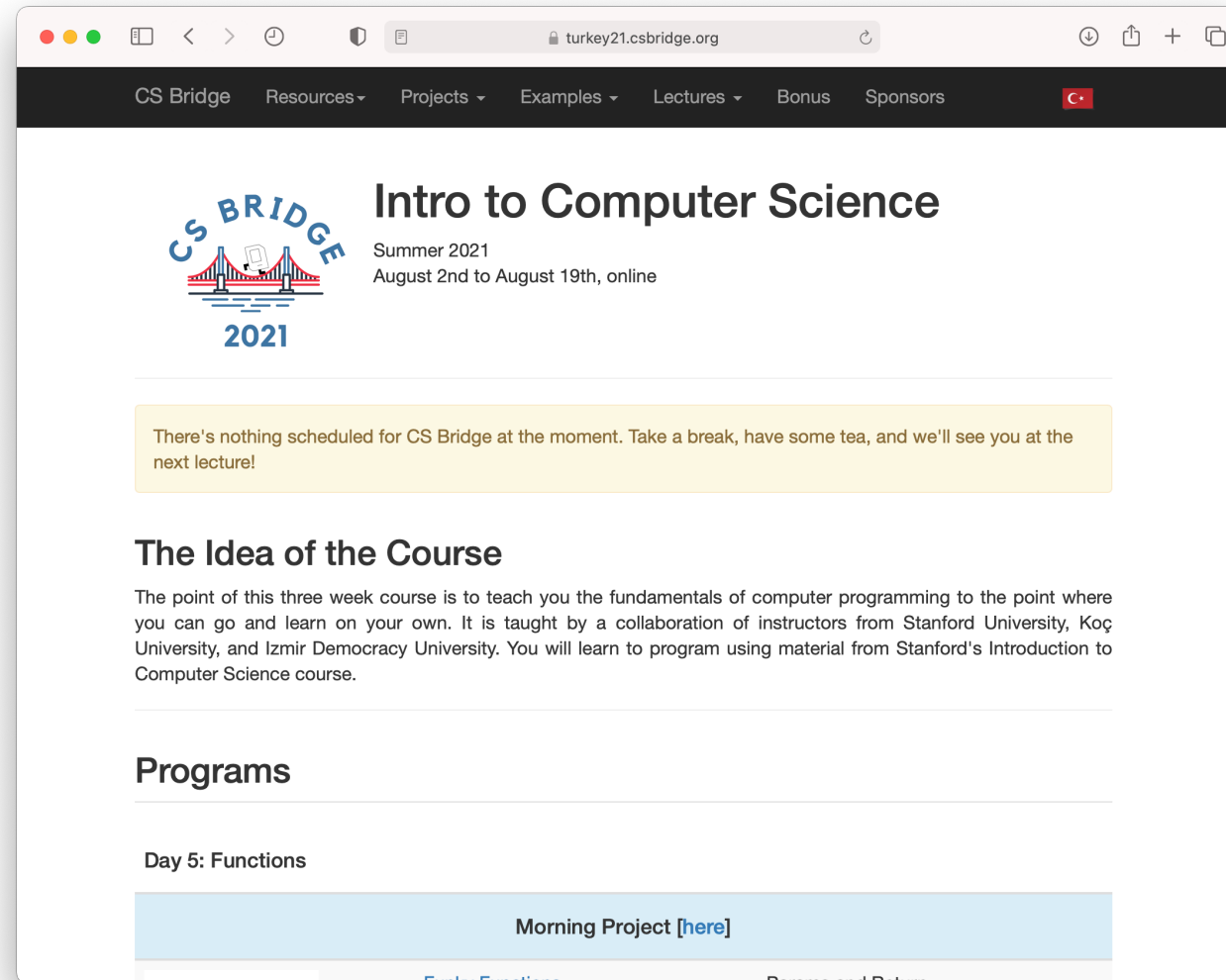
*Does this copy the canvas??!!*

*Variables are stored using a reference.  
Which is like a **URL**. The URL gets copied  
when you pass the variable*

