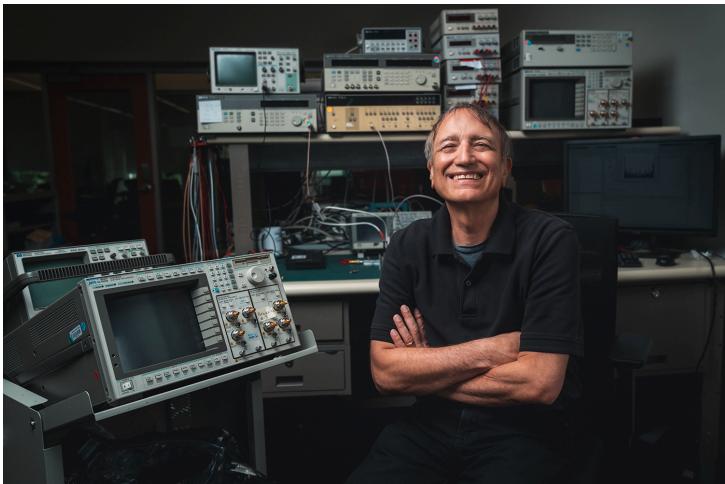




Animation

Chris Piech
CS Bridge 2020

Turing Award Winner Talks to CS106A



- **Turing Award** is like the nobel prize in CS
- Professor at Stanford
- Founding employee at Pixar
- Wrote RenderMan, won 3 Academy Awards
- And just a really wonderful human.

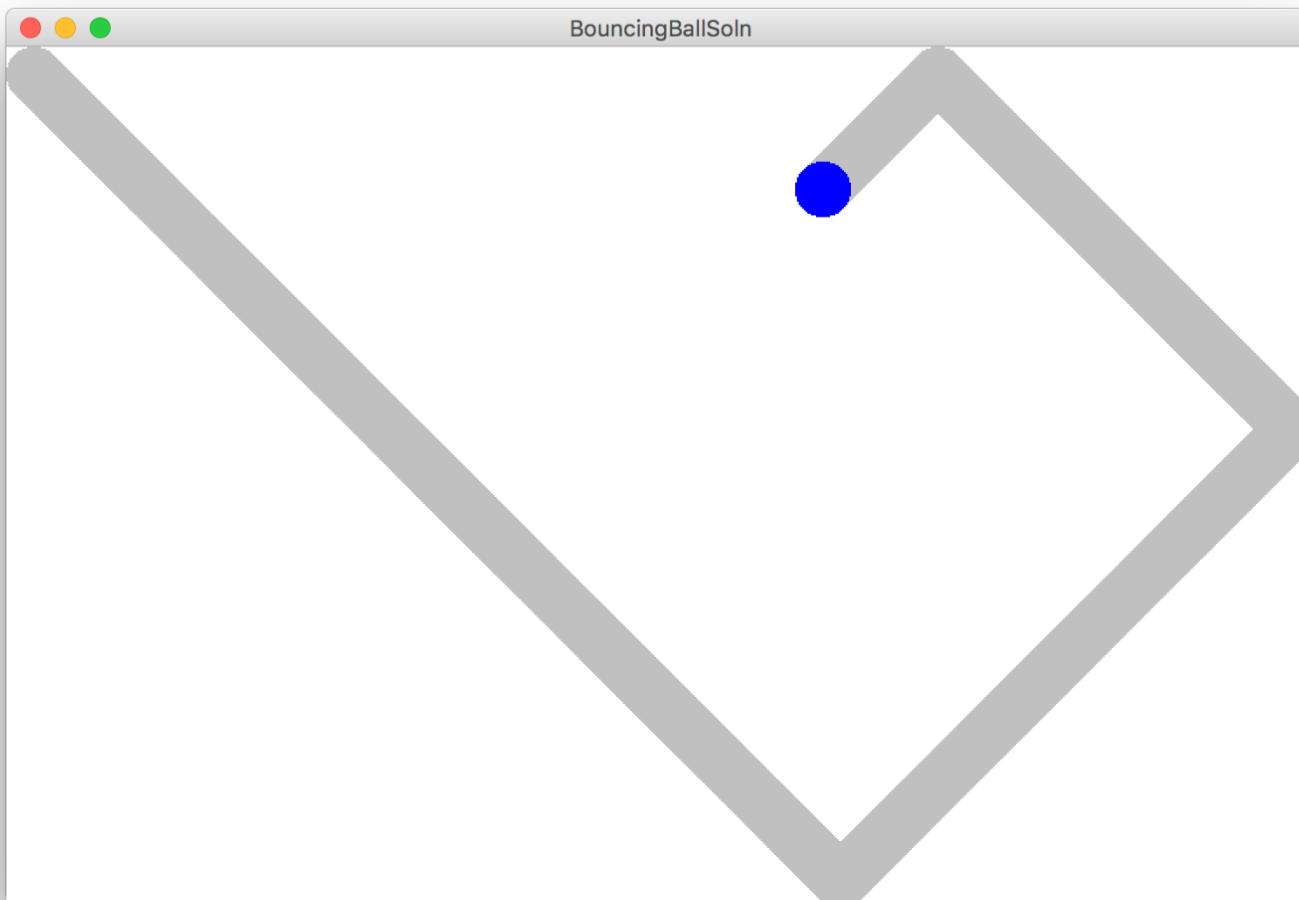


Learning Goals

1. Feel more confident debugging
2. Write animated programs



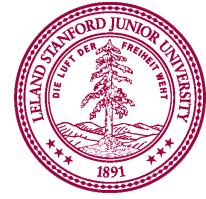
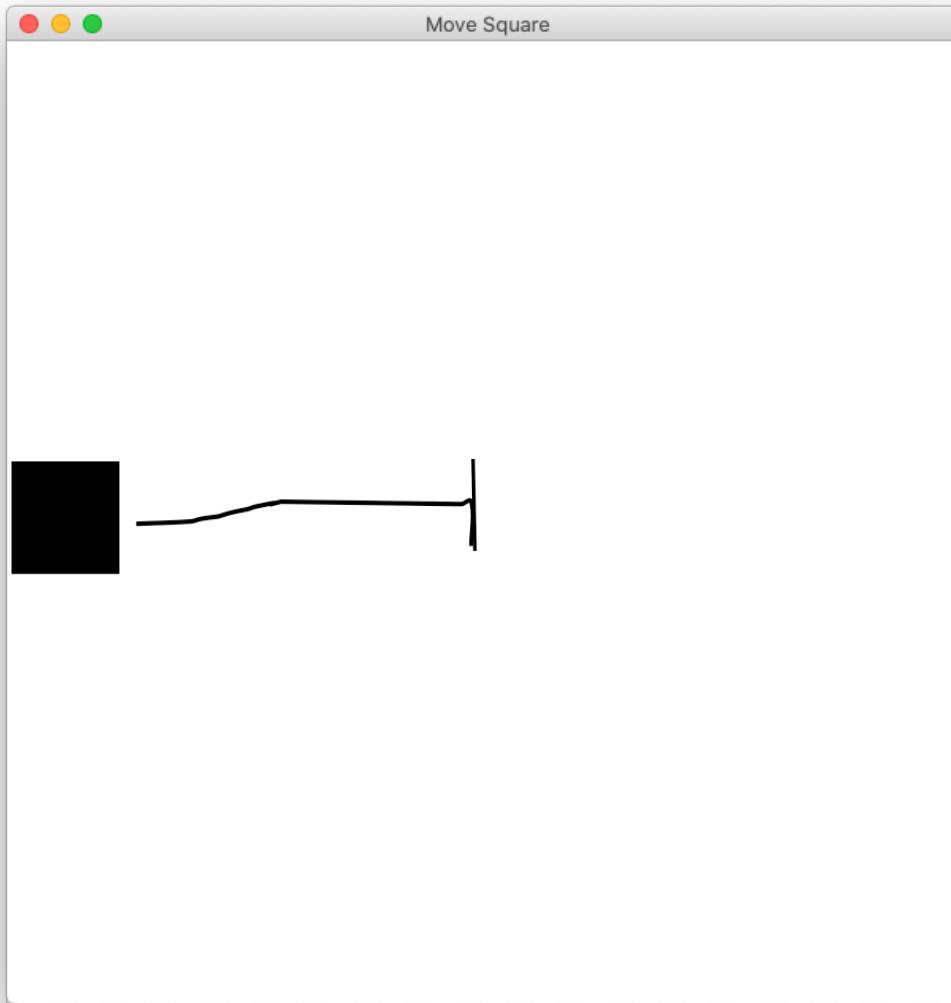
You will be able to write Bouncing Ball



Great foundation



Move to Center



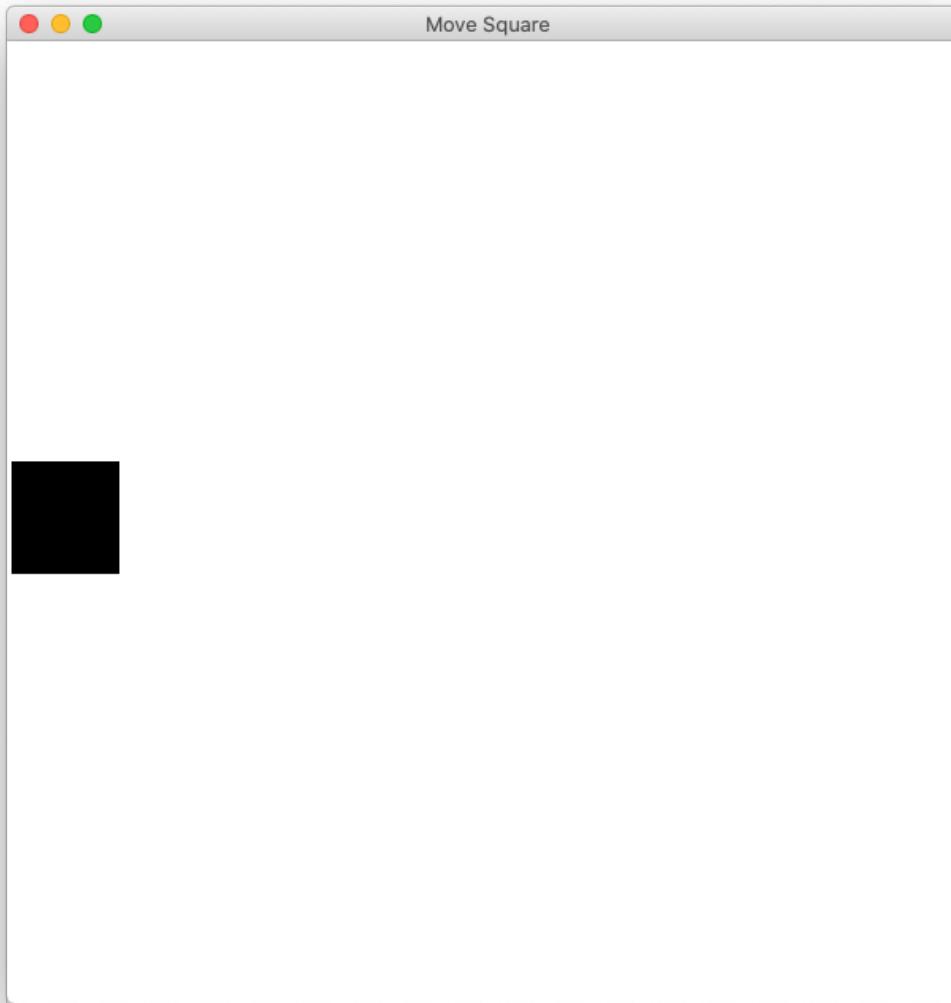
In our last episode...

