



Animation

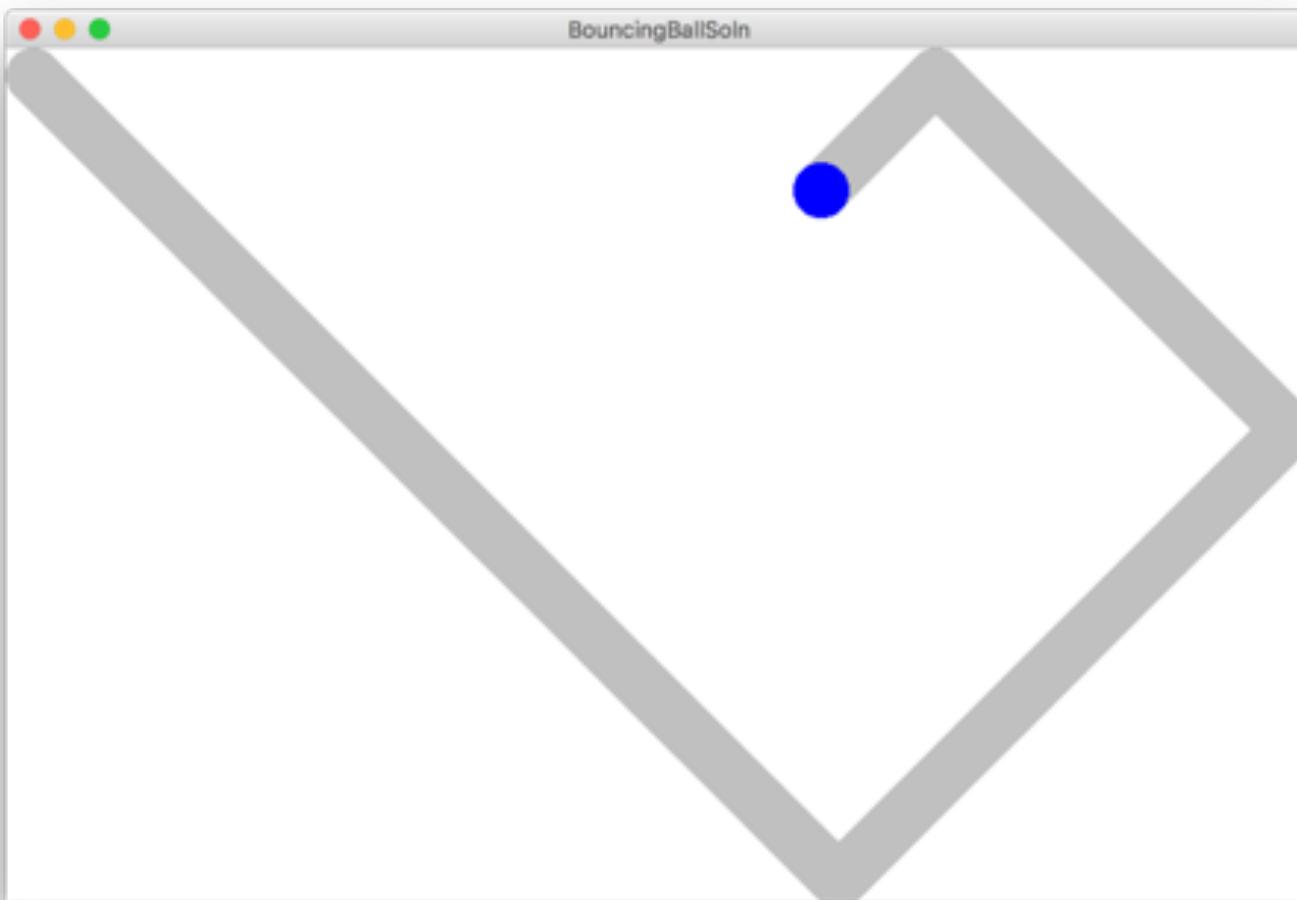
**Chris Piech + Mehran Sahami
CS106A, Stanford University**

Learning Goals

1. Feel more confident debugging
2. Write animated programs



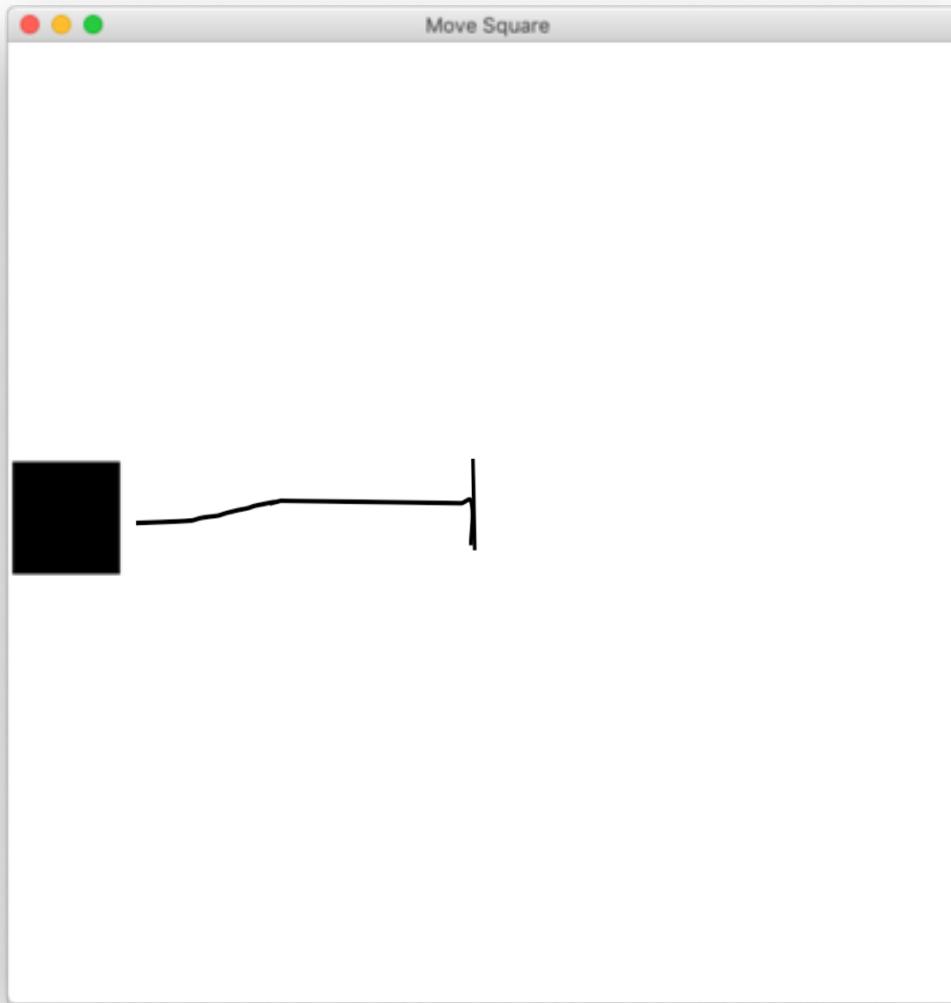
You will be able to write Bouncing Ball



Great foundation



Move to Center



In our last episode...

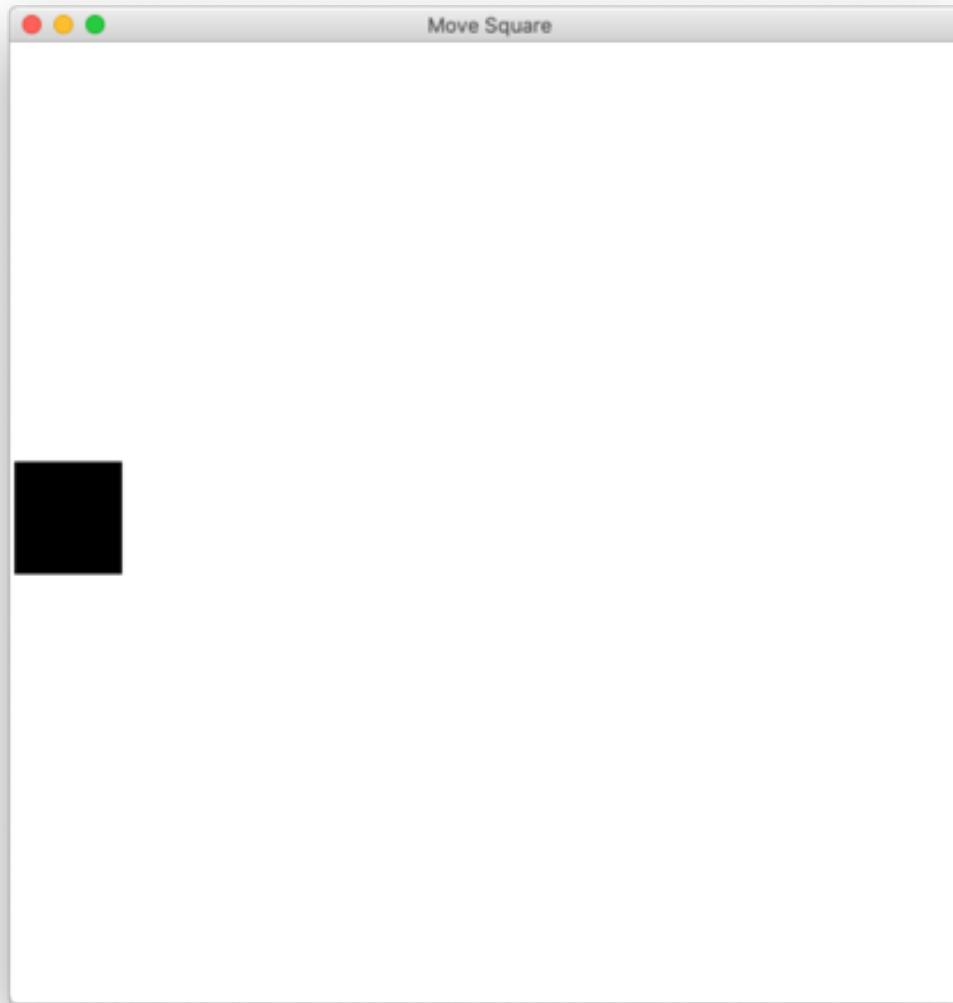
Graphics from tkinter (aka tk)

```
import tkinter
```

```
# we write this for you, and include it
# in all of your projects!
def make_canvas(width, height, title):
```



Add square



Graphics from tkinter (aka tk)

```
import tkinter
```

```
# we write this for you, and include it
# in all of your projects!
def make_canvas(width, height, title):
```



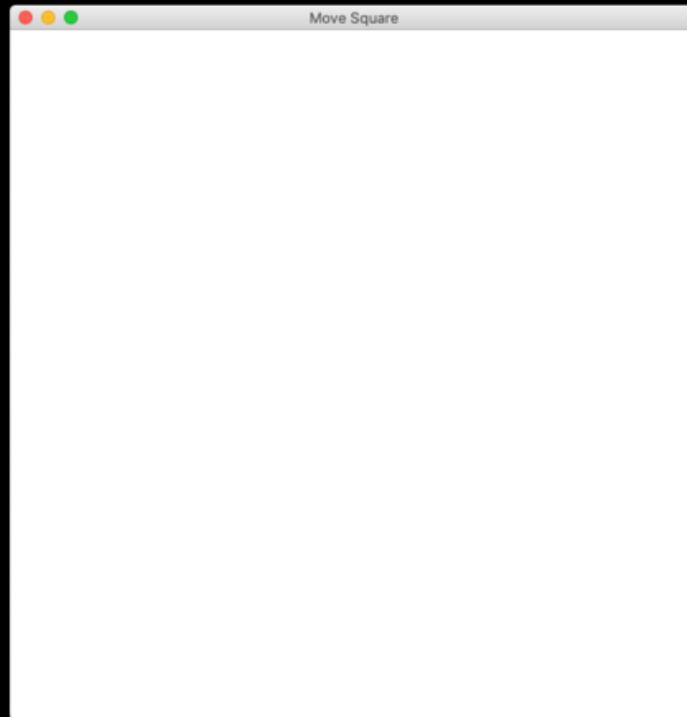
```
def main():
    canvas = make_canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



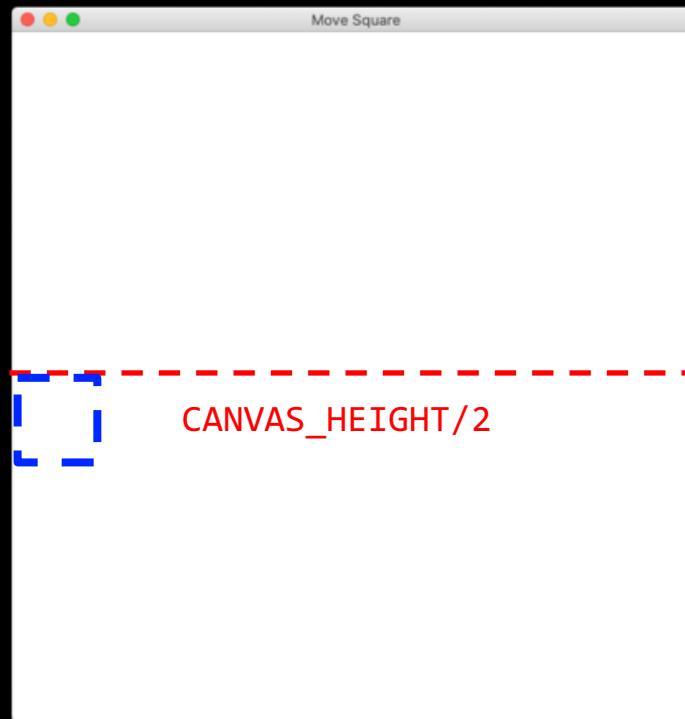
```
def main():
    canvas = make_canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