# Lecture Plan

- Review: Graphics
- Animation Loop Structure
- Example: Move To Center
- Practice: Bouncing Ball
- Passing Parameters

# How do we share websites?



[turkey21.csbridge.org](https://turkey21.csbridge.org)

# Passing Parameters

```
def main():  
    canvas = Canvas()  
    make_ball(canvas)  
  
def make_ball(canvas):  
    canvas.create_oval( ... )
```

---

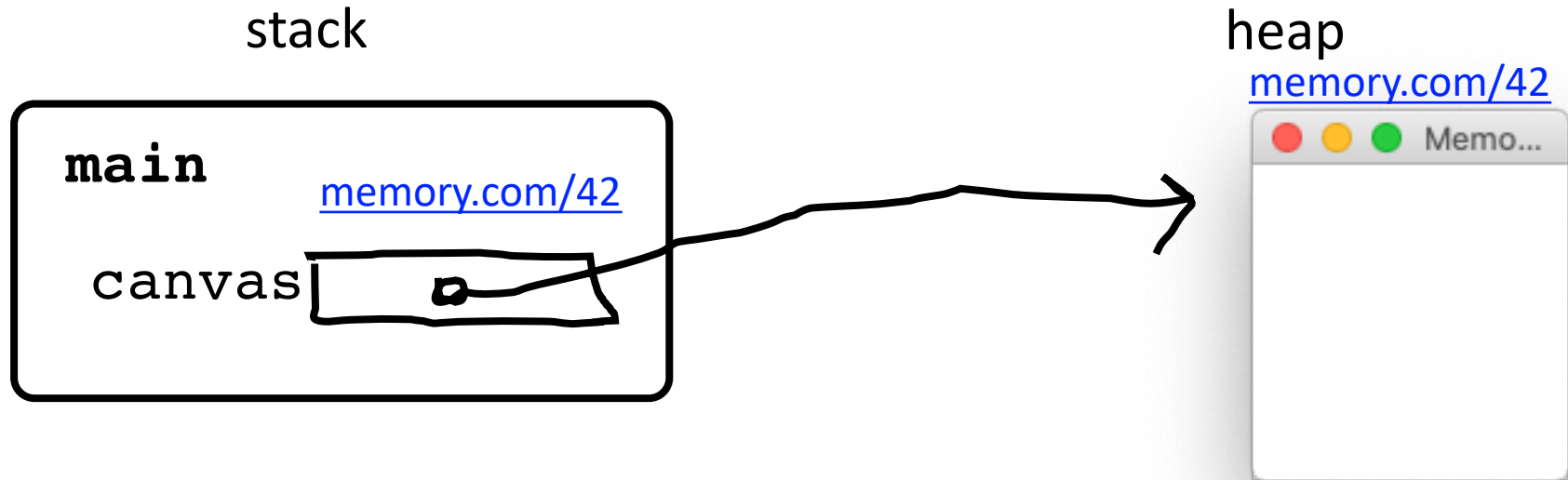
stack

heap

**main**

# Passing Parameters

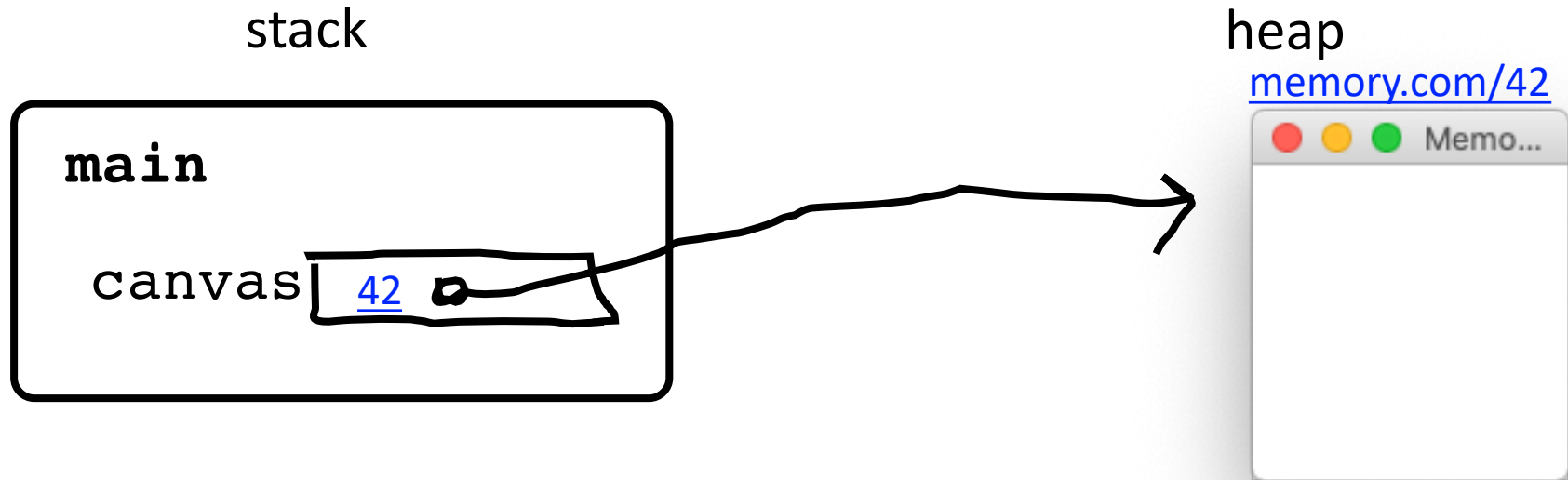
```
def main():  
    canvas = Canvas()  
    make_ball(canvas)  
  
def make_ball(canvas):  
    canvas.create_oval( ... )
```