Graphics from tkinter (aka tk)

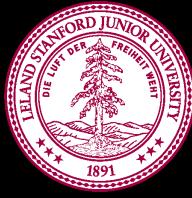
```
from graphics import Canvas
```



Add square



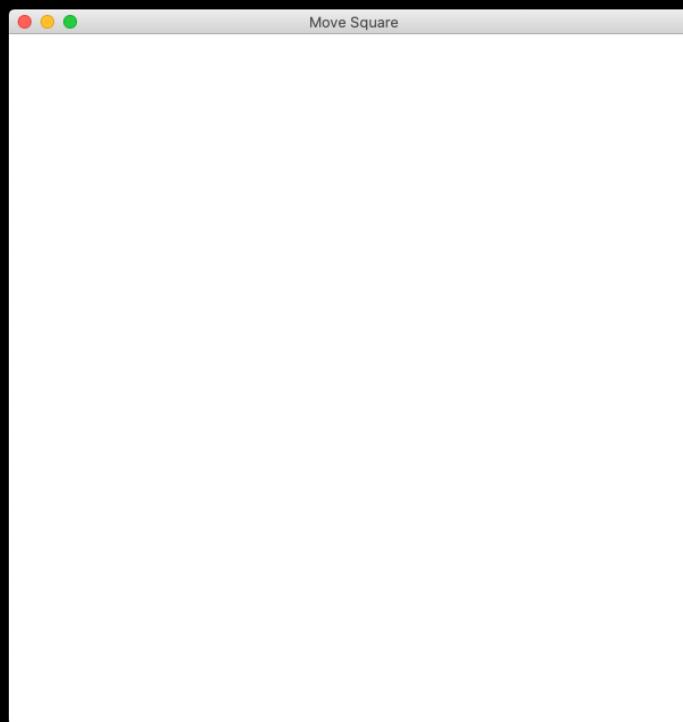
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y)
    canvas.set_color('black')
    canvas.mainloop()
```



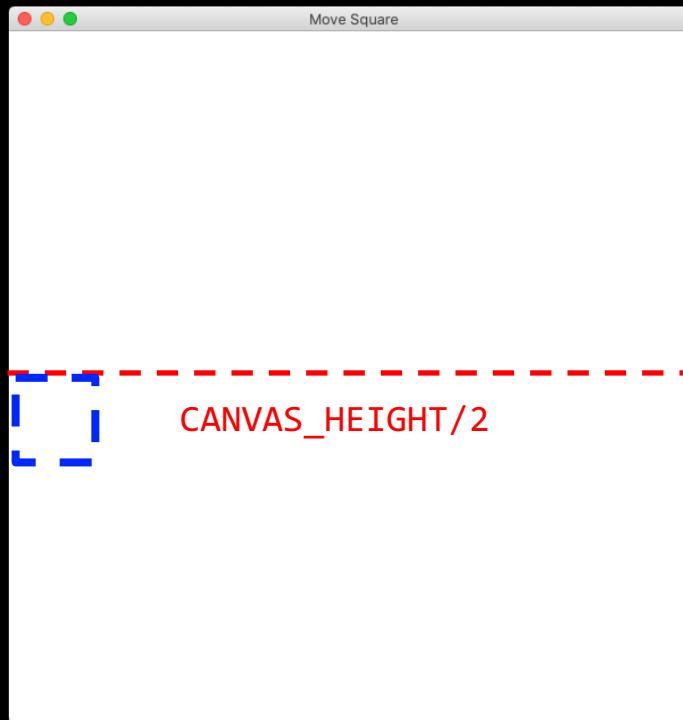
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y)
    canvas.set_color('black')
    canvas.mainloop()
```



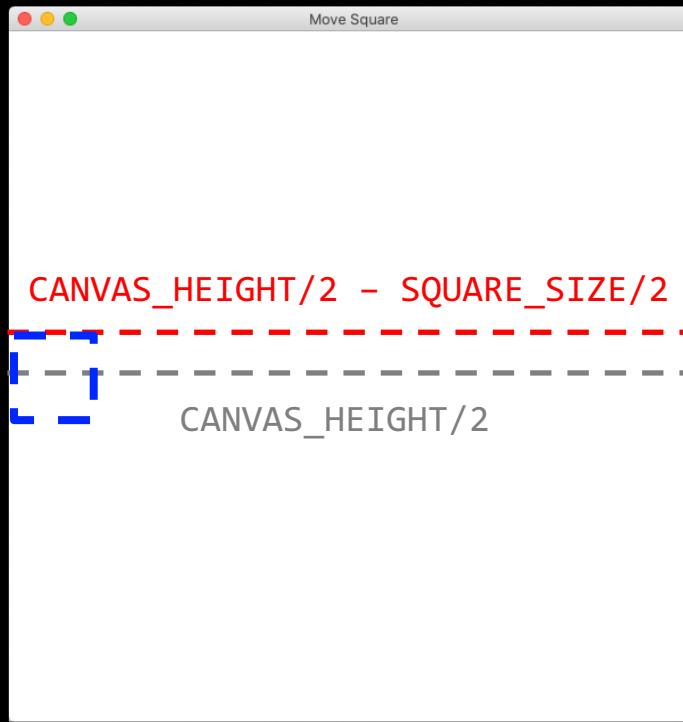
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y)
    canvas.set_color('black')
    canvas.mainloop()
```



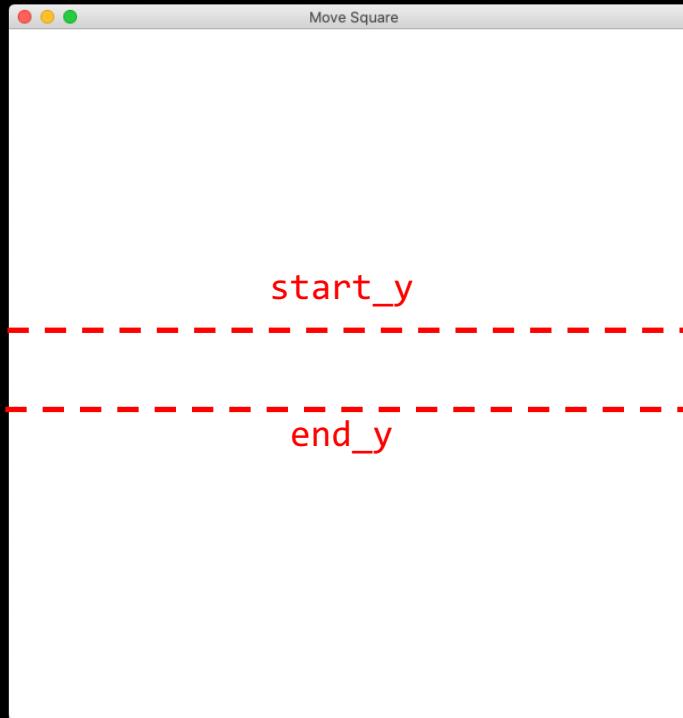
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y)
    canvas.set_color('black')
    canvas.mainloop()
```



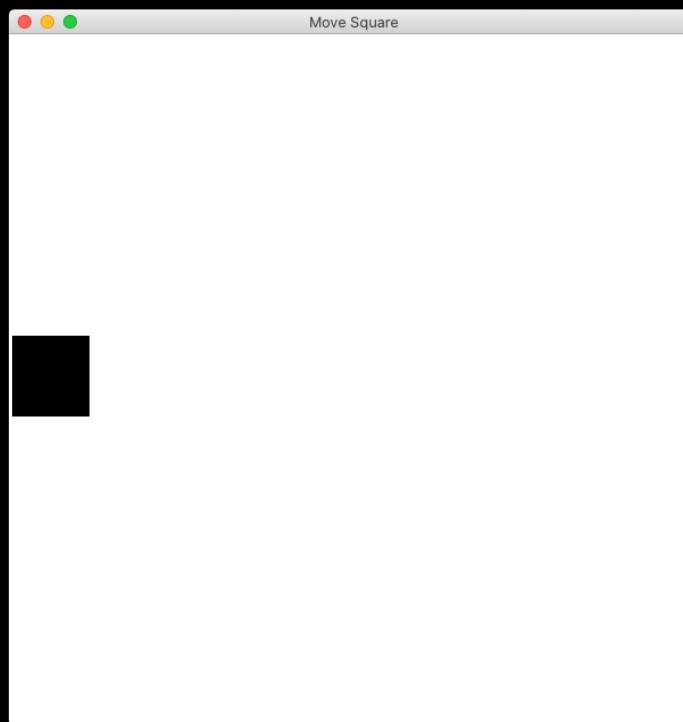
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y)
    canvas.set_color('black')
    canvas.mainloop()
```



```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    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)
    canvas.set_color('black')
    canvas.mainloop()
```



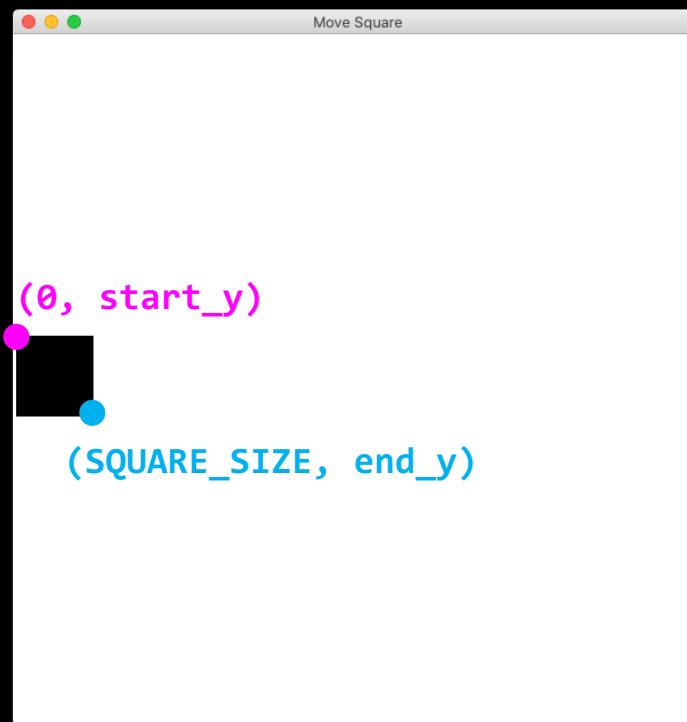
```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    start_y = CANVAS_HEIGHT / 2 - SQUARE_SIZE / 2
    end_y = start_y + SQUARE_SIZE
    rect = canvas.create_rectangle(0, start_y, SQUARE_SIZE, end_y)
    canvas.set_color('black')
    canvas.mainloop()
```



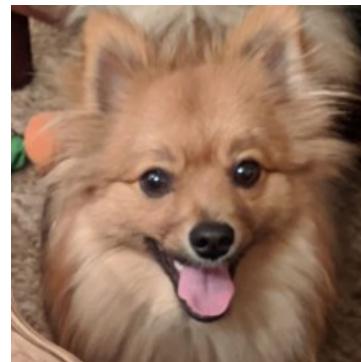
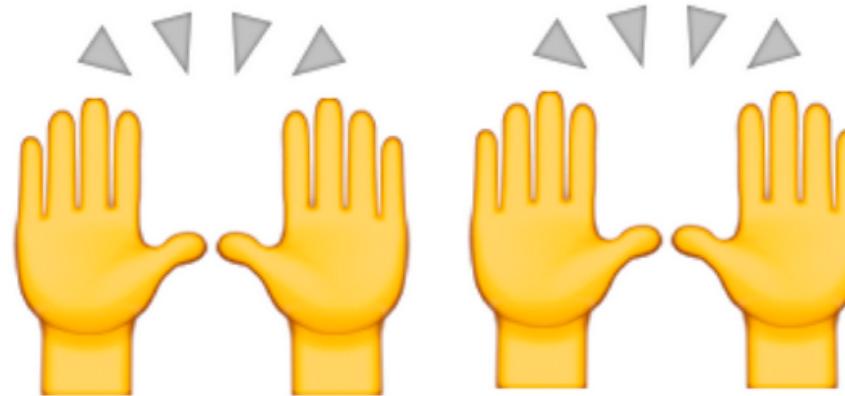
Some “heavy duty” variables allow you to call functions on them