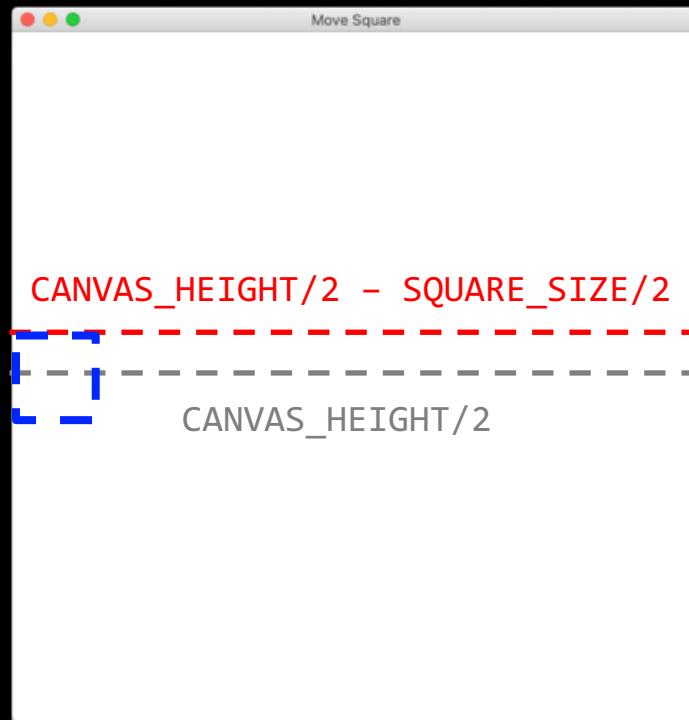
```
def main():
    canvas = make_canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



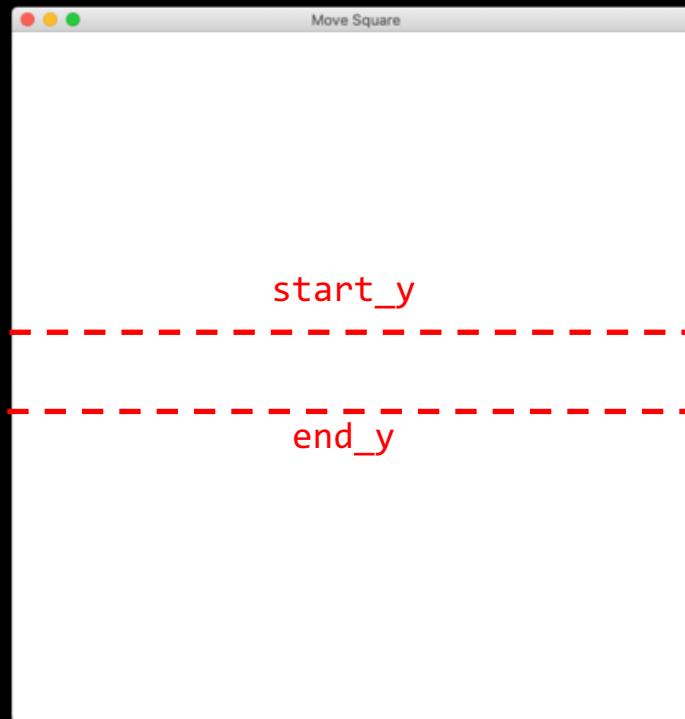
```
def main():
    canvas = make_canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



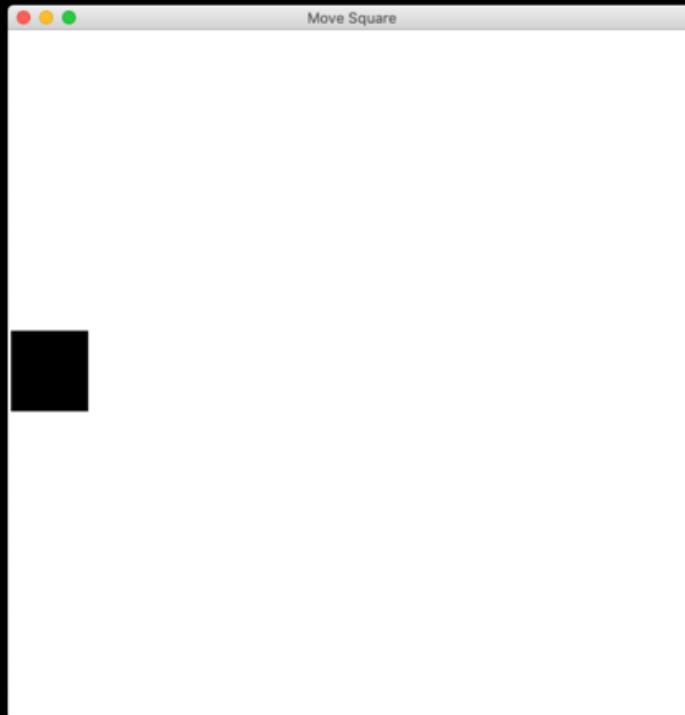
```
def main():
    canvas = make_canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



```
def main():
    canvas = make_canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



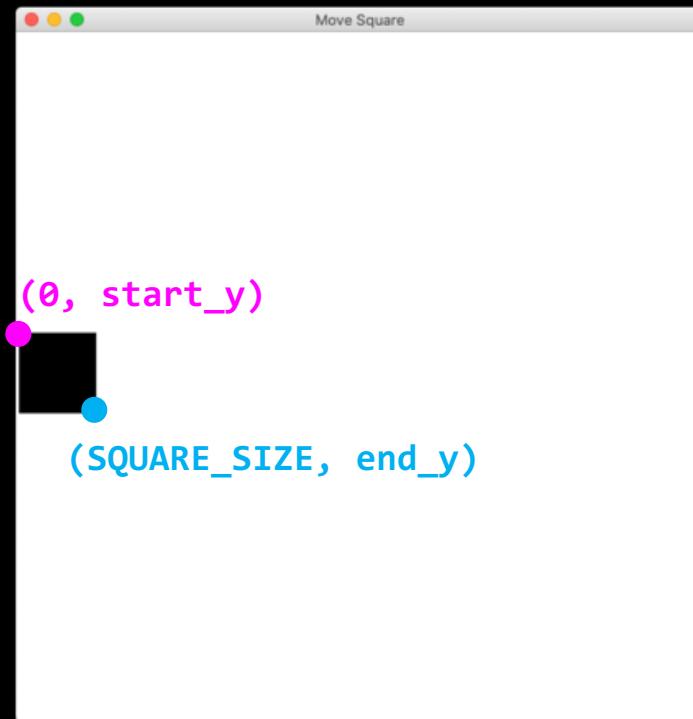
```
def main():
    canvas = make_canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



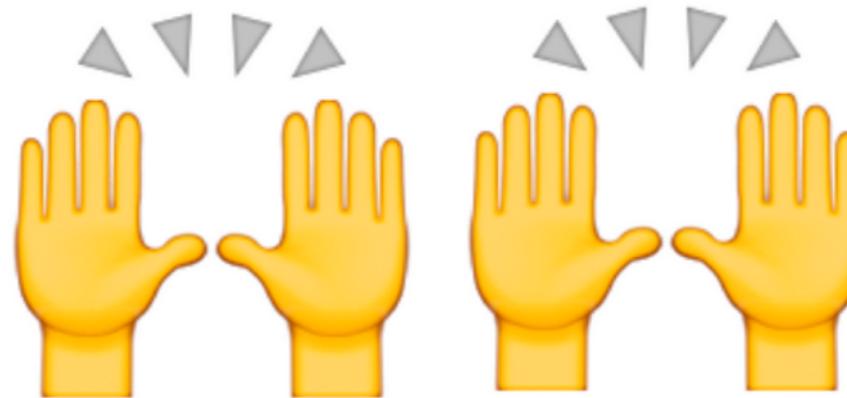
Some “heavy duty” variables allow you to call functions on them



```
def main():
    canvas = make_canvas(CANVAS_WIDTH, CANVAS_HEIGHT, 'Move Square')
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y, fill='black')
    canvas.mainloop()
```



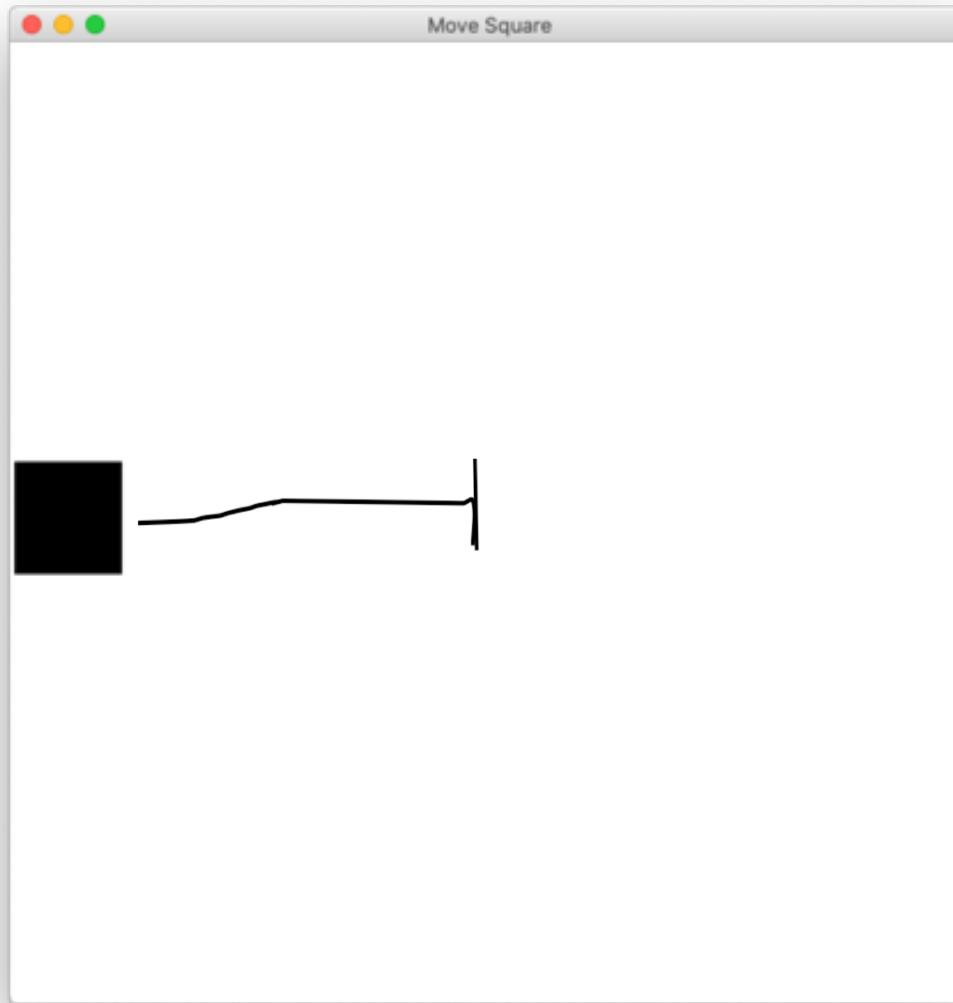
You're now all graphics programmers!



End review...

How do movies or games
animate?

Move to Center



* That's not quite toy story, but it is a start...



Animation Loop

```
def main():
    # setup

    while True:
        # update world

        # pause
        time.sleep(DELAY)
```



Animation Loop

```
def main():
    # setup

    while True:
        # update world

        # pause
        time.sleep(DELAY)
```

Make all the variables you need.



Animation Loop

```
def main():
    # setup
    while True:
        # update world

        # pause
        time.sleep(DELAY)
```

The animation loop is a repetition of heartbeats



Animation Loop

```
def main():
    # setup

    while True:
        # update world
        # pause
        time.sleep(DELAY)
```

Each heart-beat, update
the world forward one
frame



Animation Loop

```
def main():
    # setup

    while True:
        # update world

        # pause
        time.sleep(DELAY)
```

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



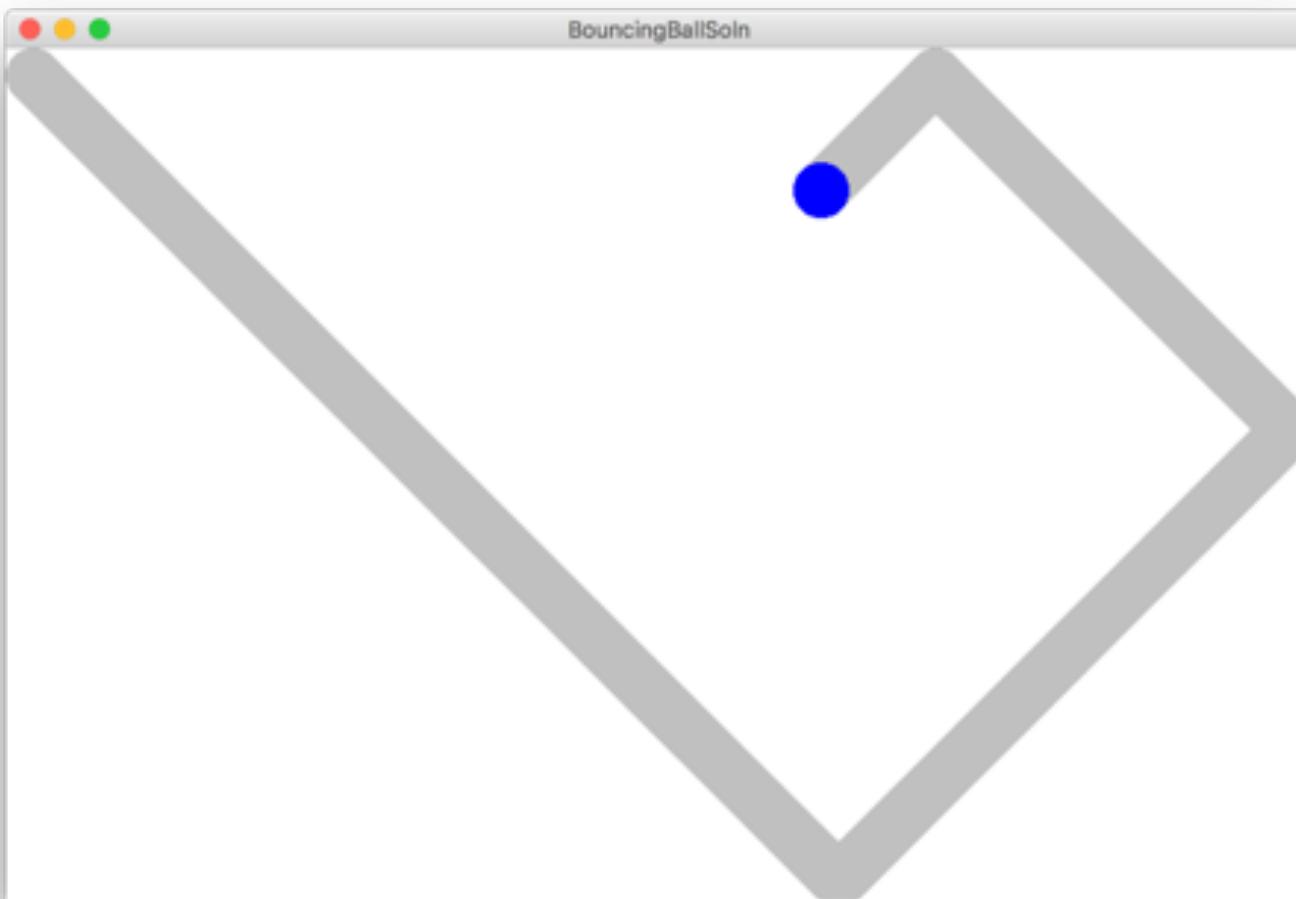
Move To Center