# Passing Parameters

```
def main():  
    canvas = Canvas()  
    make_ball(canvas)
```

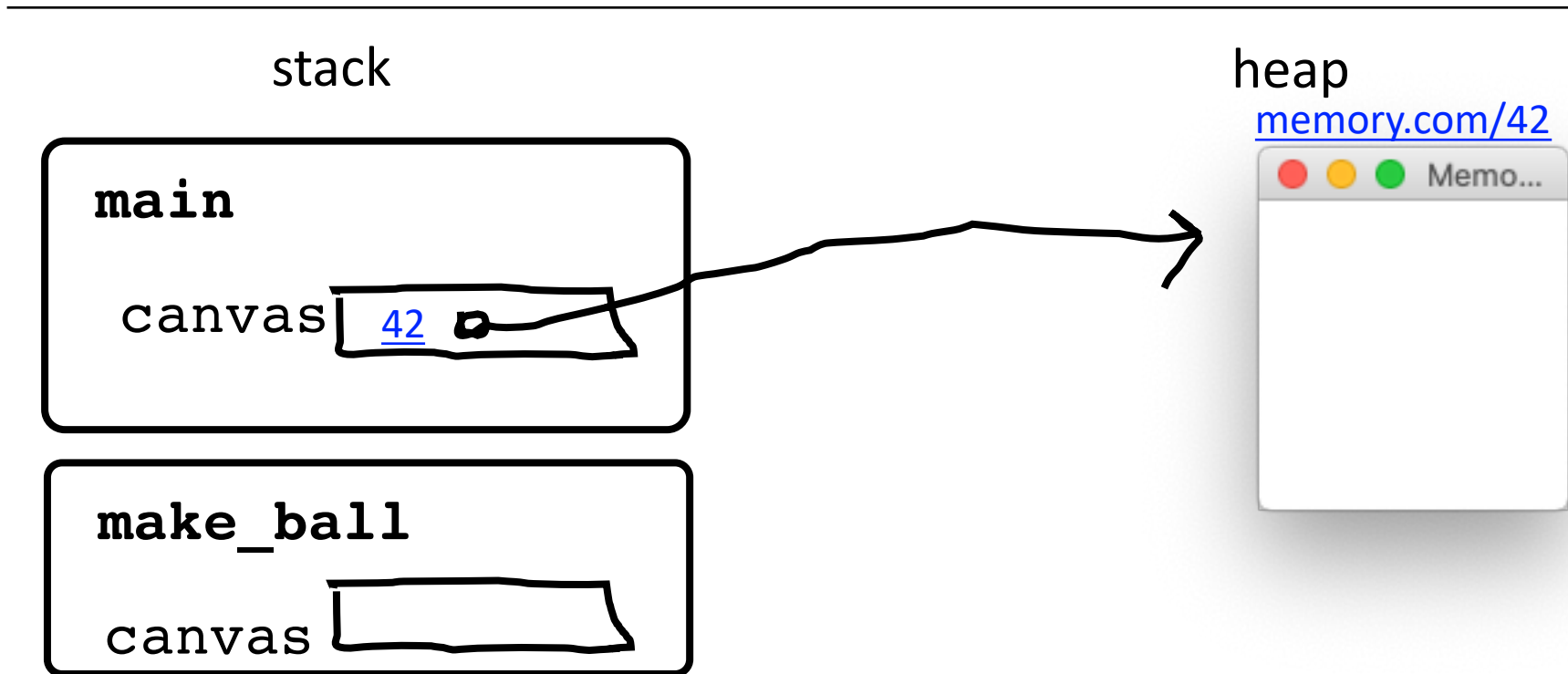
```
def make_ball(canvas):  
    canvas.create_oval( ... )
```