```
def main():
    canvas = Canvas(CANVAS_WIDTH, CANVAS_HEIGHT)
    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)
    canvas.set_color('black')
    canvas.mainloop()
```



You're now all graphics programmers!

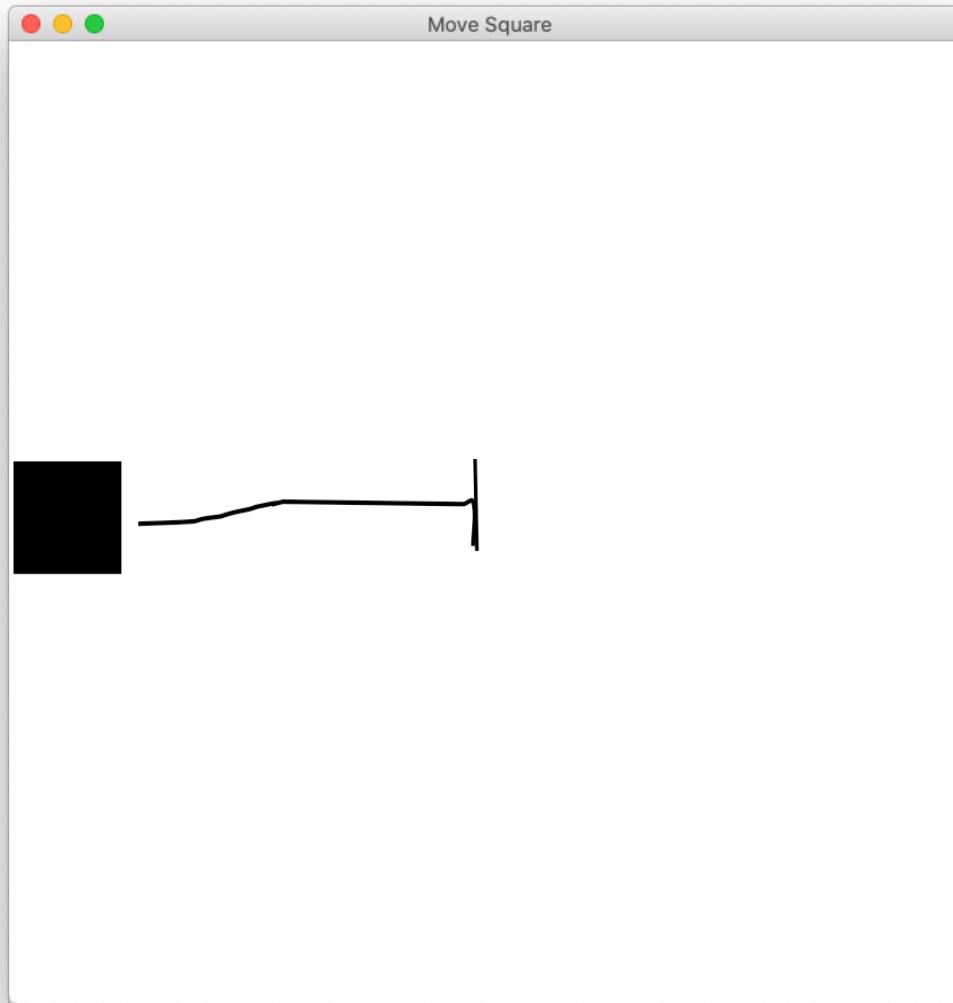


Woot!

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



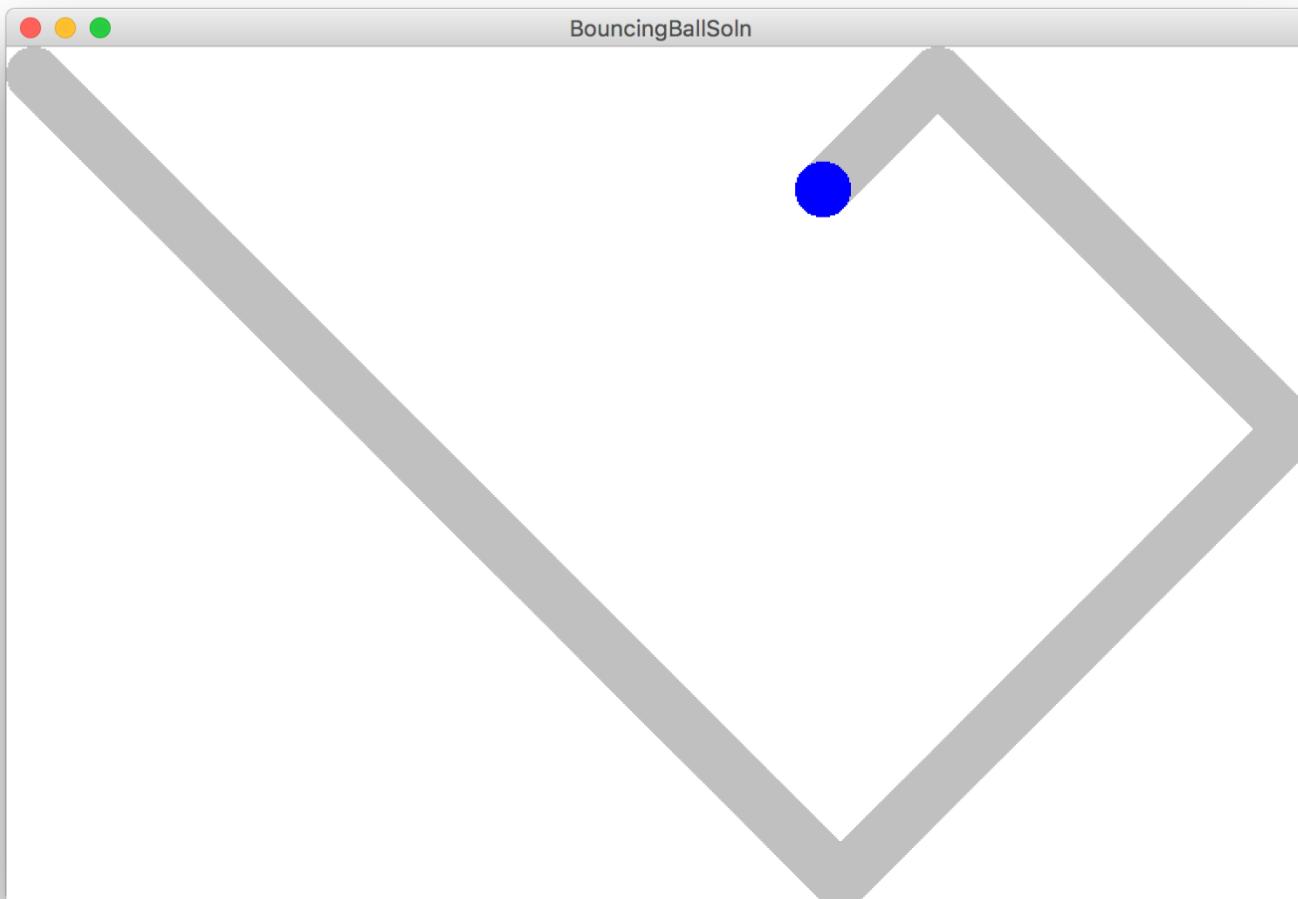
Move To Center