```
def main():
    # setup
    canvas = make_canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    r = canvas.create_rectangle(0, 0, 100, 100)
    while not is_past_center(canvas, r):
        # update world
        canvas.move(r, 1, 0)
        canvas.update()
        # pause
        time.sleep(DELAY)
```

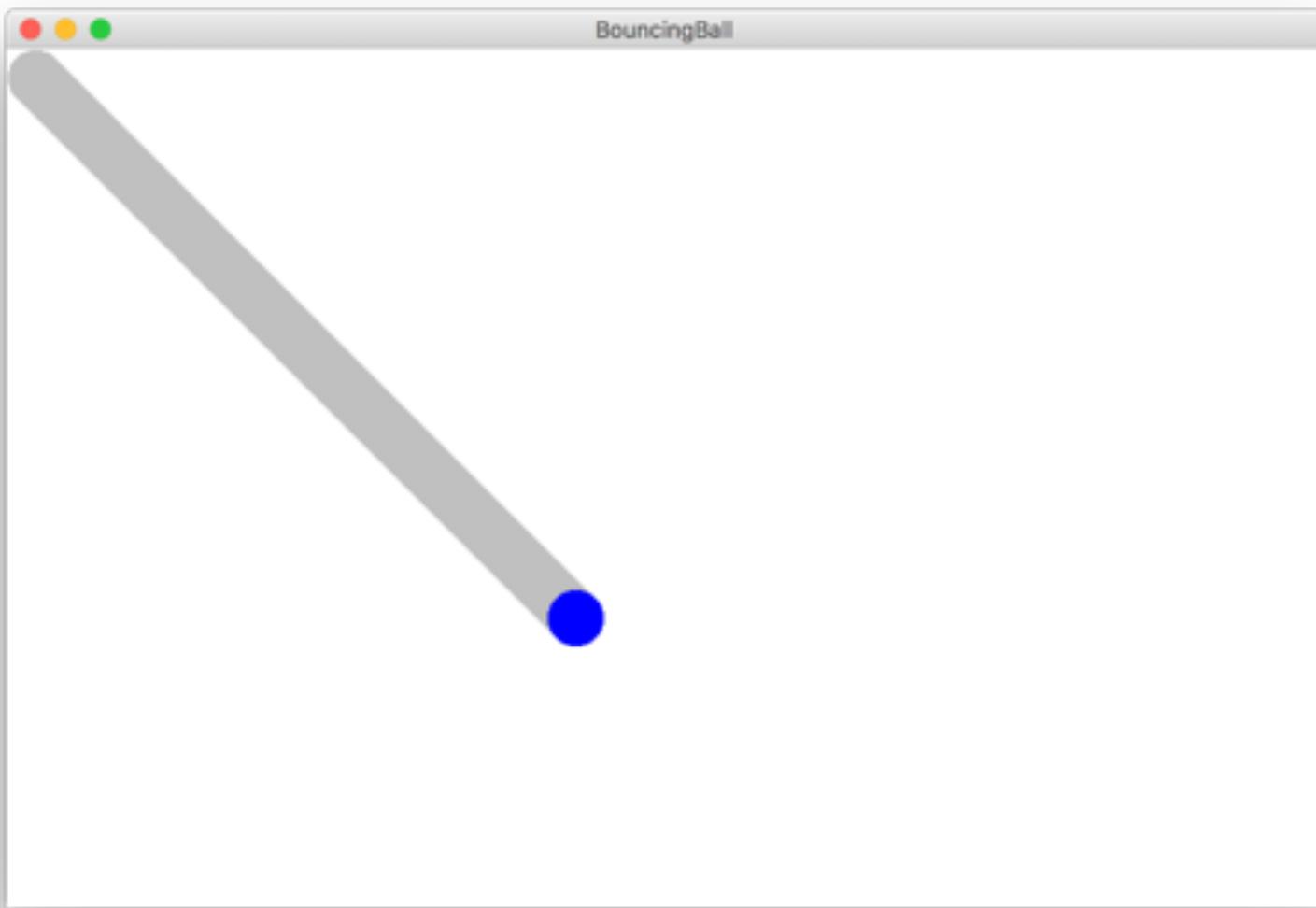


We are ready...

Bouncing Ball

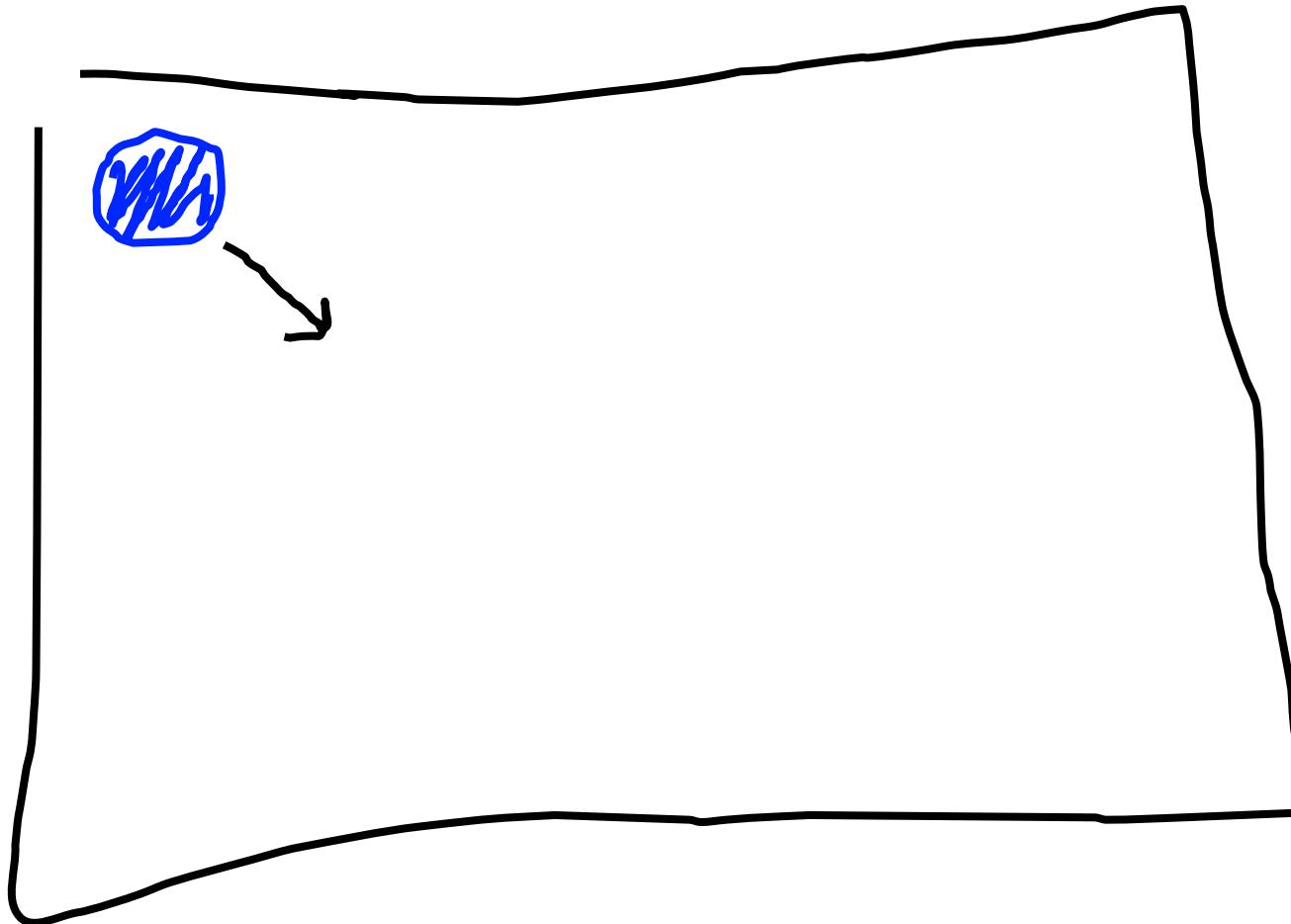


Milestone #1



Bouncing Ball

First heartbeat

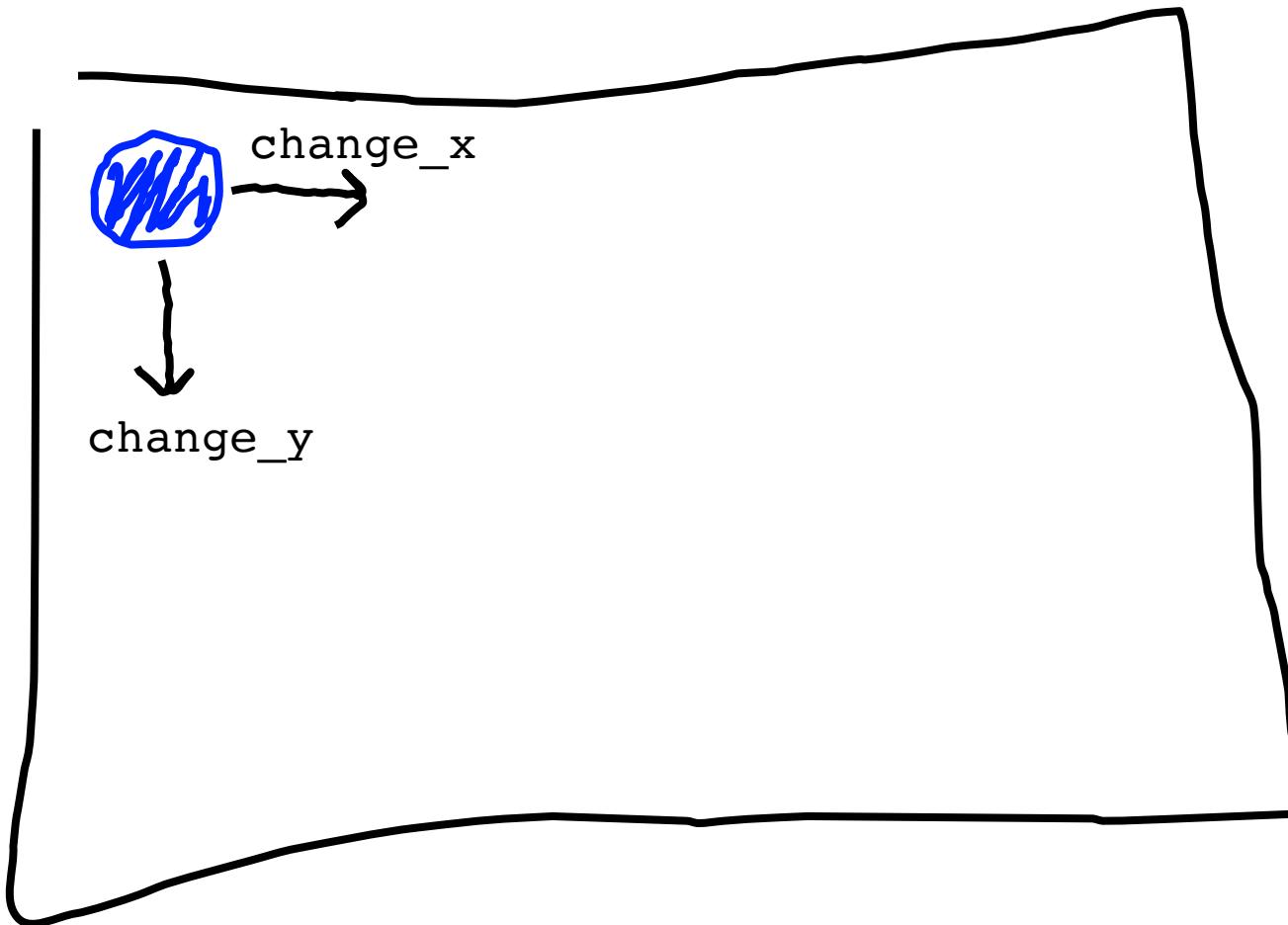


Key variable: how much the ball position
change each heartbeat?



Bouncing Ball

First heartbeat



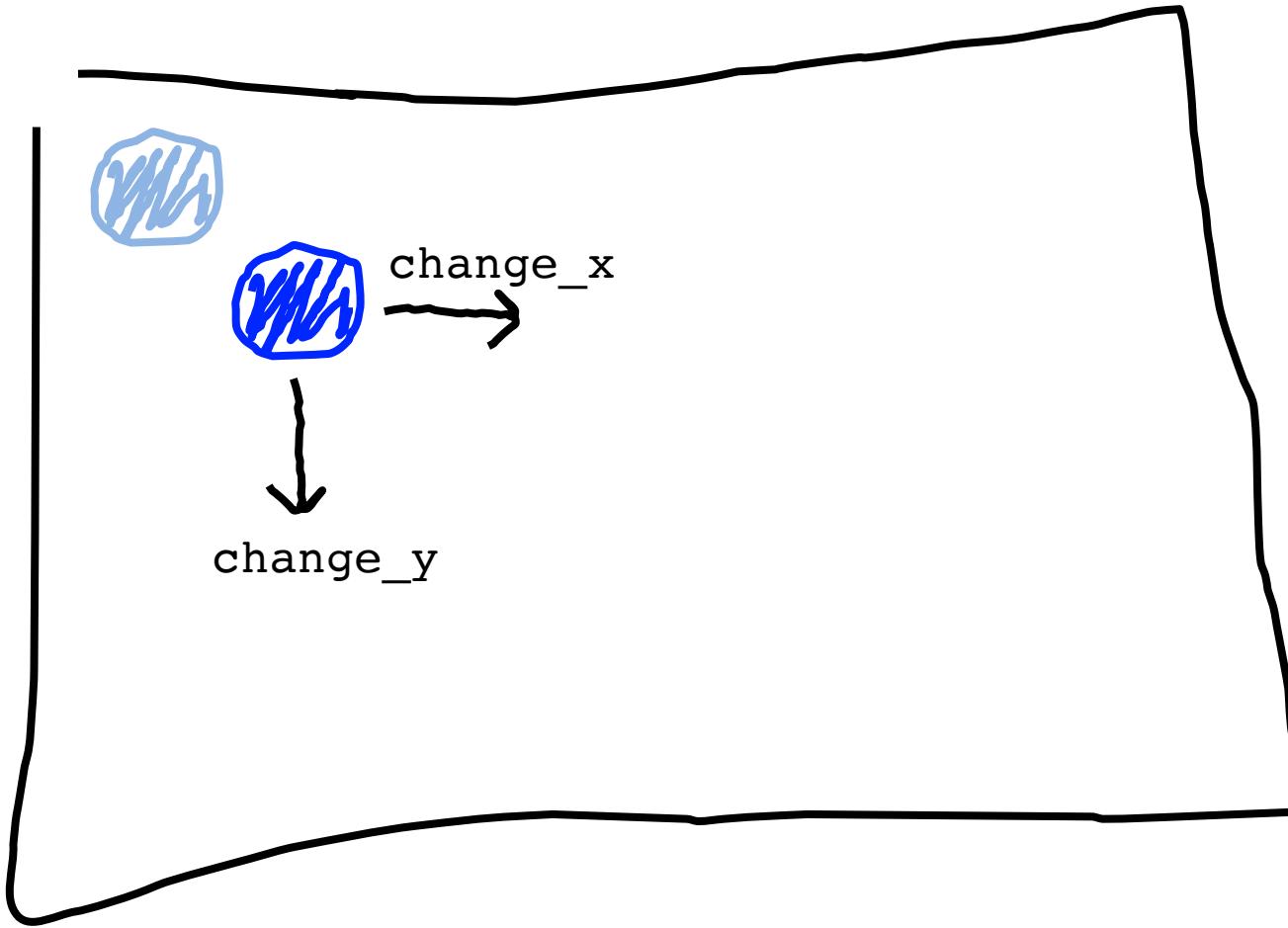
The **move** function takes in
a change in x and a change in y

Pi



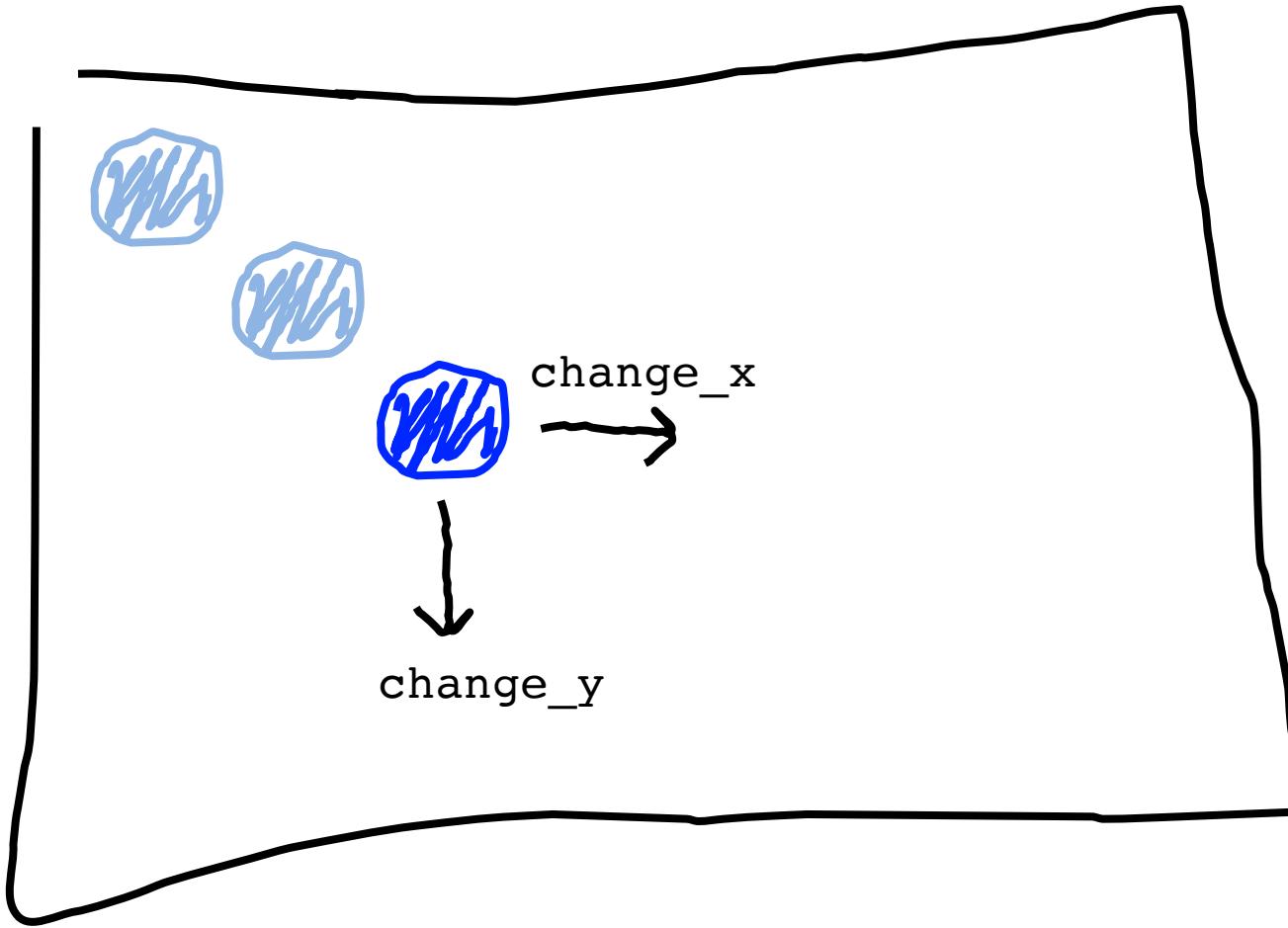
Bouncing Ball

Second heartbeat



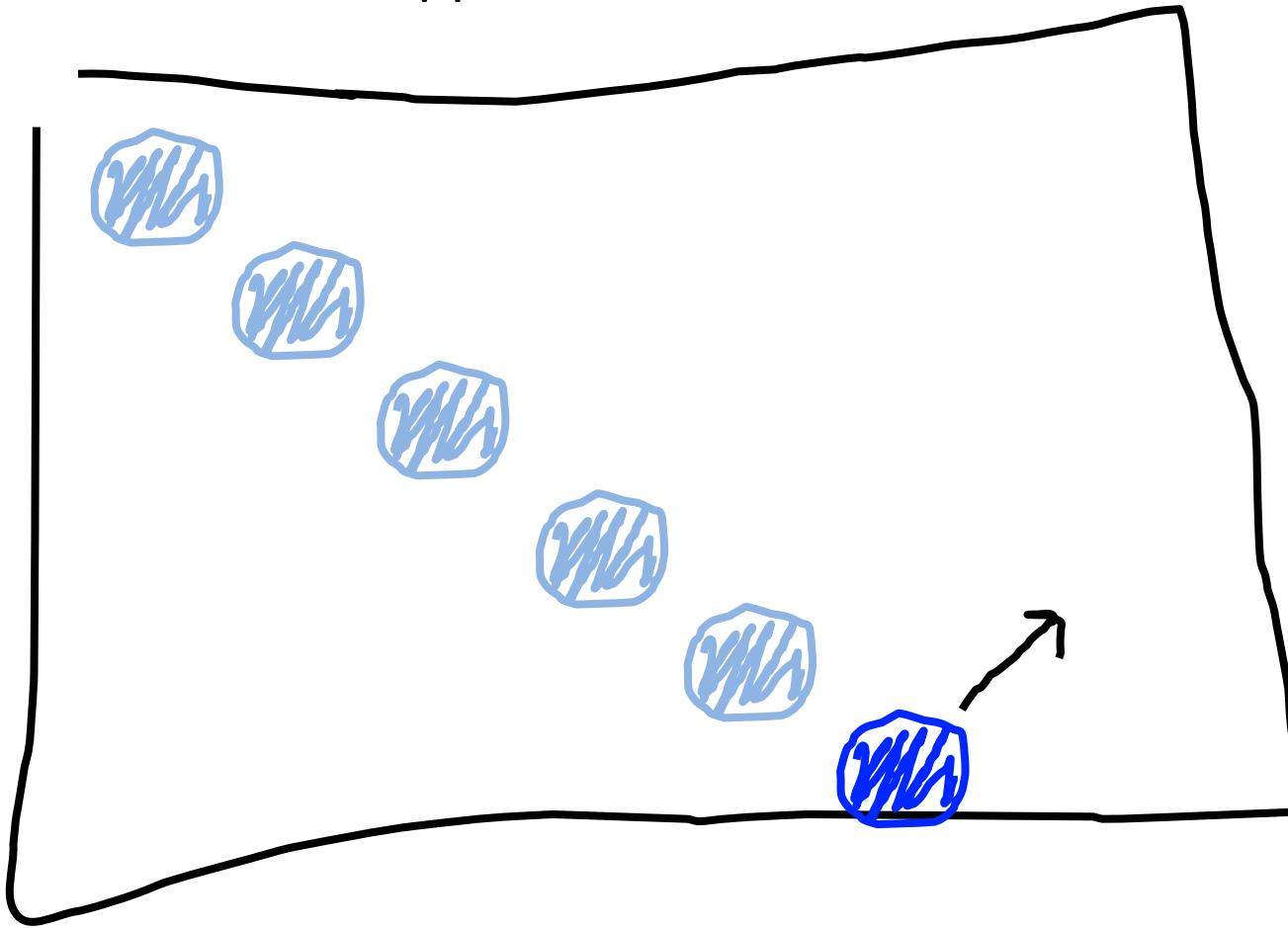
Bouncing Ball

Third heartbeat



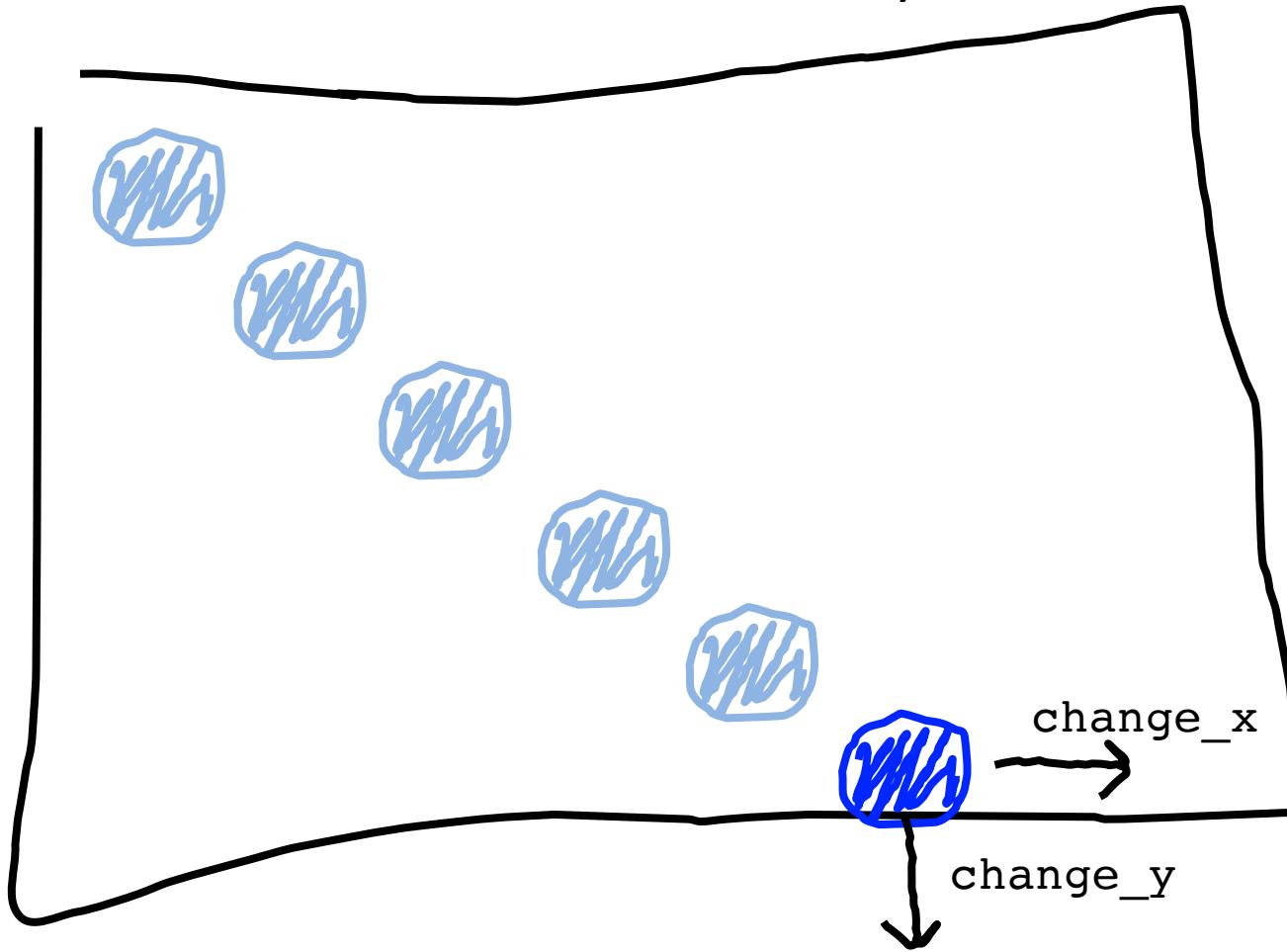
Bouncing Ball

What happens when we hit a wall?



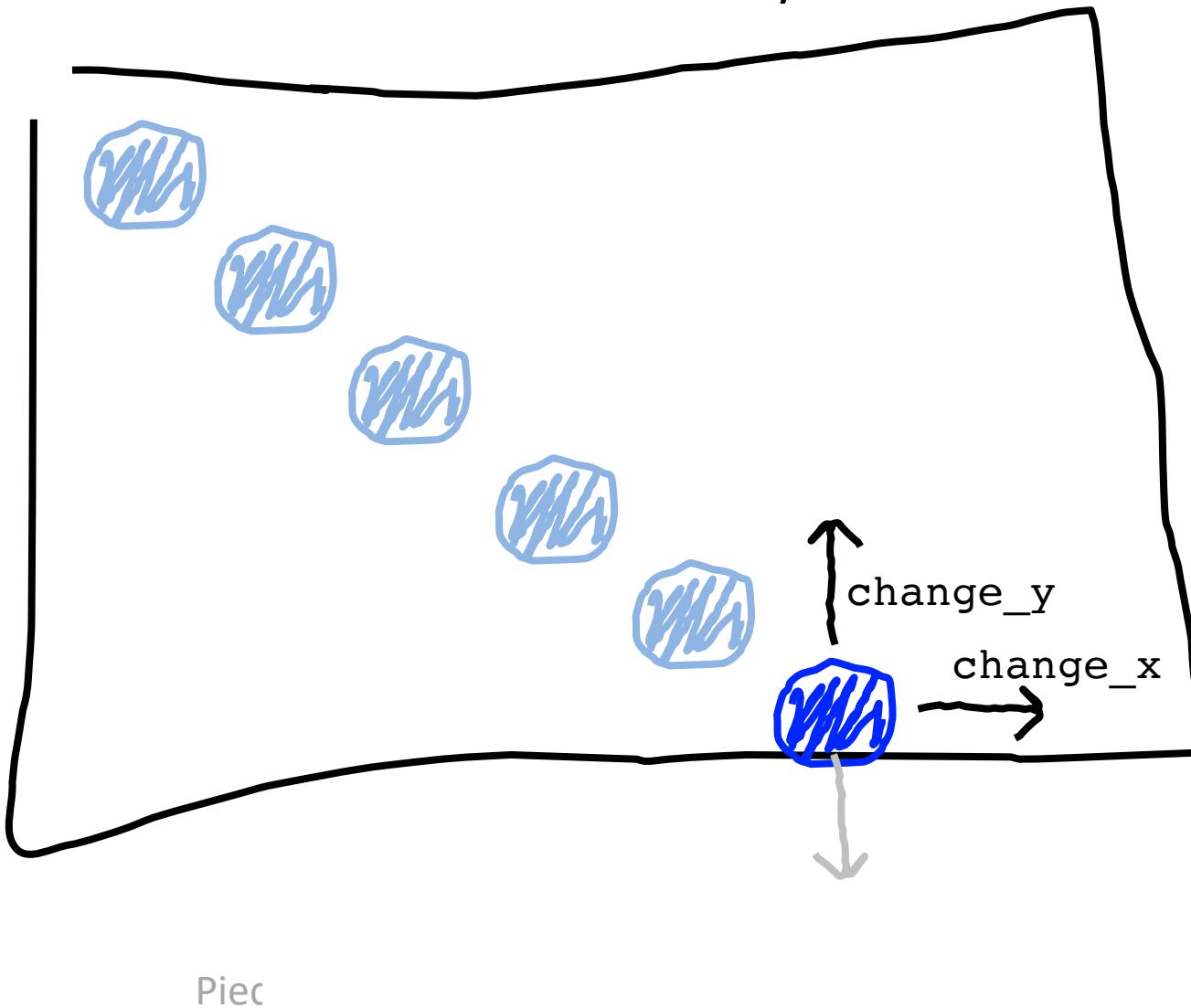
Bouncing Ball

We have this velocity



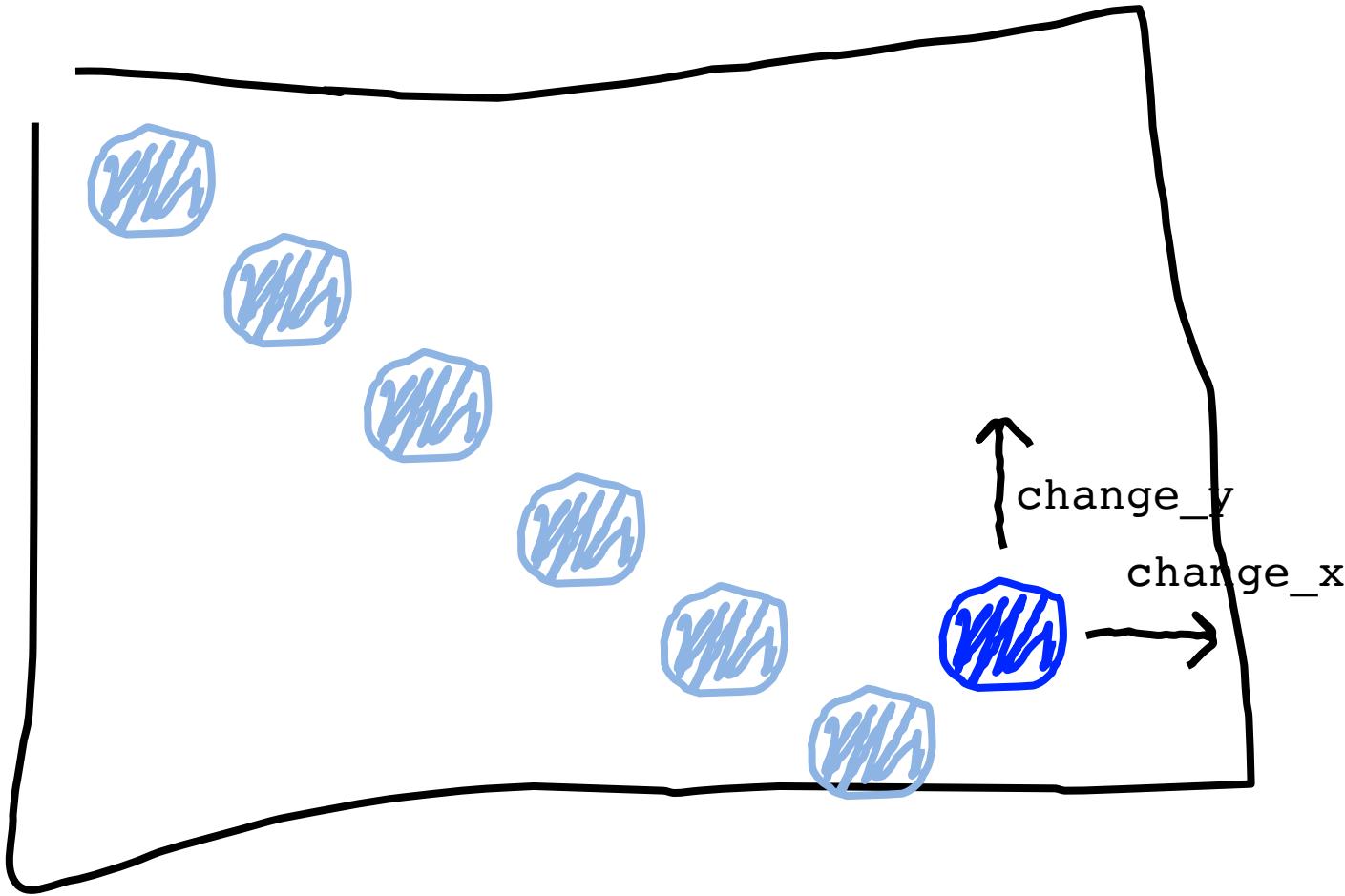