# Passing Parameters

```
def main():  
    canvas = Canvas()  
    make_ball(canvas)
```

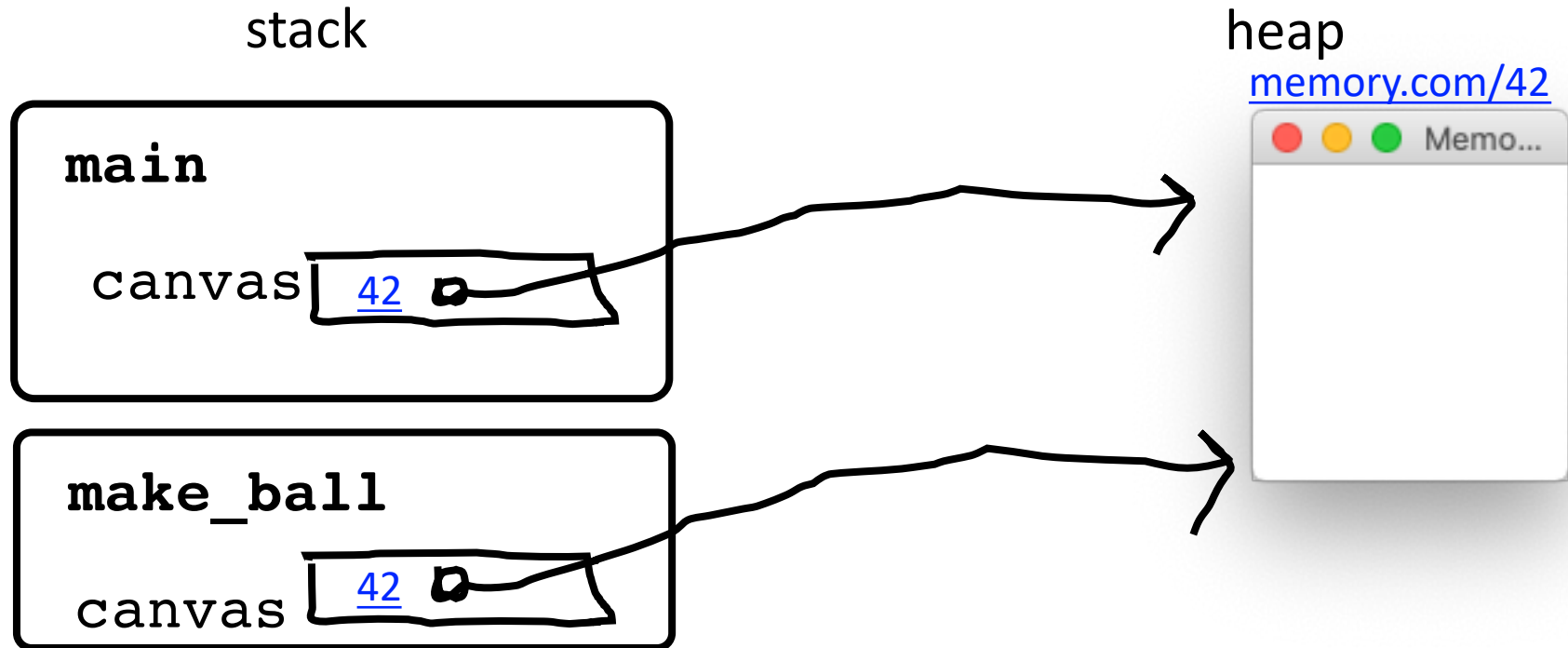
```
def make_ball(canvas):  
    canvas.create_oval( ... )
```