```
def main():
    # setup
    canvas = 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)
```

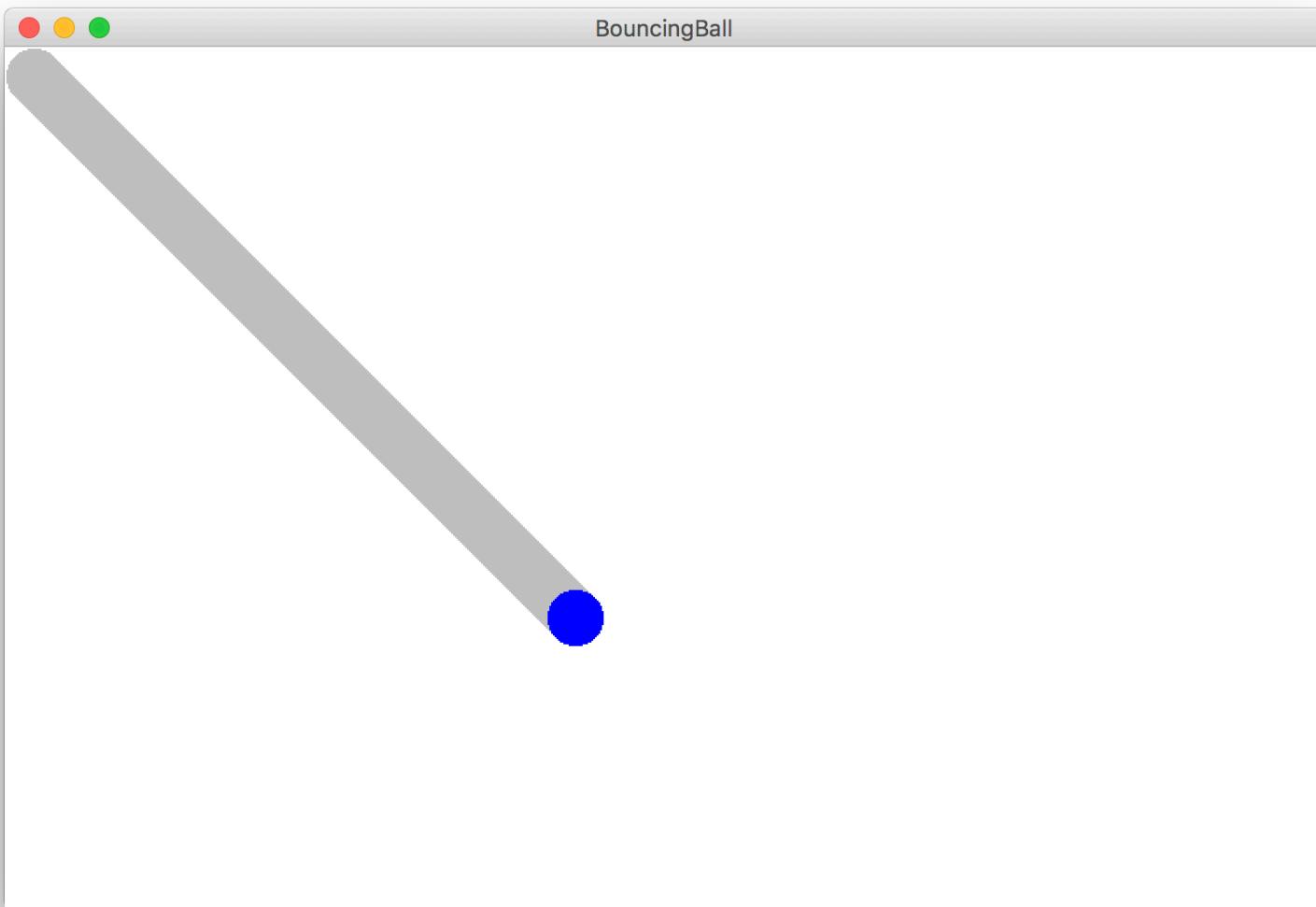


We are ready...

Bouncing Ball

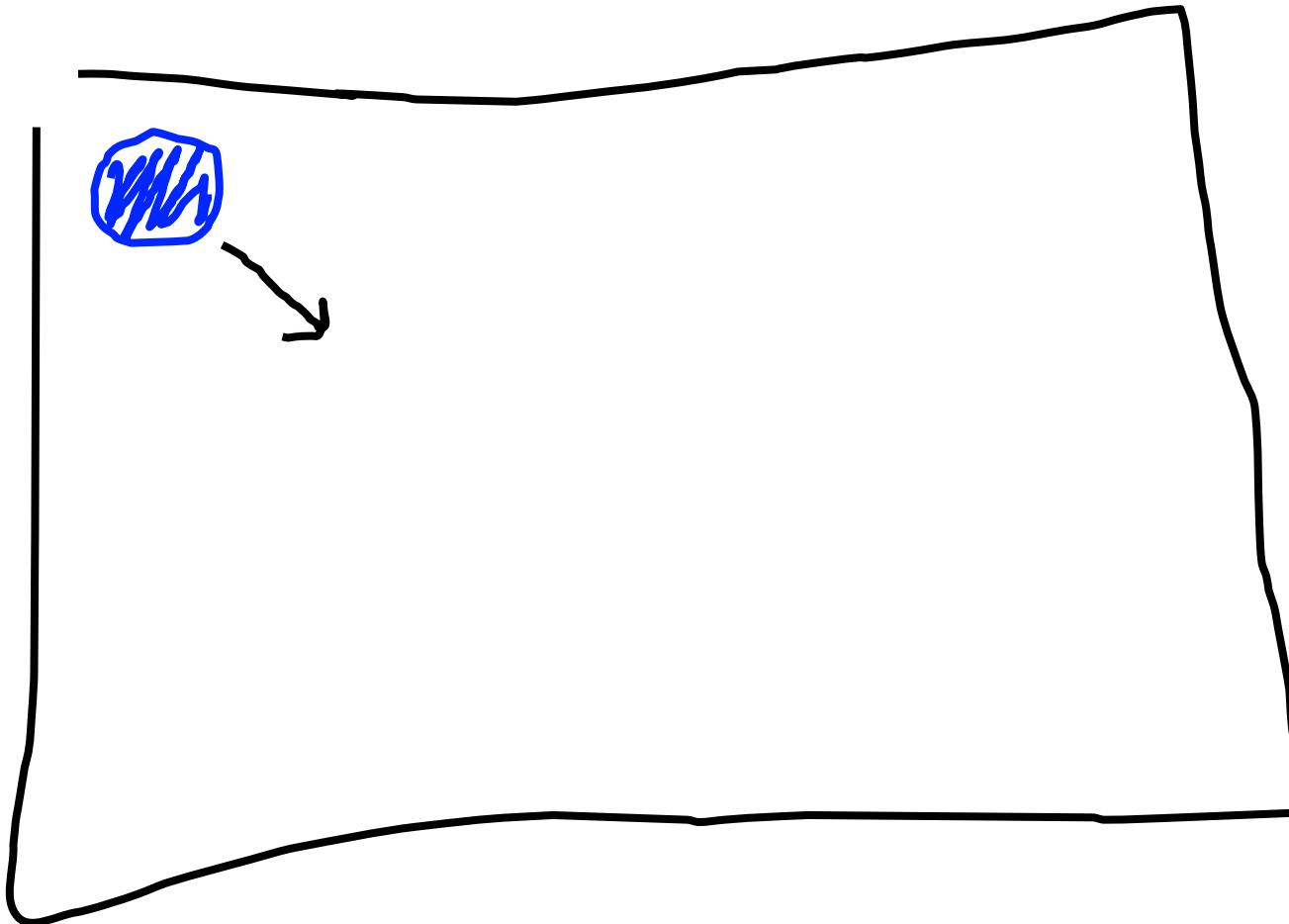


Milestone #1



Bouncing Ball

First heartbeat

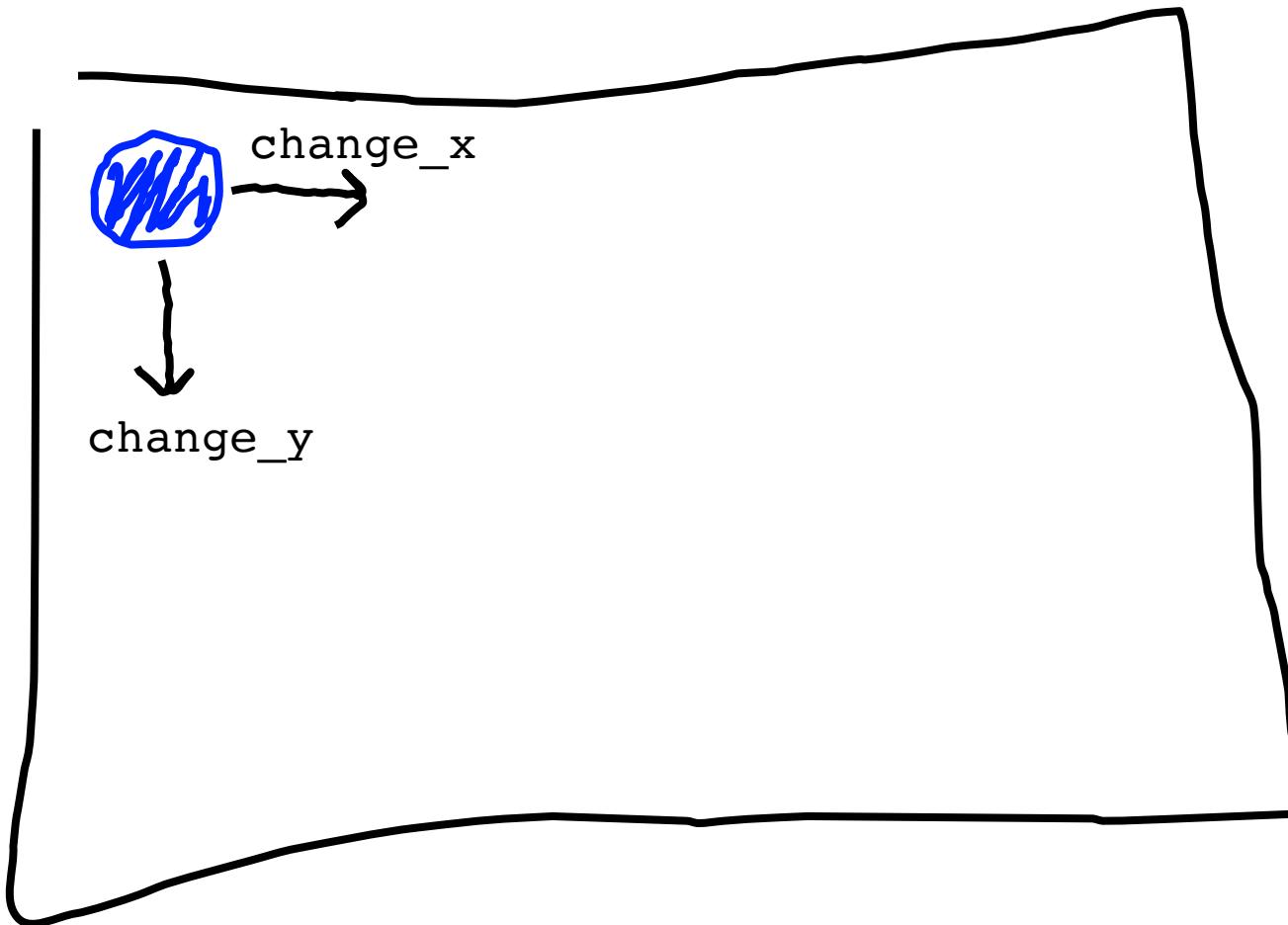


Key variable: how much the ball position
change each heartbeat?



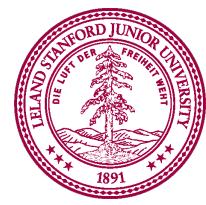
Bouncing Ball

First heartbeat



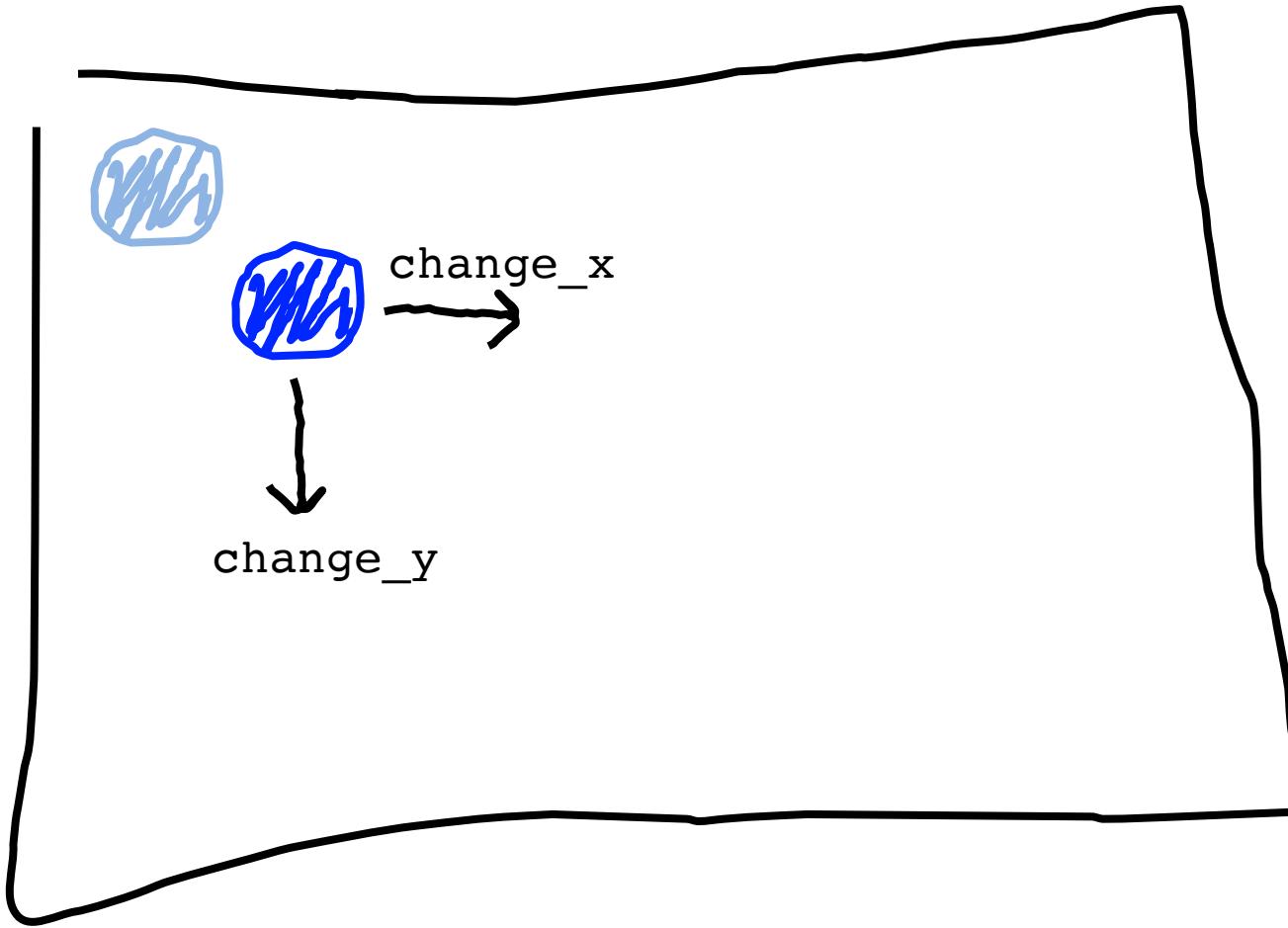
The **move** function takes in