Bouncing Ball

Our new velocity



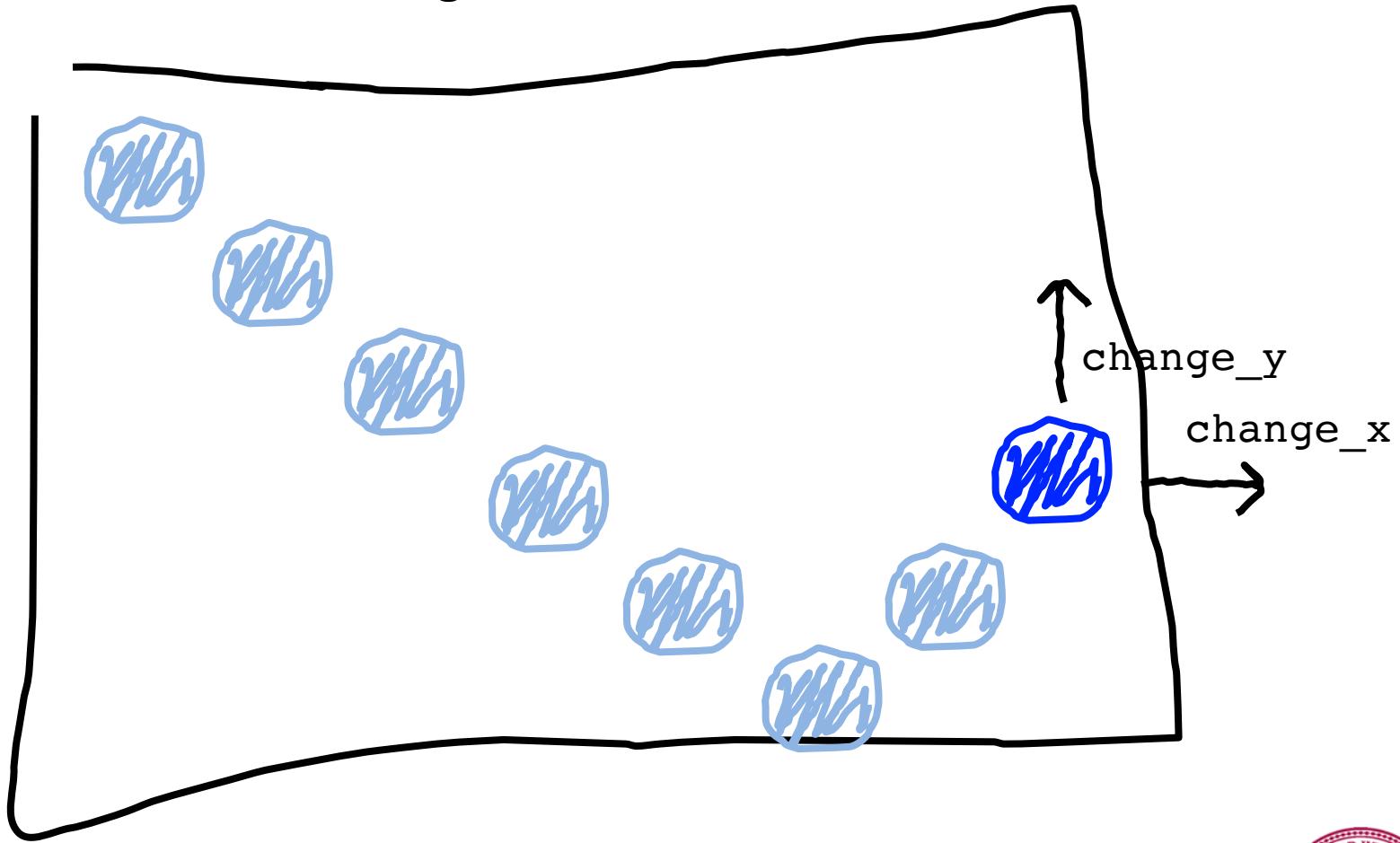
Bouncing Ball

Seventh heartbeat



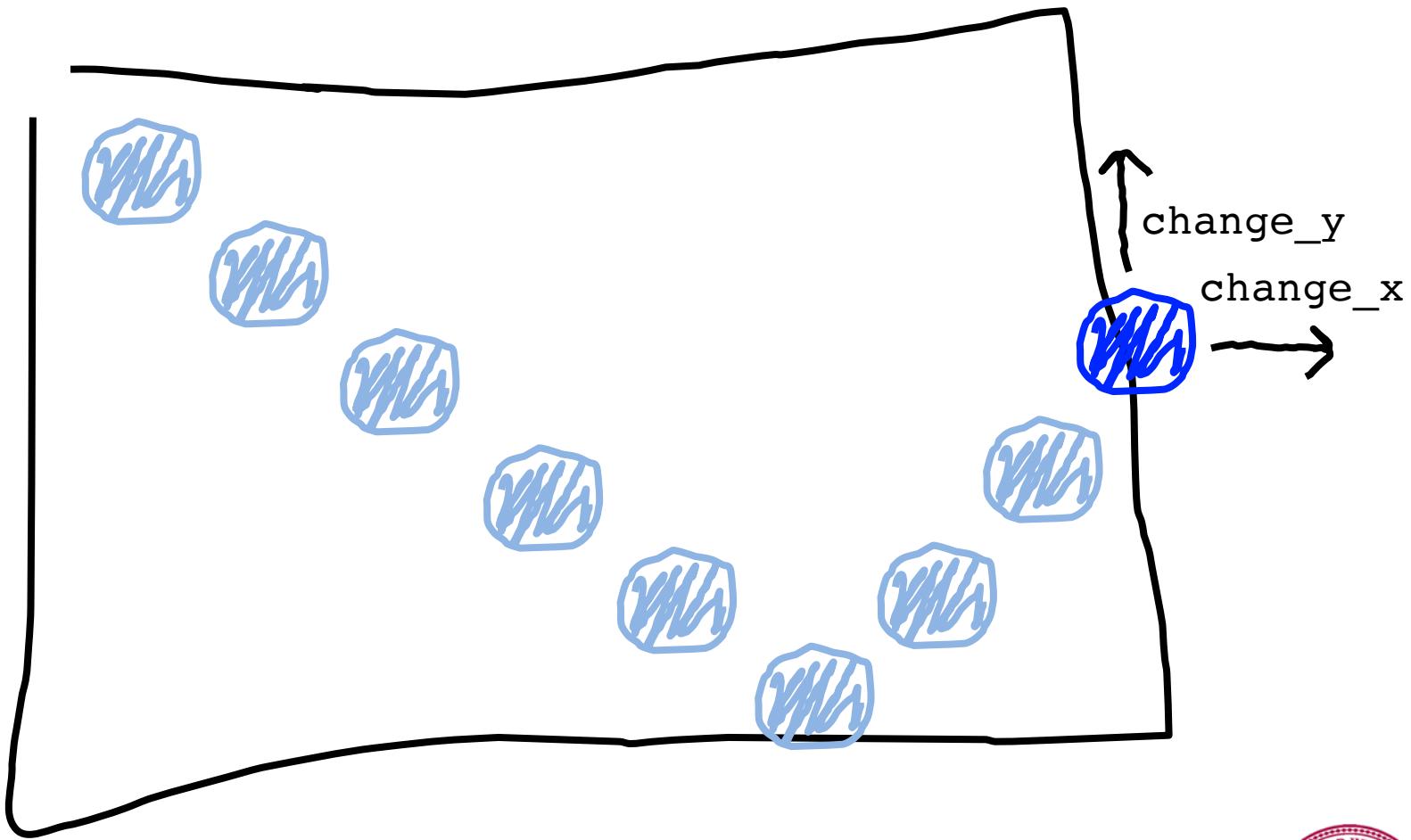
Bouncing Ball

Eighth heartbeat



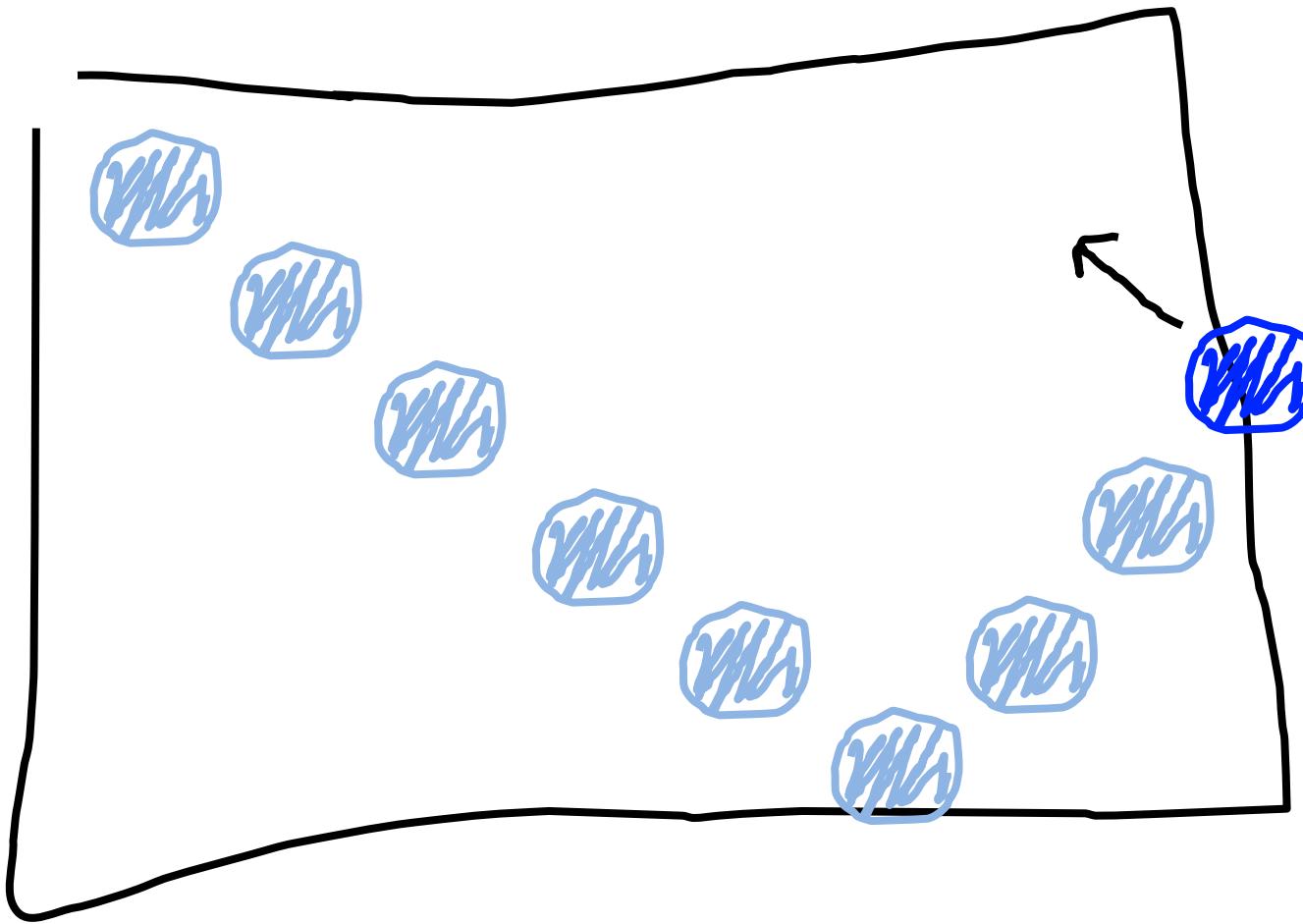
Bouncing Ball

Ninth heartbeat



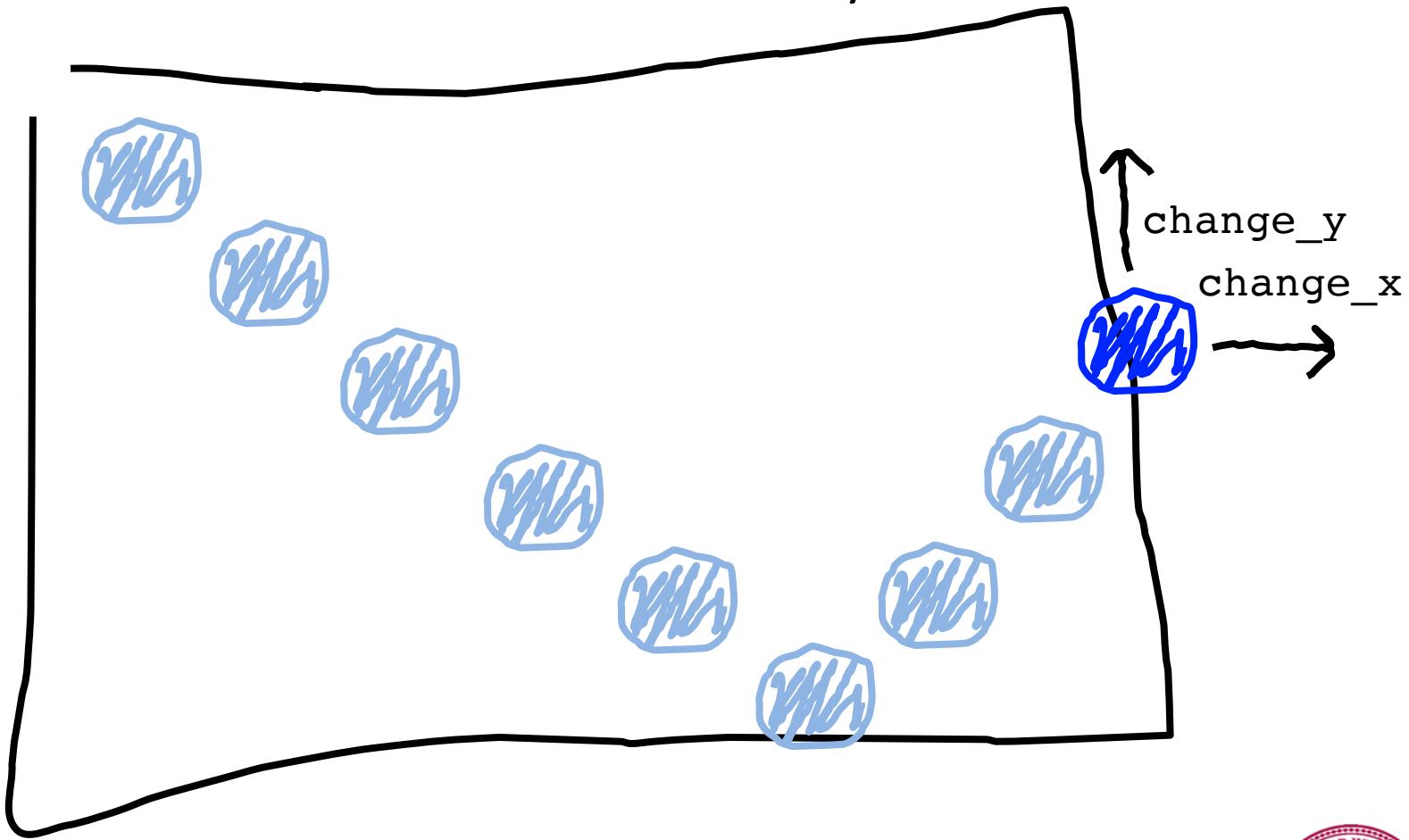
Bouncing Ball

We want this!



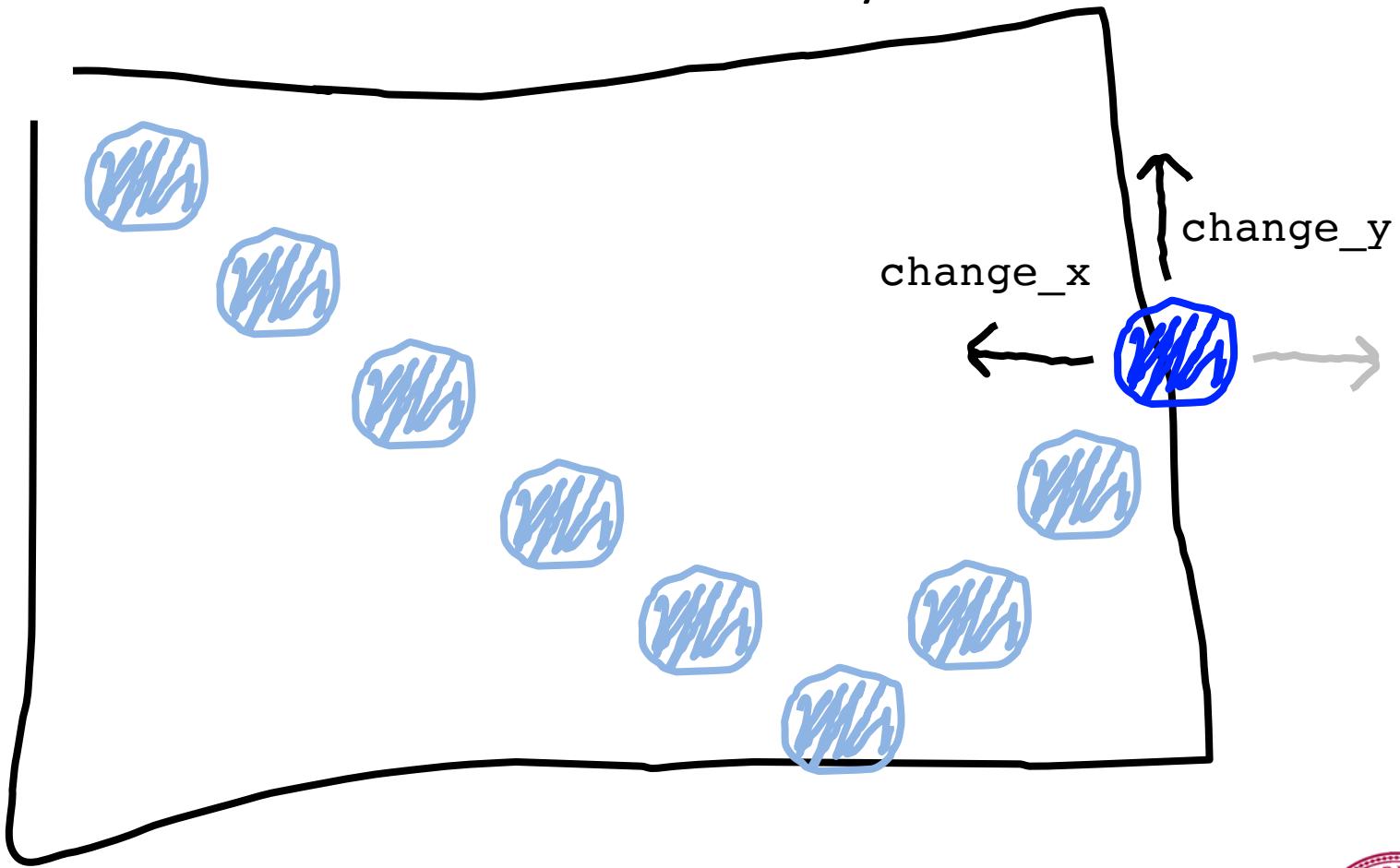
Bouncing Ball

This was our old velocity



Bouncing Ball

This is our new velocity

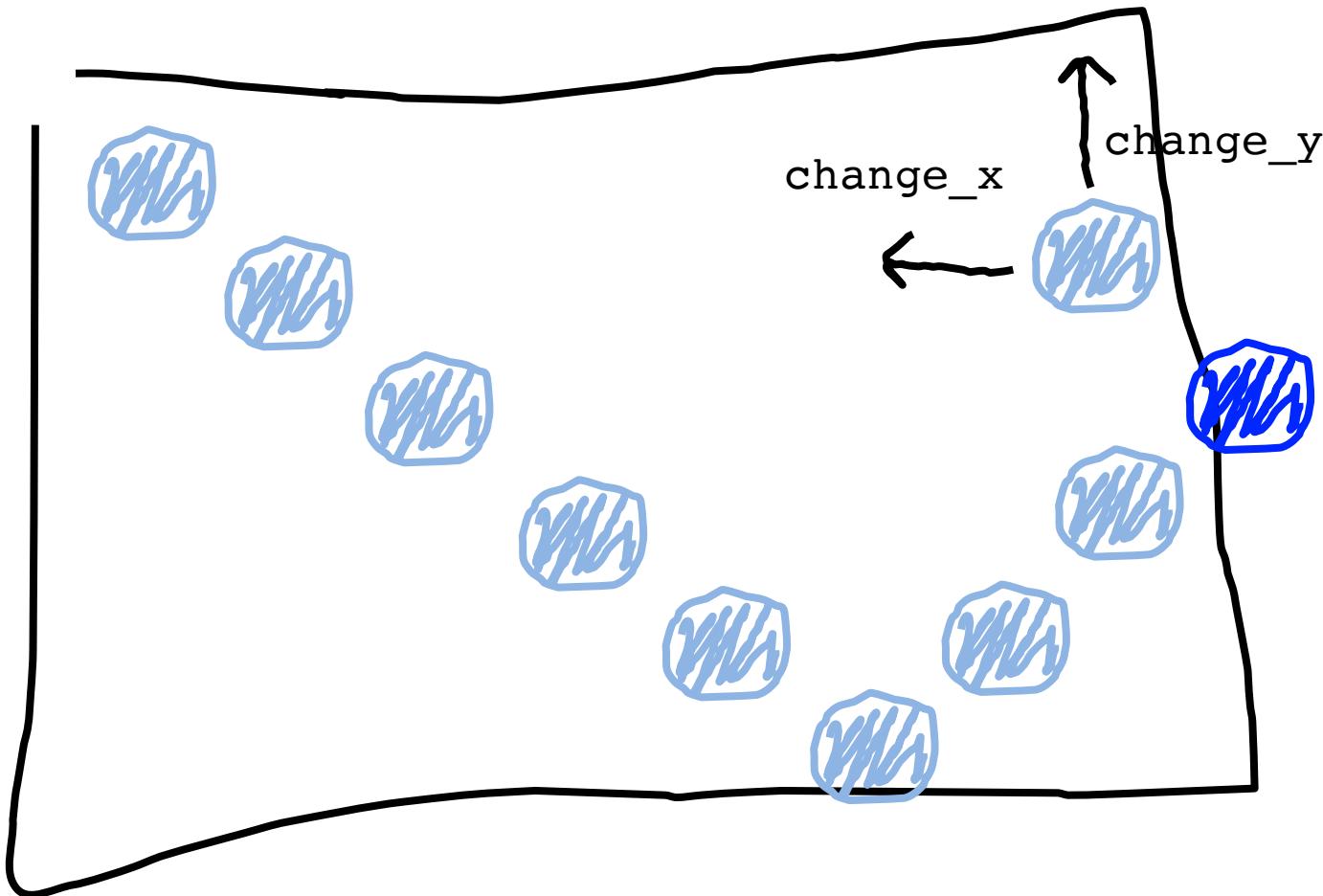


When reflecting horizontally: `change_x = -change_x`



Bouncing Ball

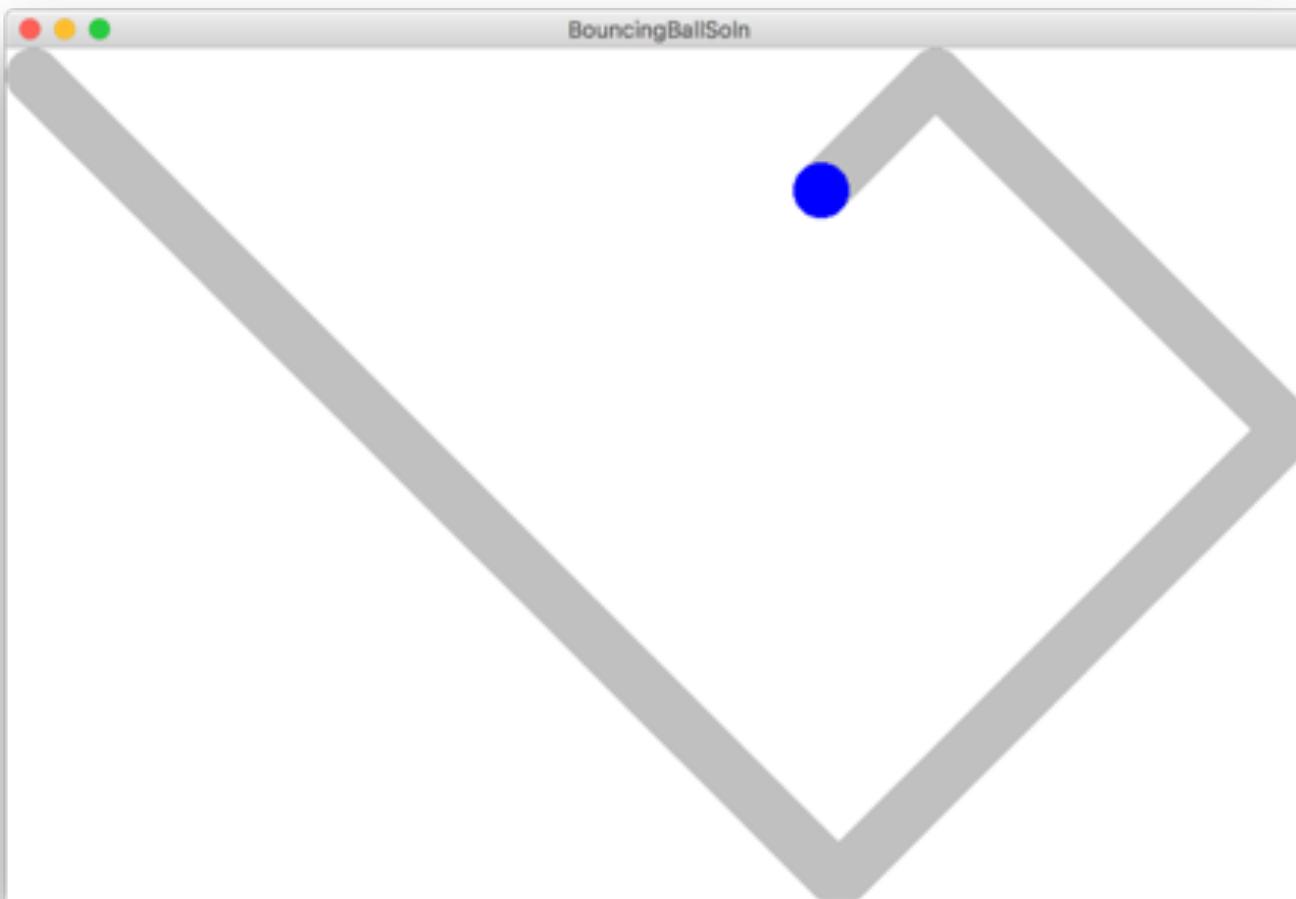
Tenth heartbeat



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



Bouncing Ball



Hold up!

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

If you get a copy when you pass a parameter. Does this copy the canvas??!!

*Large variables (objects) are stored using a reference. Which is like a **URL**. The URL gets copied when you pass the variable*



How do you share google docs?

The screenshot shows a Google Docs interface with the following details:

- Title:** Apollo 11 research