# Passing Parameters

```
def main():  
    canvas = Canvas()  
    make_ball(canvas)
```

```
def make_ball(canvas):  
    canvas.create_oval( ... )
```

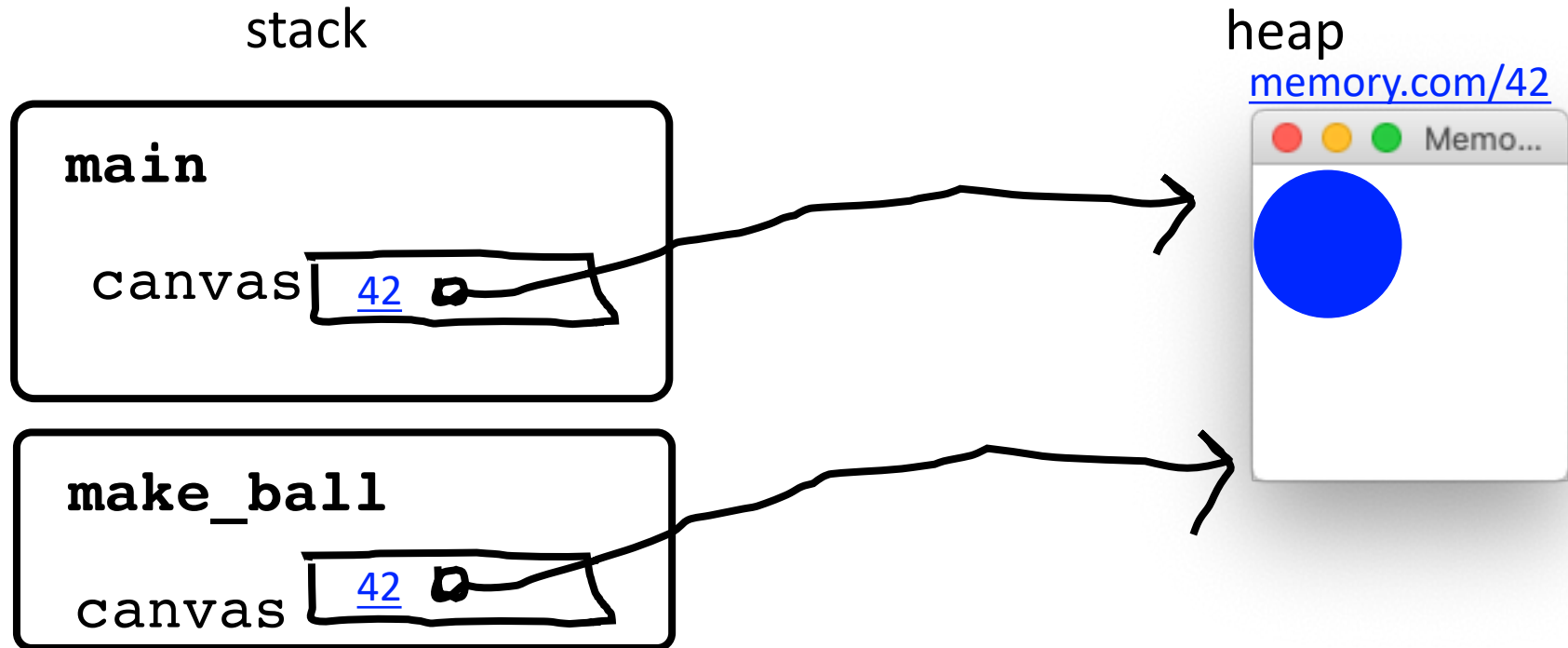