a change in x and a change in y

Pi



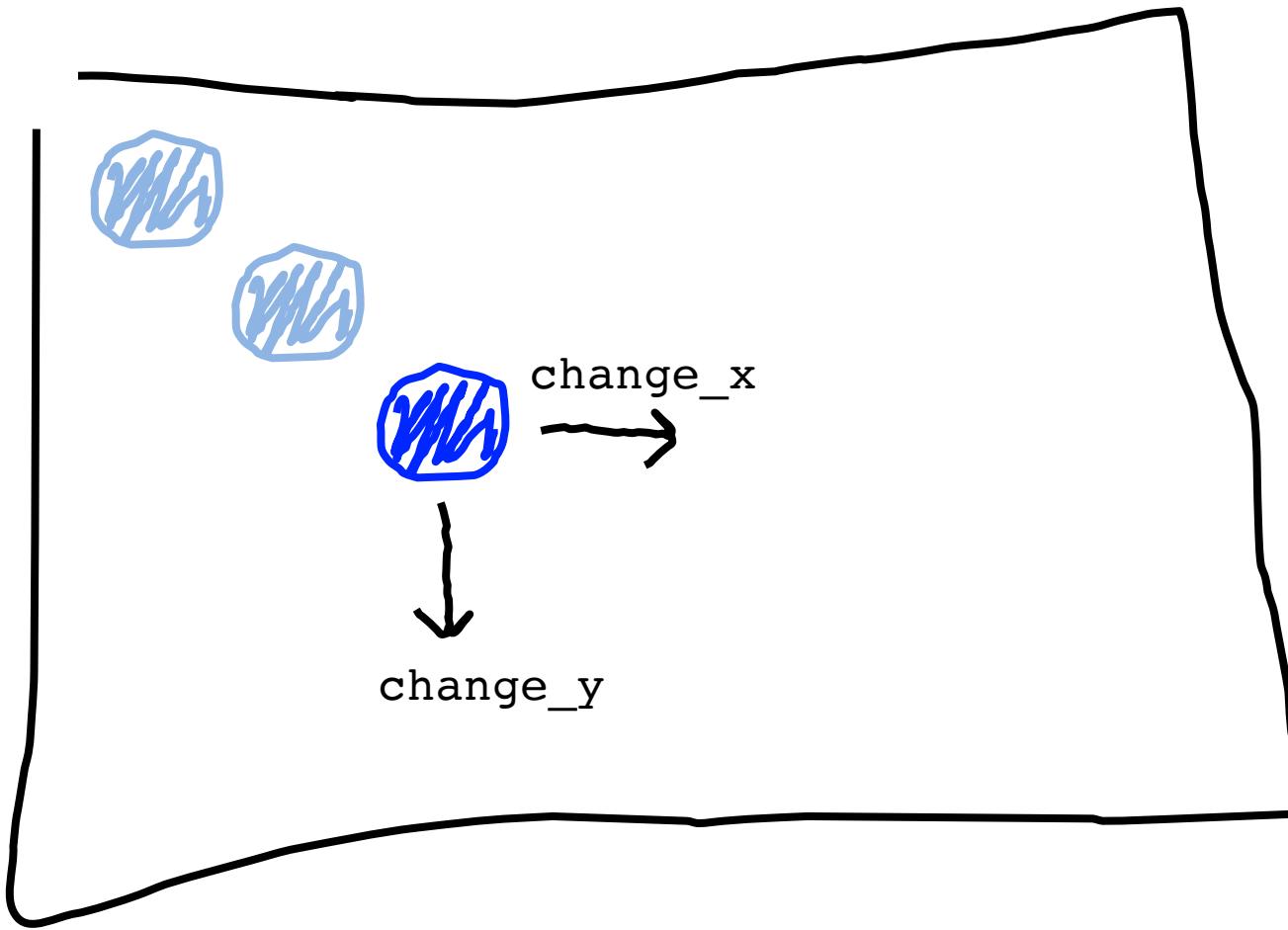
Bouncing Ball

Second heartbeat



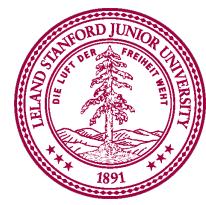
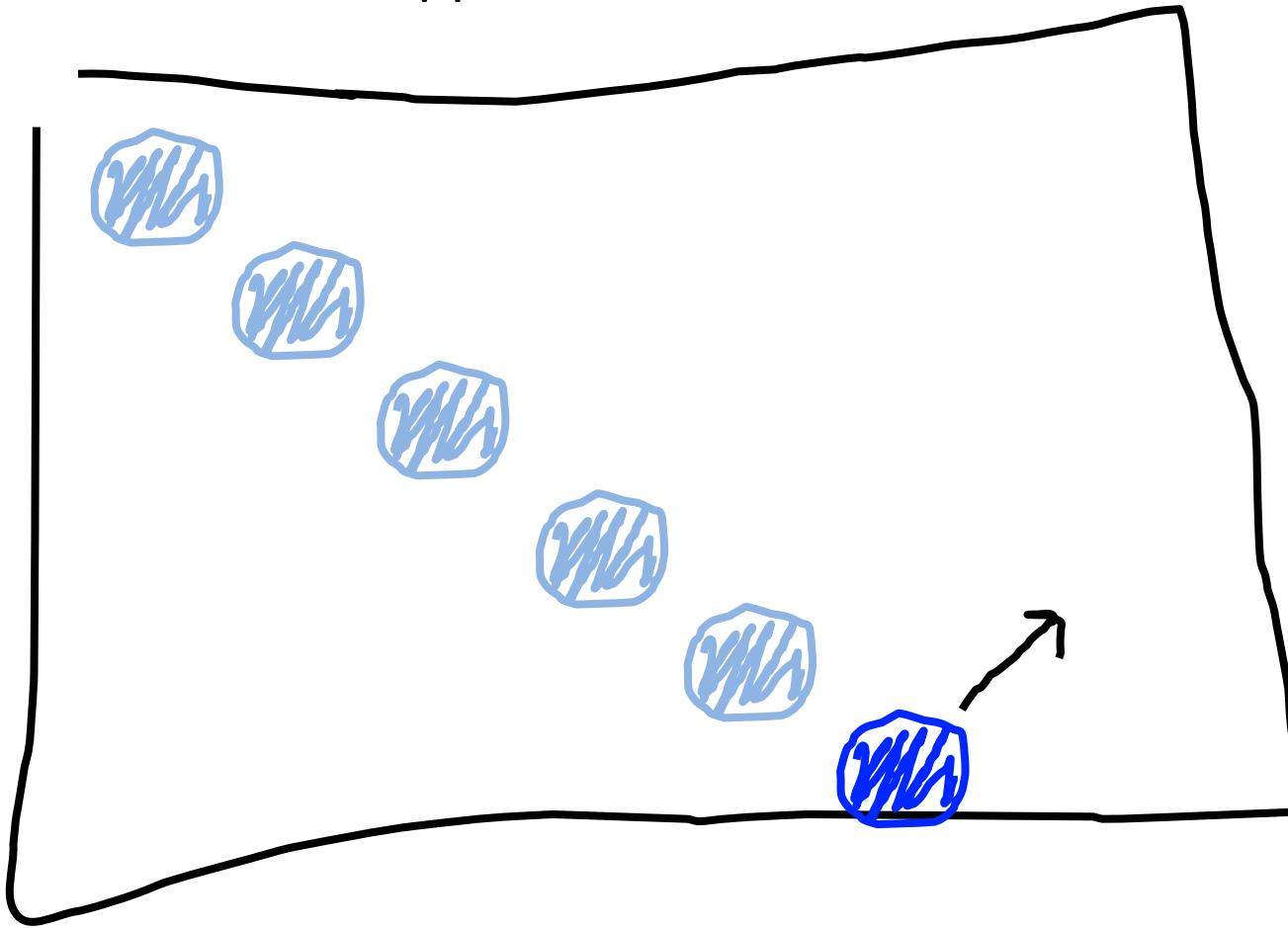
Bouncing Ball

Third heartbeat



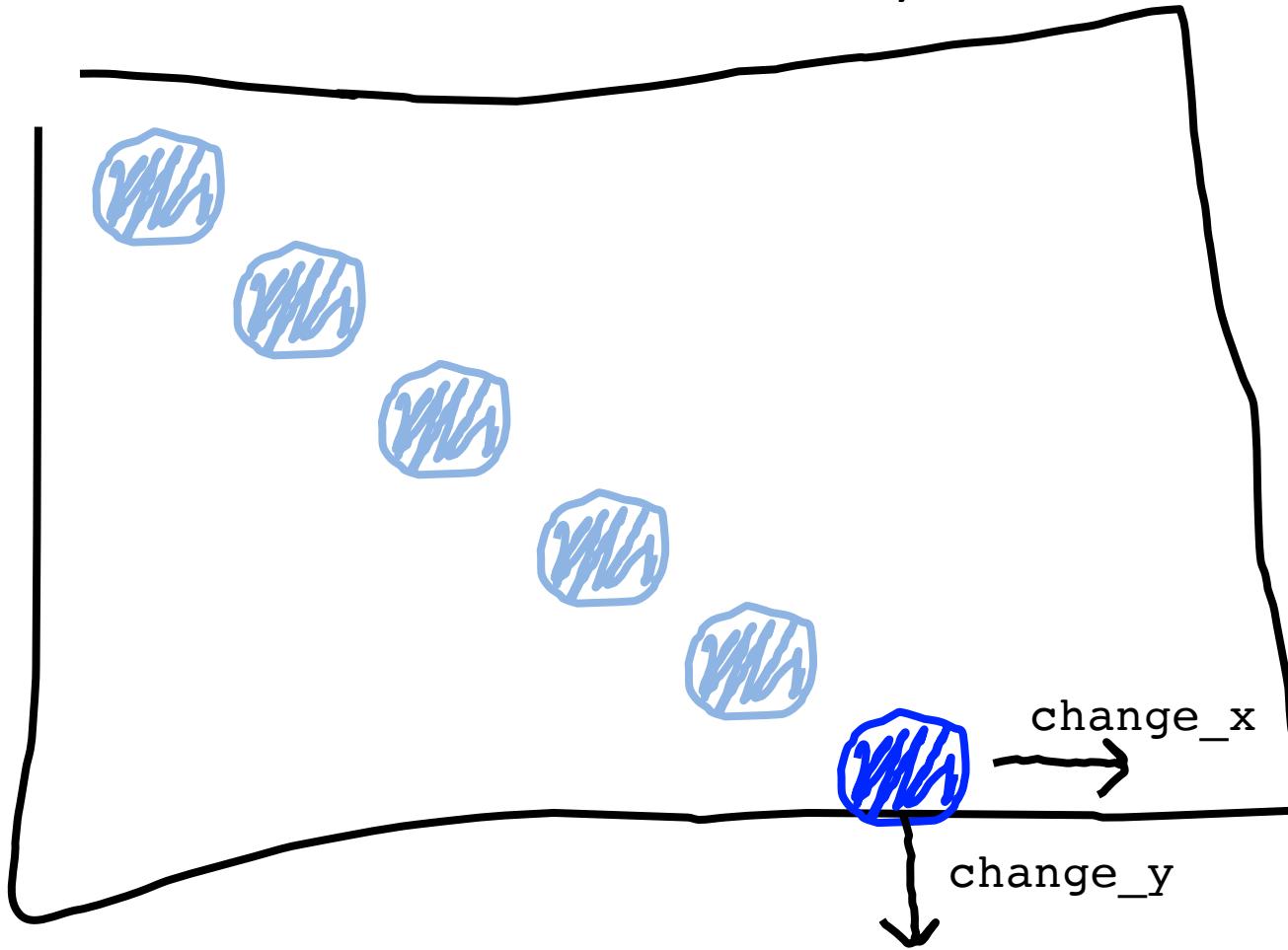
Bouncing Ball

What happens when we hit a wall?



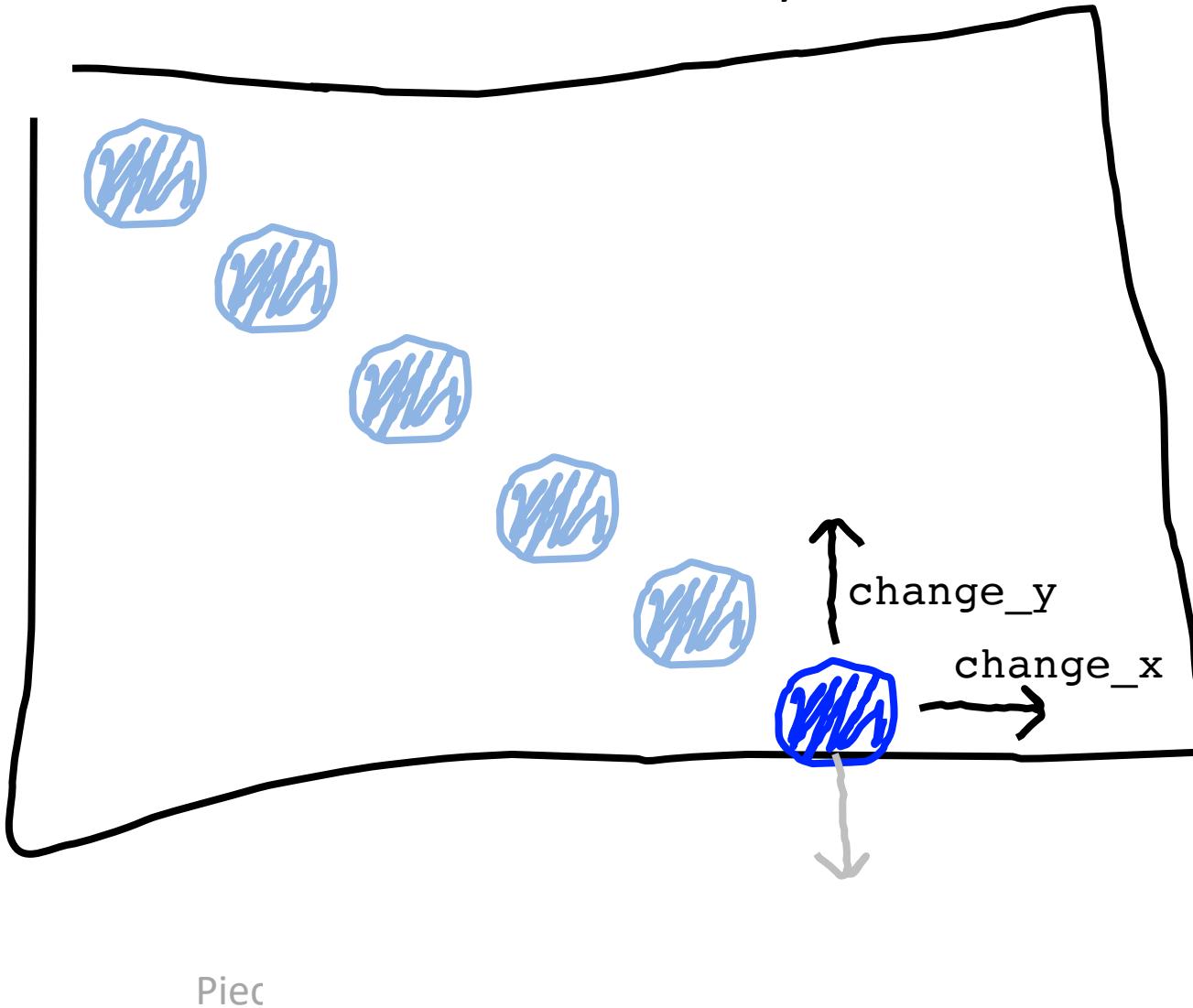
Bouncing Ball

We have this velocity



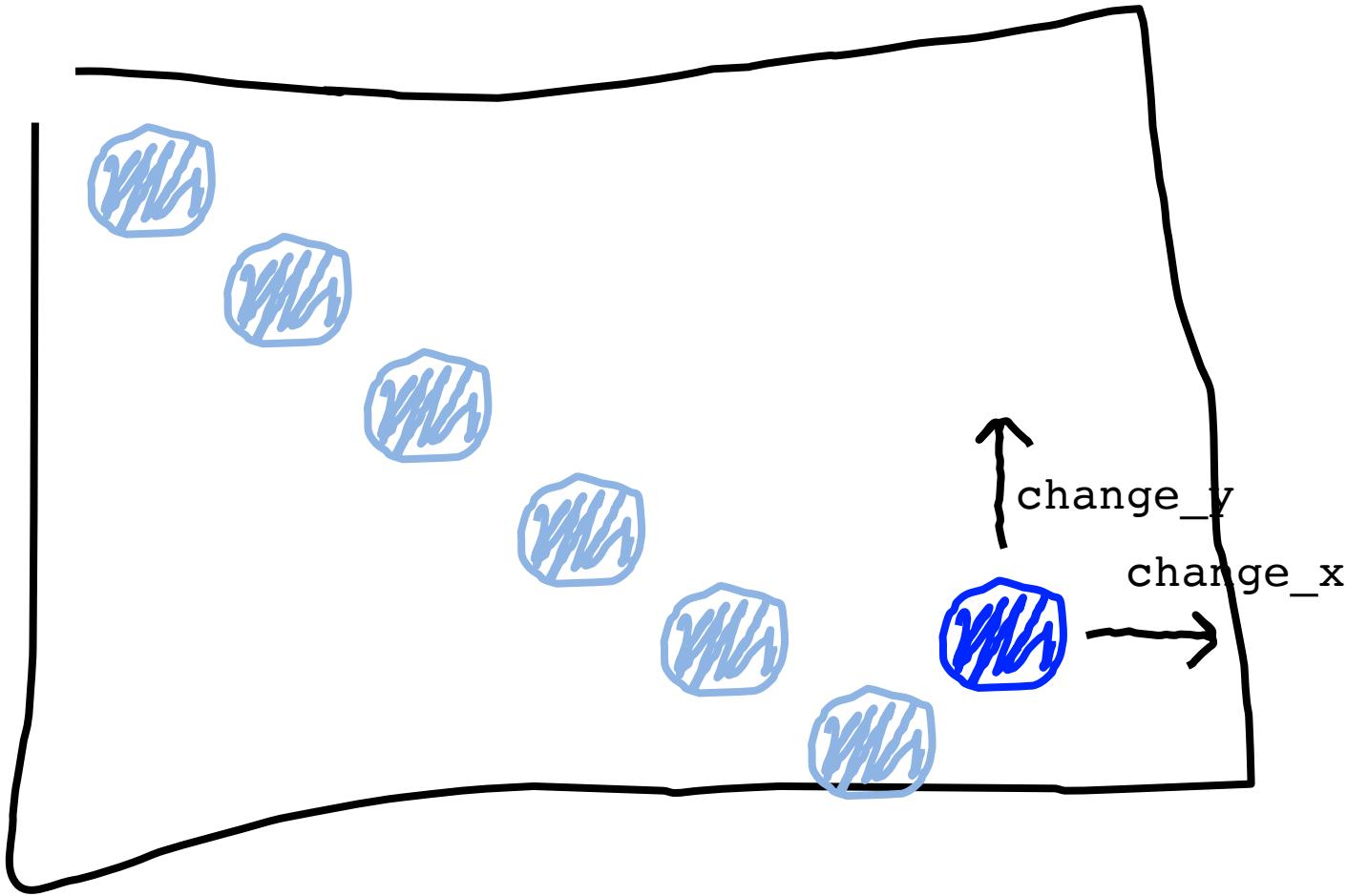
Bouncing Ball

Our new velocity