- Toolbar:** Includes File, Edit, View, Insert, Format, Tools, Table, Add-ons, Help, and a note that the last edit was 3 hours ago.
- Share Buttons:** Comments and Share buttons are visible in the top right.
- Document Outline:** A sidebar on the left lists sections: Document Outline, Apollo 11, Summary, The Spacecraft, Design, Command module, Service module, The People, Neil Armstrong, Buzz Aldrin, Mission Highlights, The Launch, The Landing, and Return Trip. The "Summary" section is currently selected.
- Content Area:** The main area contains the following text:

Apollo 11

Summary

This is a research paper about the Apollo 11 moon mission in which Neil Armstrong, Buzz Aldrin, and Michael Collins landed at Tranquility Base on the moon. The Apollo 11 lunar module, AKA The Eagle, landed on the moon on July 20, 1969. When they landed, the message they sent back to Mission Control was 'Tranquility Base here. The Eagle has landed.'

The Spacecraft

The Apollo 11 mission had three spacecraft: the Command Module Columbia, a Service Module, and the Lunar Module Eagle. Columbia was the only part of the spacecraft to return to Earth.

Design

The key NASA spacecraft involved in the Apollo 11 mission were the following: a Saturn V rocket, an Apollo CSM-107 (Command/Service Module) and an Apollo LM-5 (Lunar Module, AKA "The Eagle").

Command module

The Command/Service Module (CSM) was one of two spacecraft, along with the Lunar Module, used for the United States Apollo program which landed astronauts on the Moon. It was built for NASA by North American Aviation. It was launched by itself into suborbital and low Earth orbit test missions with the Saturn IB launch vehicle, and three times by itself and nine times with the Lunar Module as part of the Apollo spacecraft assembly on the larger Saturn V launch vehicle, which was capable of sending it to the Moon.

Service module

The Service Module contained oxygen, water, and electric power for the command module. It also housed the service orientation system—the rocket engine that put the spacecraft into lunar orbit and later hoisted it back

<https://docs.google.com/document/d/1eBtnEiiI3KHe fFS-kSAOpXqeSXpbfTTMlmOgj6I9dvk/>