# Passing Parameters

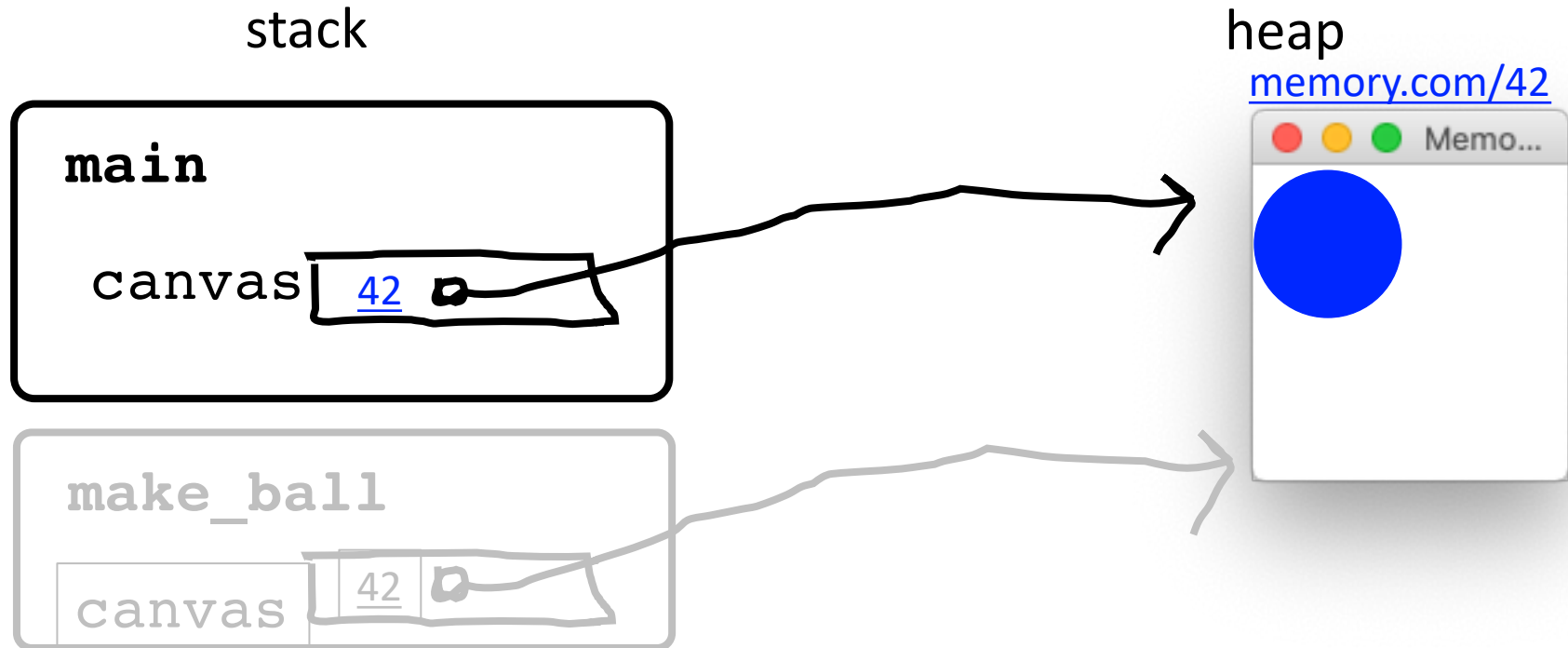
```
def main():  
    canvas = Canvas()  
    make_ball(canvas)
```

```
def make_ball(canvas):  
    canvas.create_oval( ... )
```