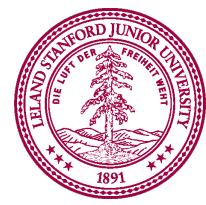
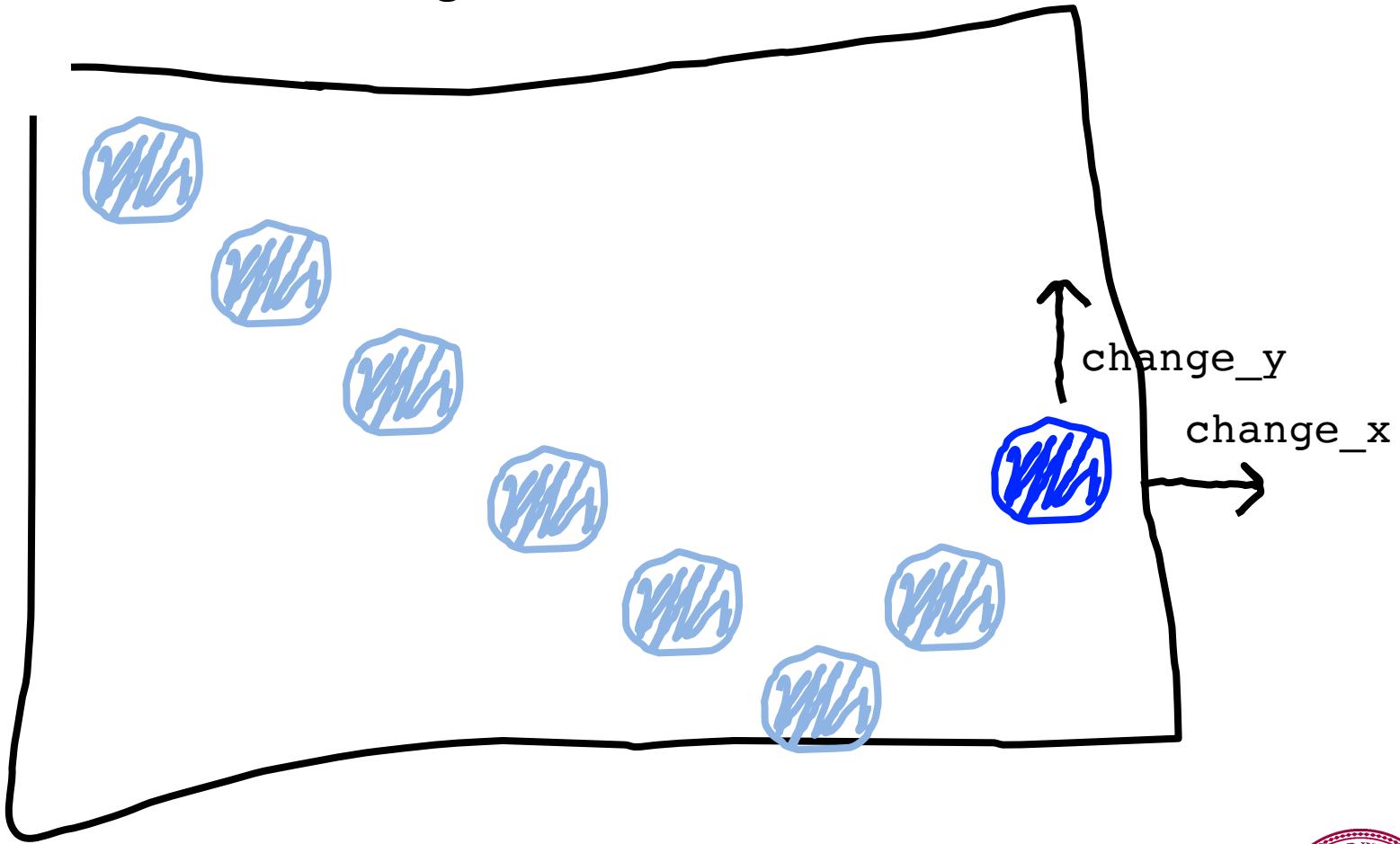
Bouncing Ball

Seventh heartbeat



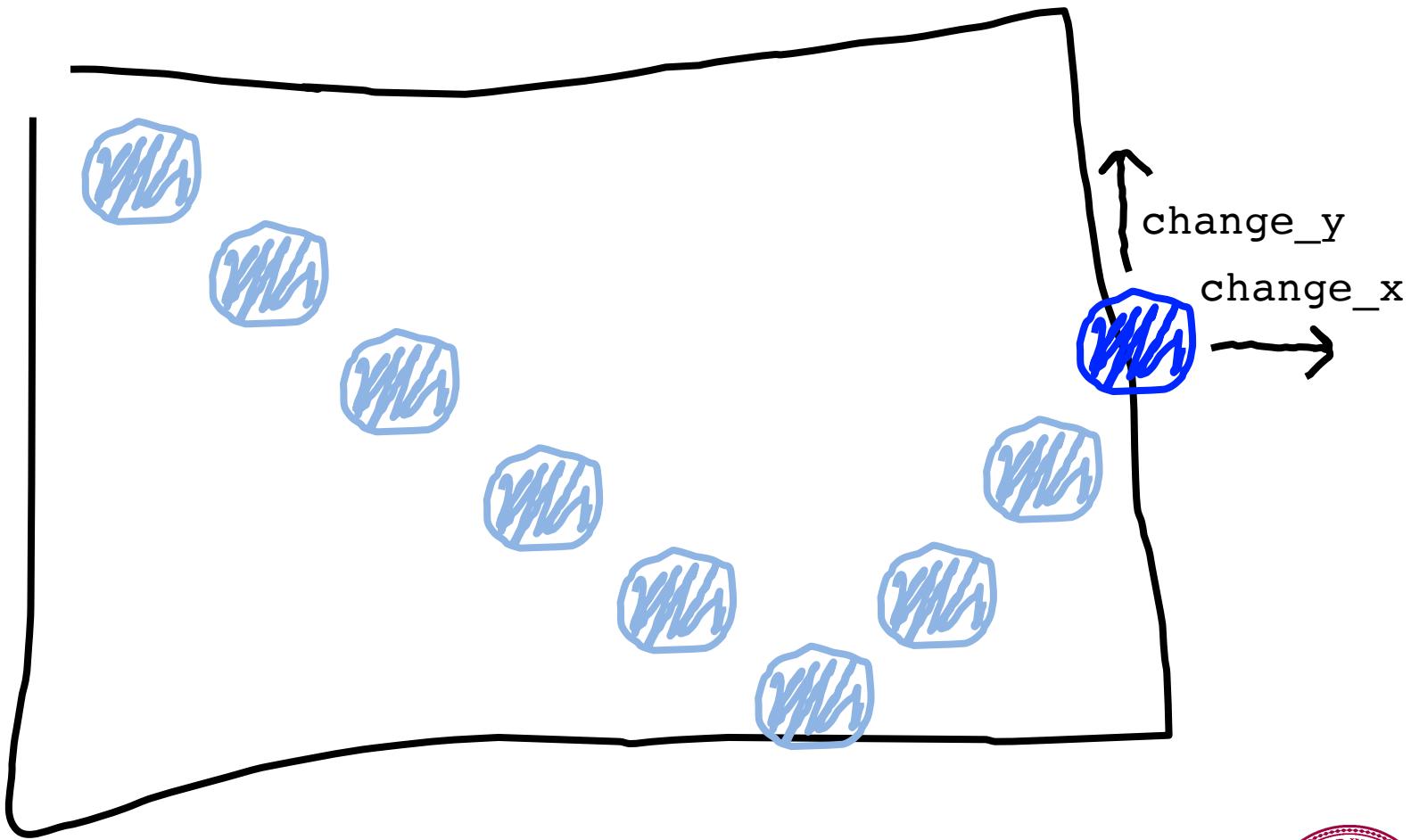
Bouncing Ball

Eighth heartbeat



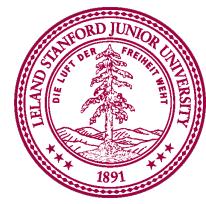
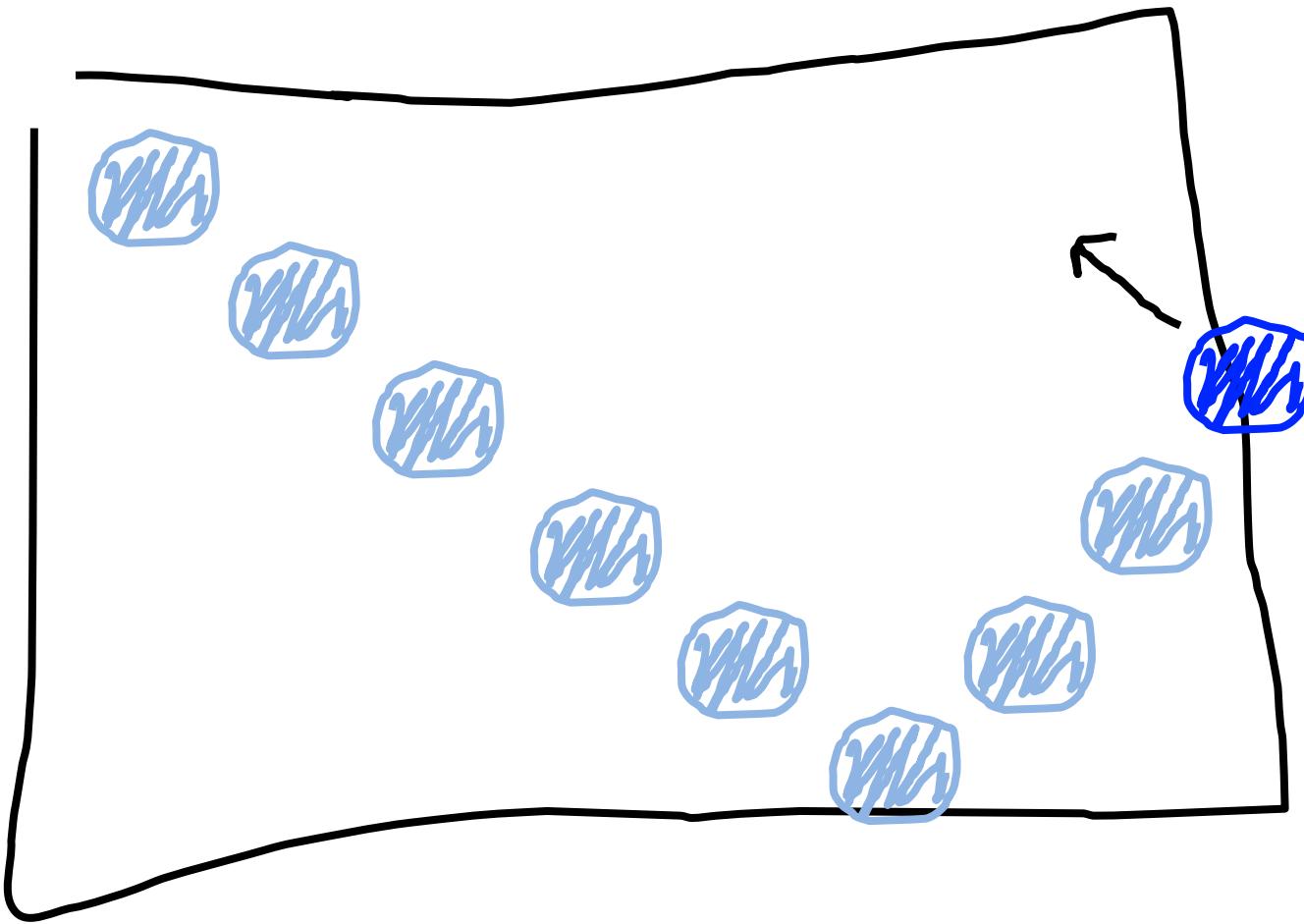
Bouncing Ball

Ninth heartbeat



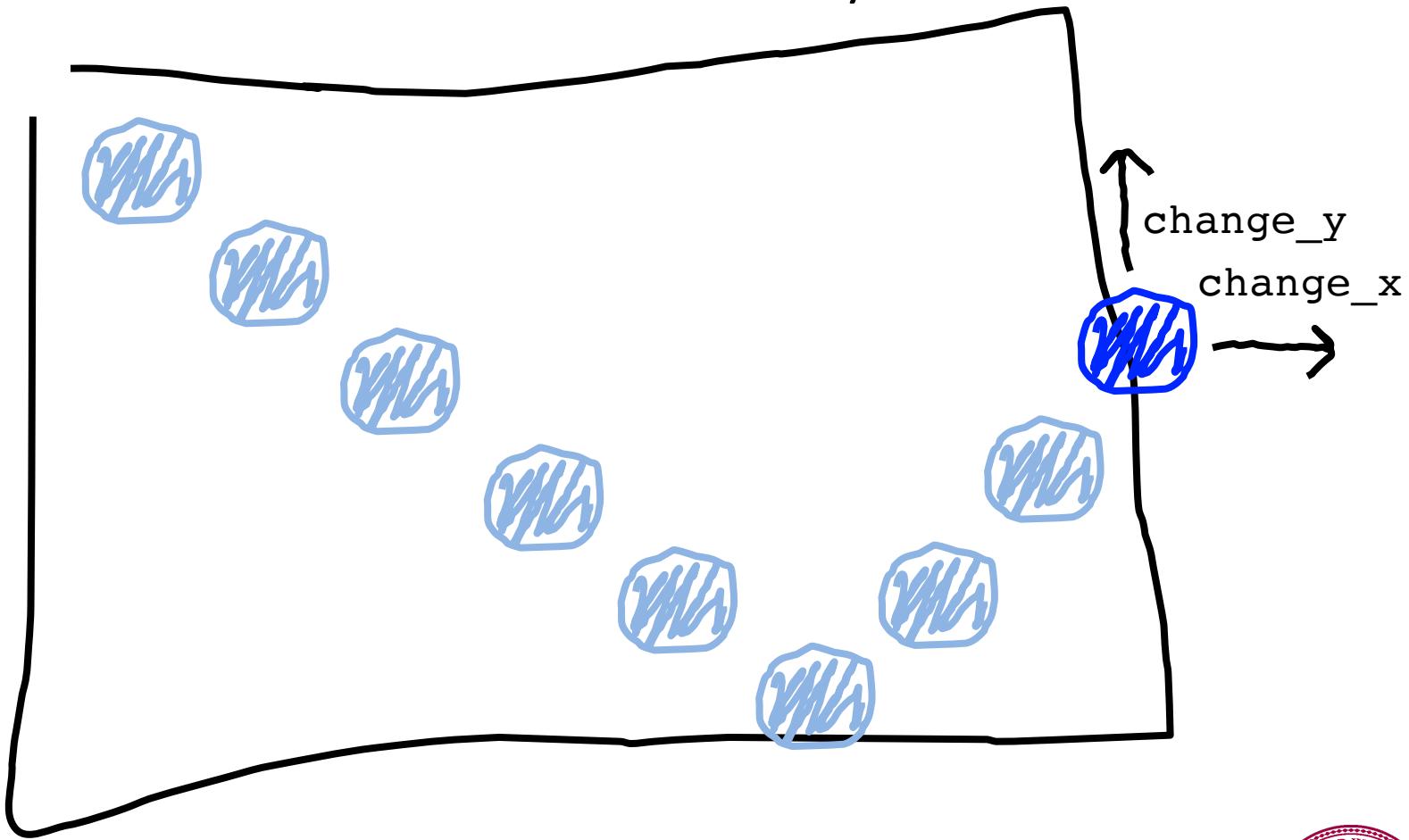
Bouncing Ball

We want this!



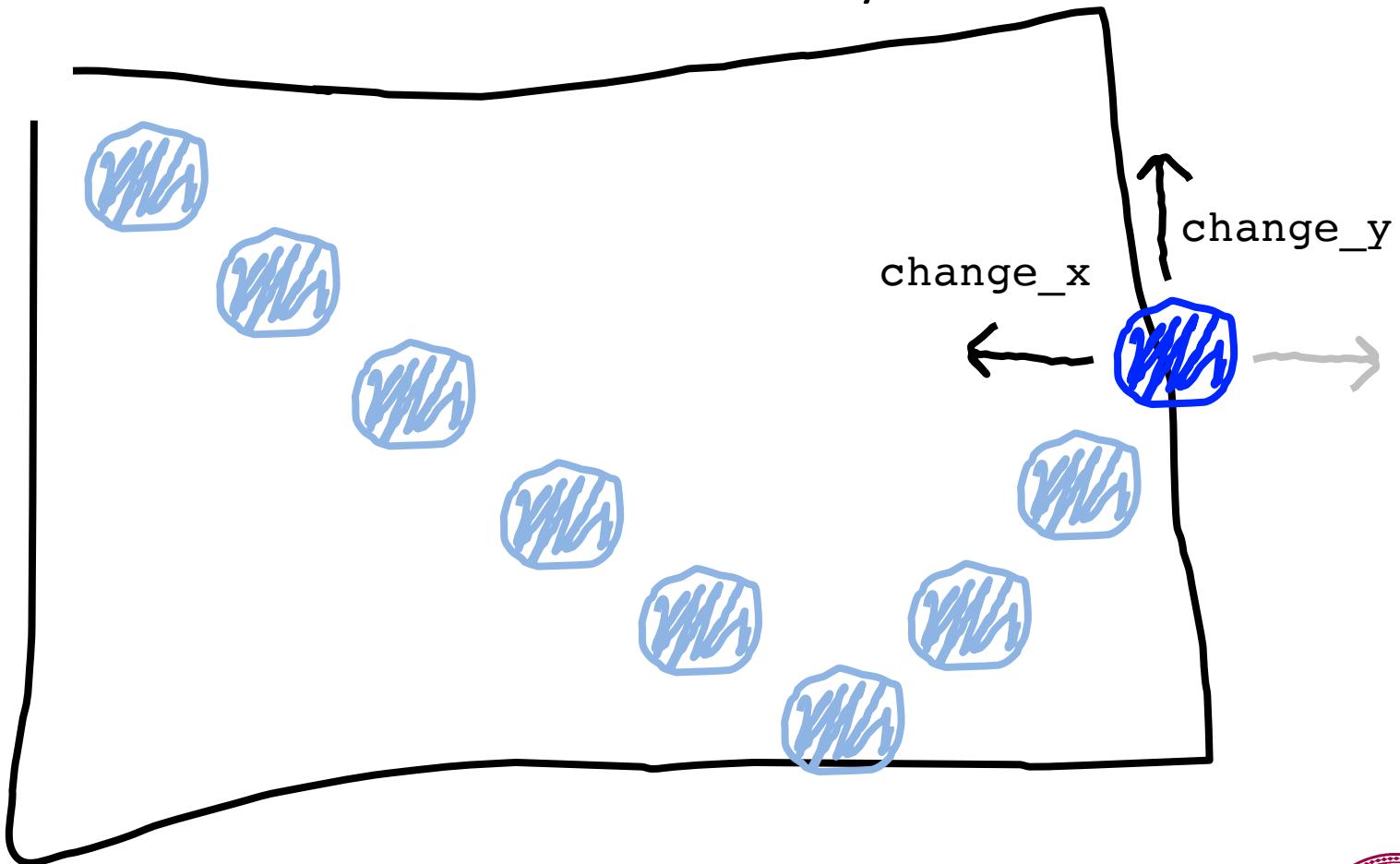
Bouncing Ball

This was our old velocity



Bouncing Ball

This is our new velocity