```
def main():
```

```
    canvas = make_canvas(...)
```

```
    make_ball(canvas)
```

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

```
    canvas.create_rectangle( ... , fill='blue')
```

stack

heap

```
main
```



```
def main():
```

```
    canvas = make_canvas(...)
```

```
    make_ball(canvas)
```

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

```
    canvas.create_rectangle( ... , fill='blue')
```

stack

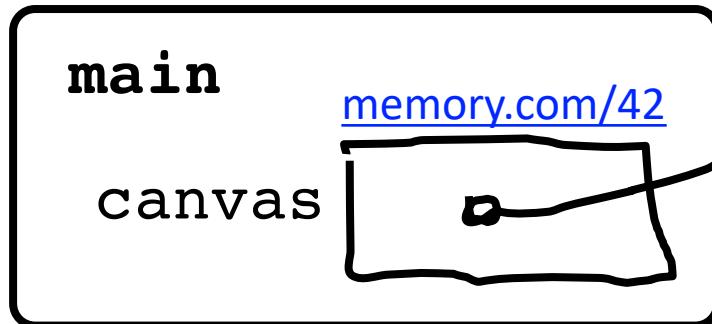
main

<memory.com/42>

canvas

heap

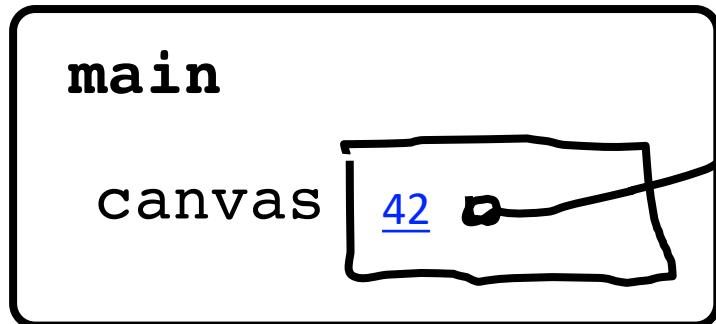
<memory.com/42>



```
def main():
    canvas = make_canvas(...)
    make_ball(canvas)
```

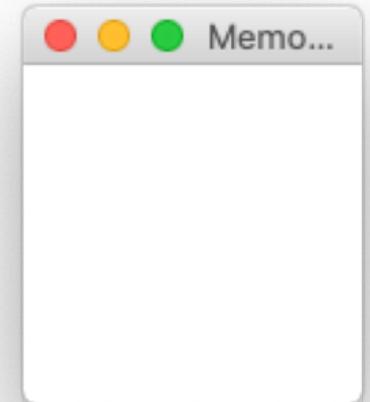
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack



heap

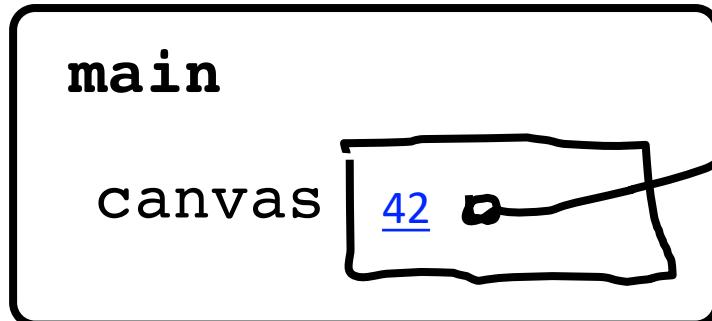
memory.com/42



```
def main():
    canvas = make_canvas(...)
    make_ball(canvas)
```

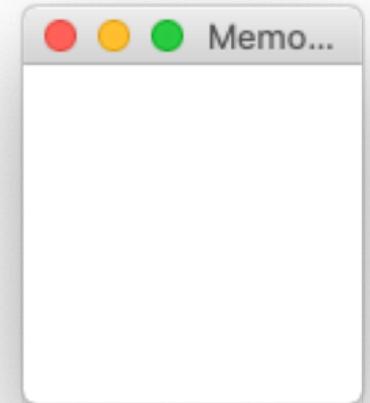
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack

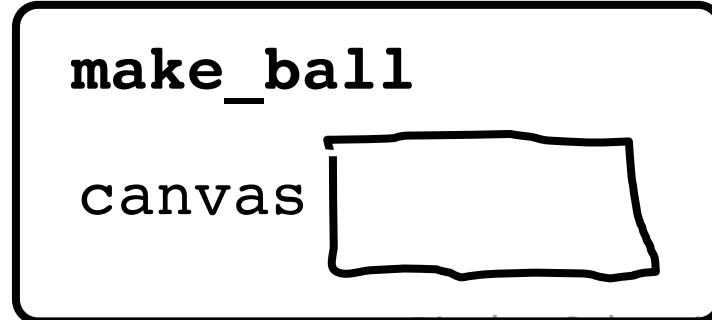


heap

memory.com/42



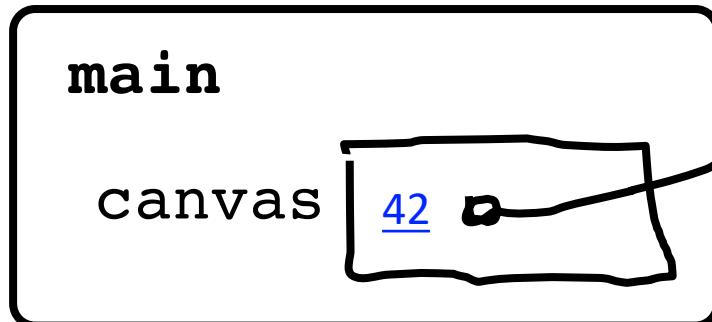
make_ball