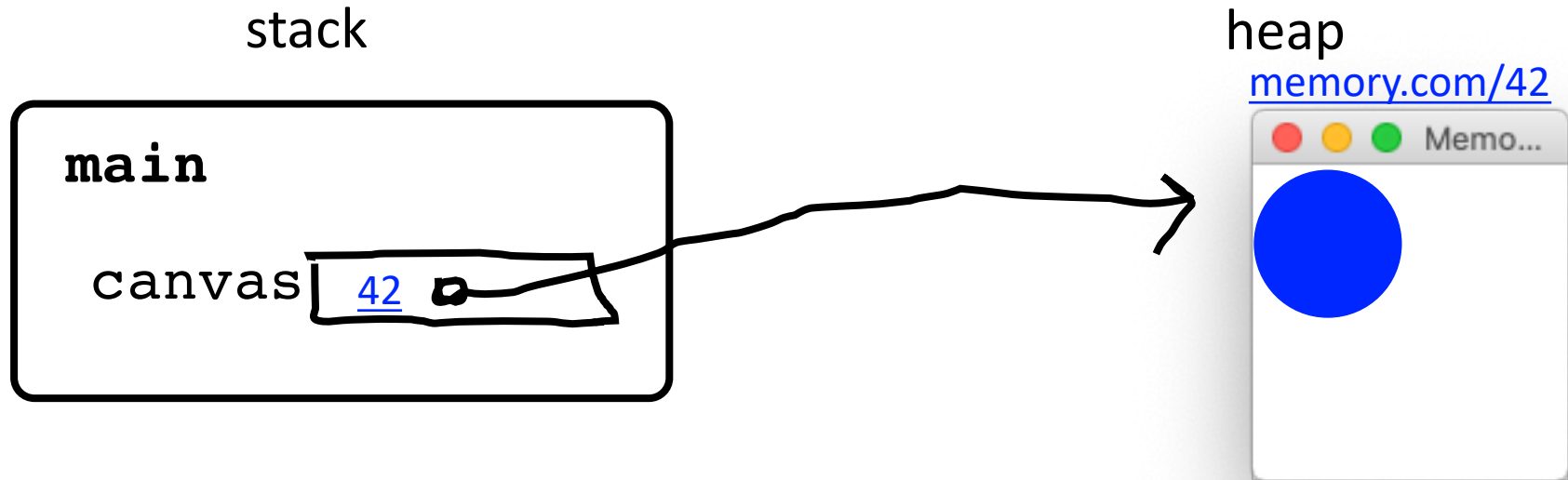
# Passing Parameters

```
def main():  
    canvas = Canvas()  
    make_ball(canvas)  
  
def make_ball(canvas):  
    canvas.create_oval( ... )
```



# Passing Parameters

```
def main():  
    canvas = Canvas()  
    make_ball(canvas)  
  
def make_ball(canvas):  
    canvas.create_oval( ... )
```



# Key Idea: Passing Parameters



When passing variables,  
some act just like you are  
passing a URL.

That allows functions to  
modify the variable