```
def main():
    canvas = make_canvas(...)
    make_ball(canvas)
```

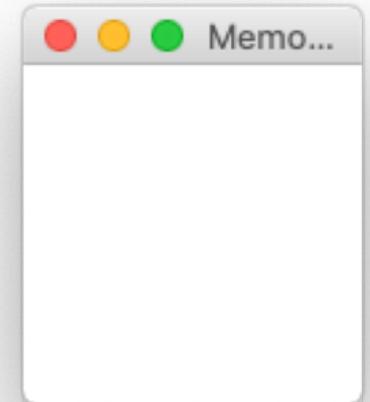
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack

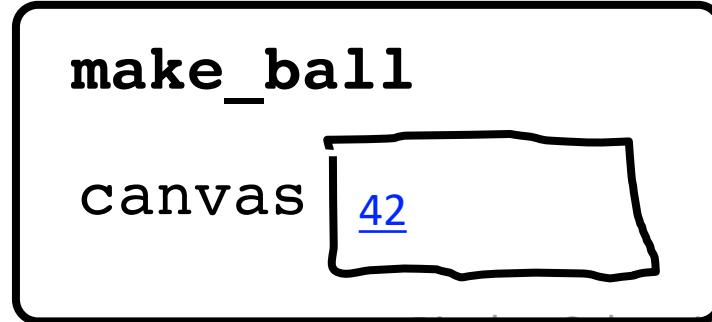


heap

memory.com/42



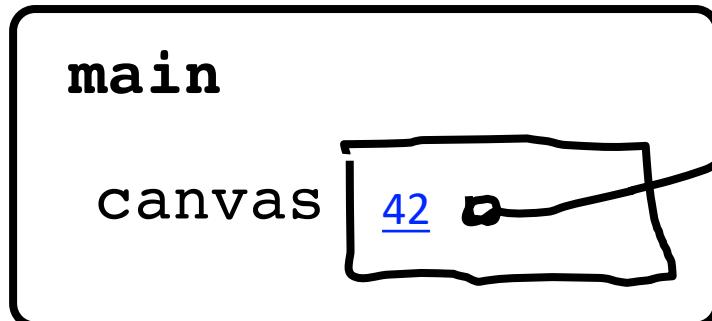
make_ball



```
def main():
    canvas = make_canvas(...)
    make_ball(canvas)
```

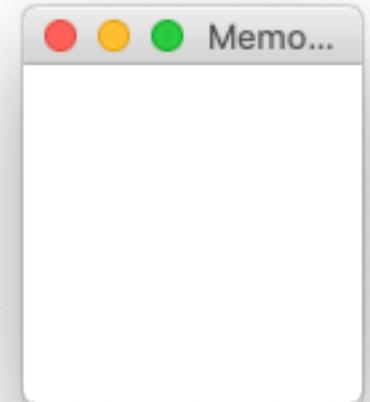
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack

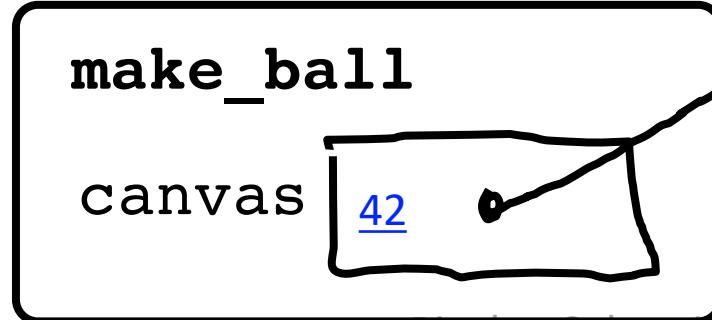


heap

memory.com/42



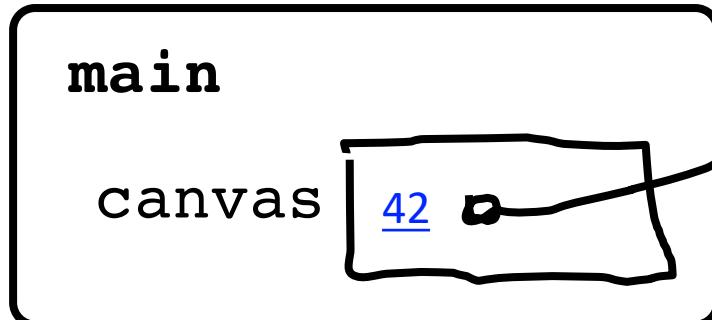
make_ball



```
def main():
    canvas = make_canvas(...)
    make_ball(canvas)
```

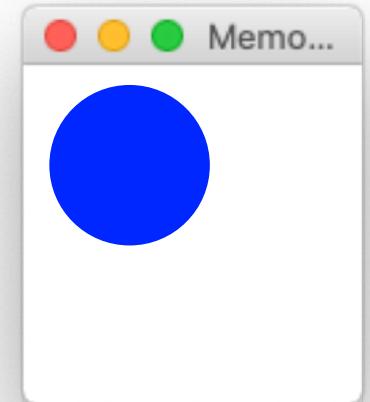
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack

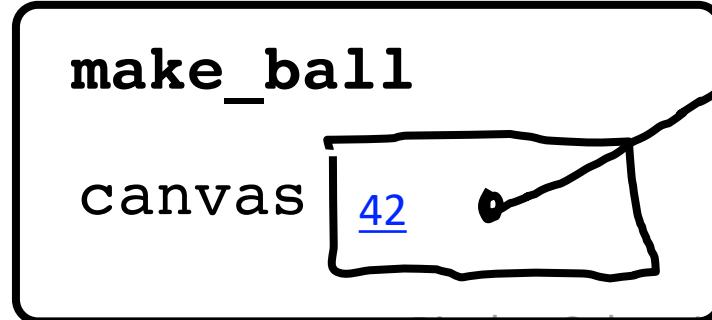


heap

memory.com/42



make_ball



```
def main():
```

```
    canvas = make_canvas(...)
```

```
    make_ball(canvas)
```

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

```
    canvas.create_oval( ... , fill='blue')
```



stack

main

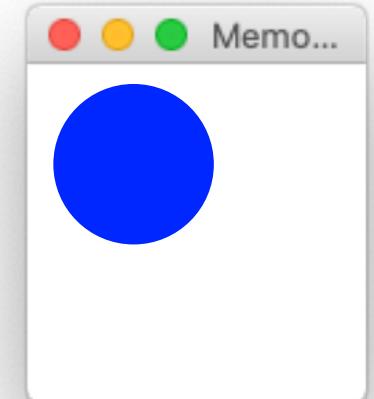
canvas

42



heap

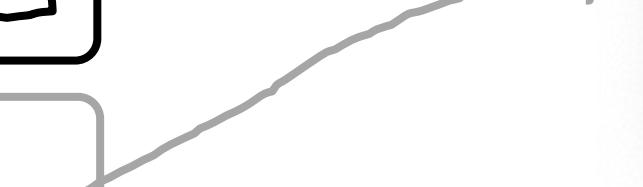
memory.com/42



make_ball

canvas

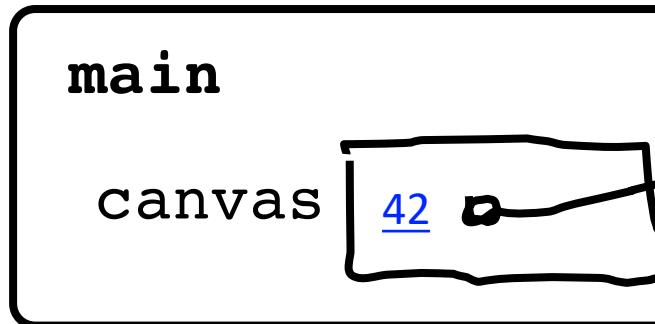
42



```
def main():
    canvas = make_canvas(...)
    make_ball(canvas)
```

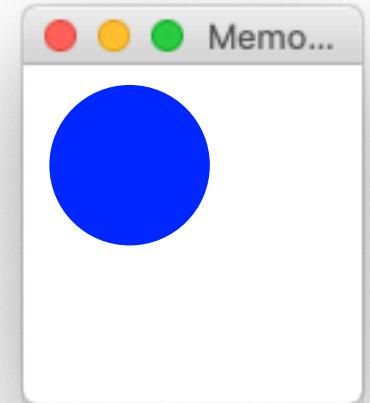
```
def make_ball(canvas):
    canvas.create_oval( ... , fill='blue')
```

stack



heap

memory.com/42





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

That allows functions to
modify the variable

When passed as parameters

Variables that act like
their **value is copied**

boolean
integer
float
string

Variables that act like
their **URL is copied**

canvas
pixel
SimpleImage
list



Learning Goals

1. Feel more confident writing methods
2. Write animated programs



Special Graphics Functions

```
# get the x location of the mouse
mouse_x = canvas.winfo_pointerx()

# 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)

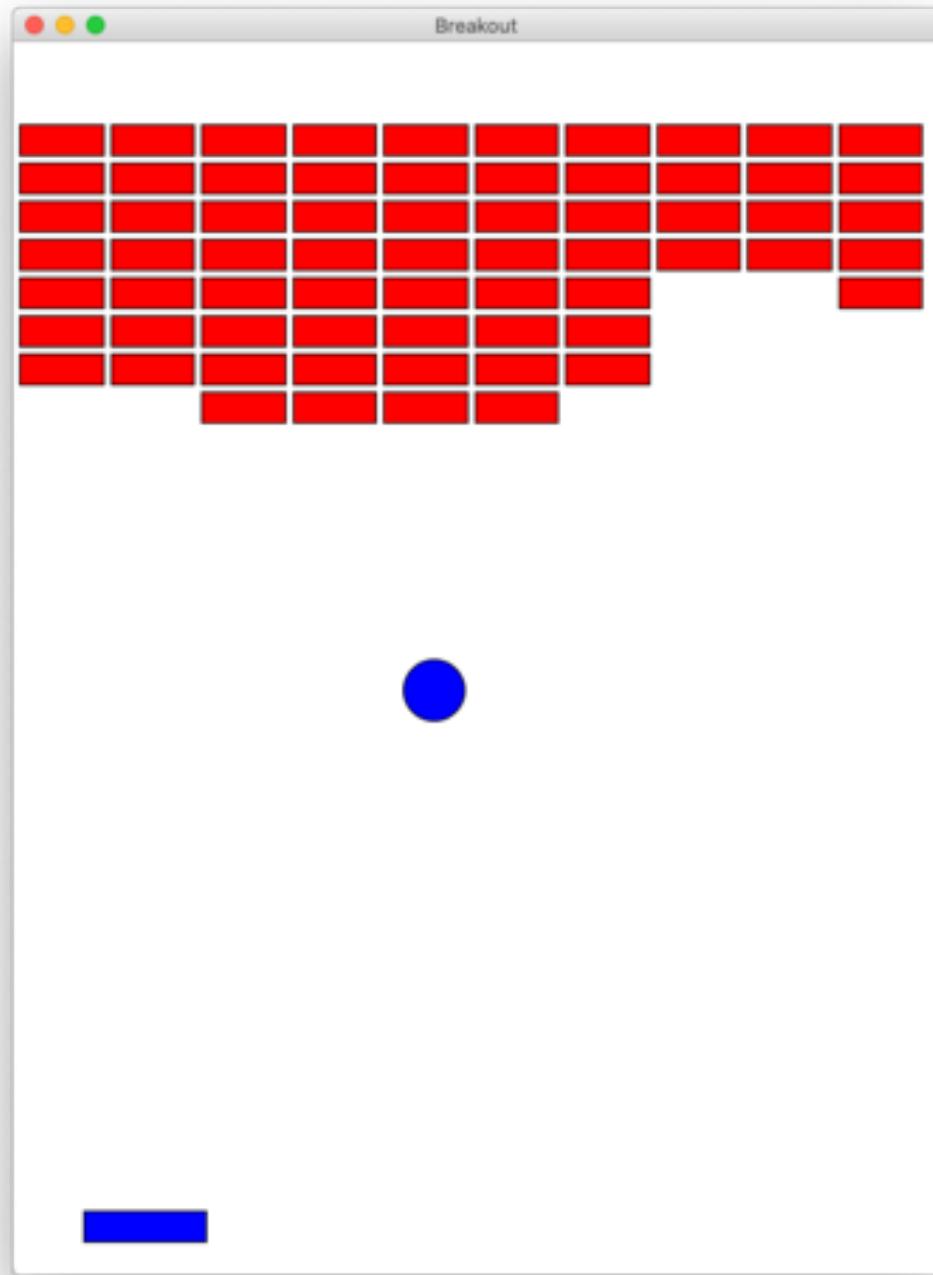
# get the coordinates of a shape
coord_list = canvas.coords(shape)

# return a list of elements in a rectangle area
results = canvas.find_overlapping(x1, y1, x2, y2)

# you can change a shapes color too
canvas.itemconfig(shape, fill=new_color, outline=...)
```

Come back on Monday to learn about lists!





Piech + Sahami, CS106A, Stanford University