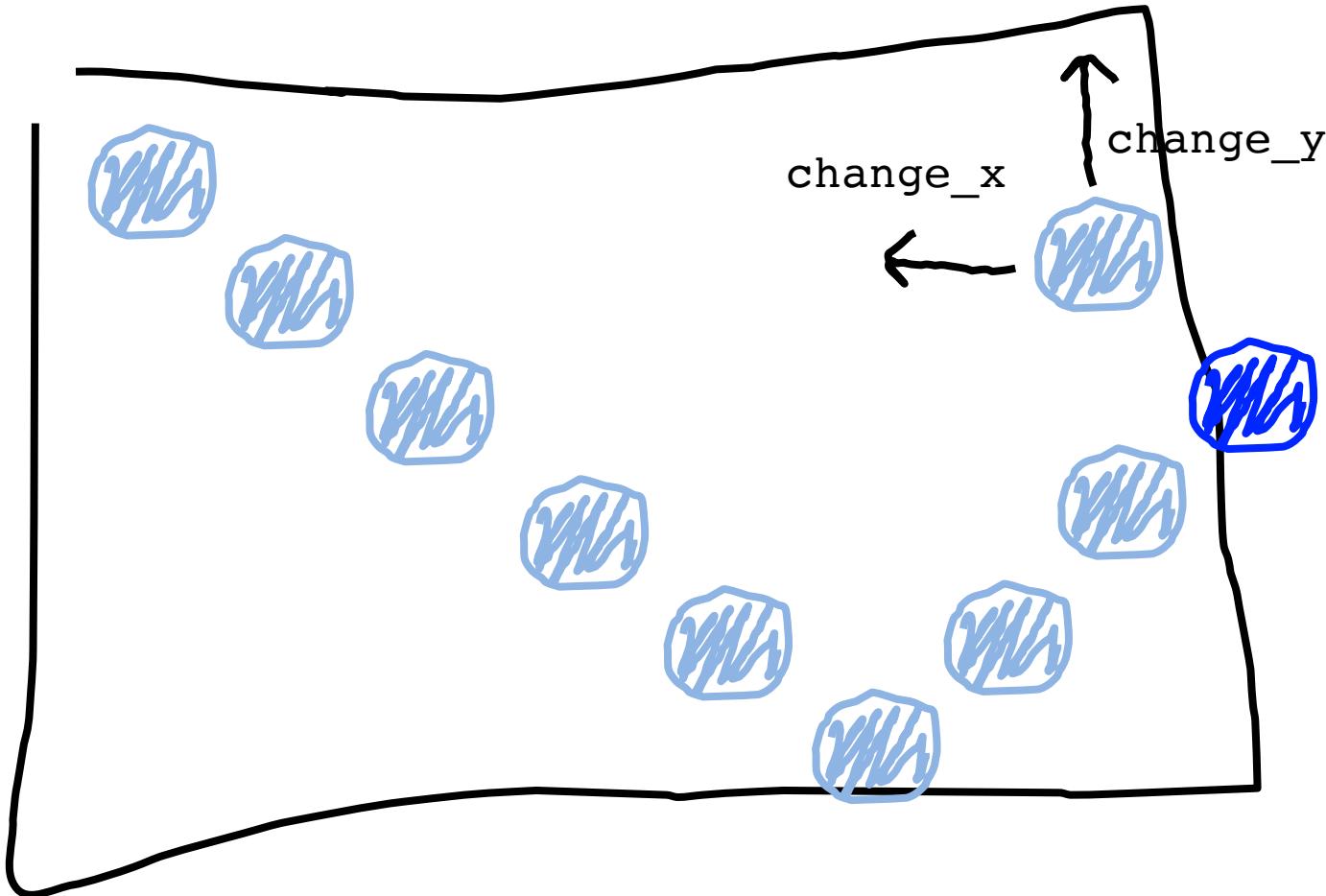


When reflecting horizontally: $\text{change}_x = -\text{change}_x$



Bouncing Ball

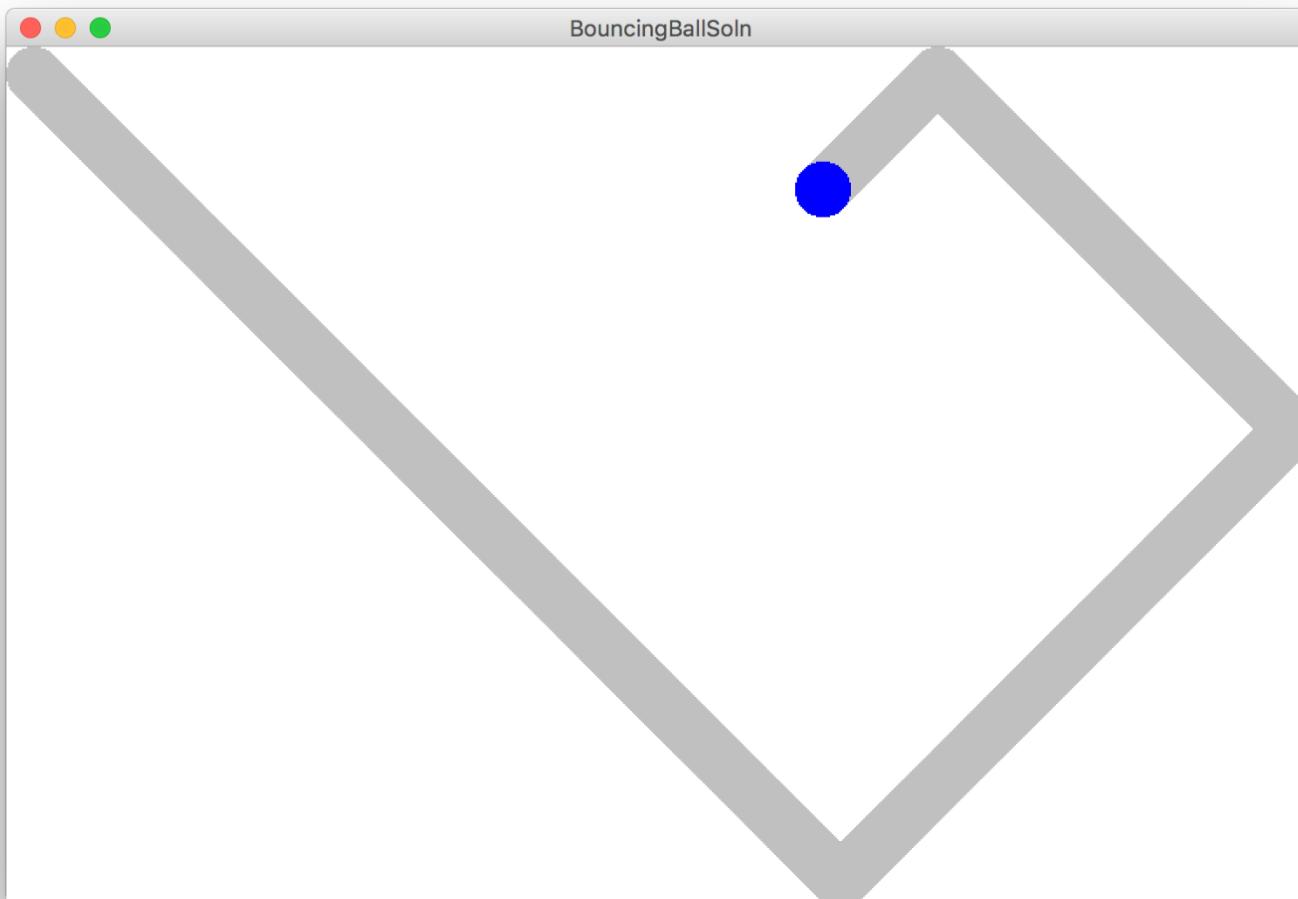
Tenth heartbeat



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



Bouncing Ball



Hold up!

```
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*



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.
- Document Outline:** On the left, it lists sections: Apollo 11 (selected), Summary, The Spacecraft, Design, Command module, Service module, The People, Neil Armstrong, Buzz Aldrin, Mission Highlights, The Launch, The Landing, and Return Trip.
- Main 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**

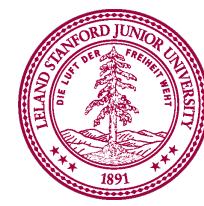
The Apollo 11 mission had three spacecraft: the Command Module Columbia, a Service Module, and the Lunar Module Eagle. Columbia was the only part of the spacecraft to return to Earth.
 - Design**

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 propulsion system—the rocket engine that put the spacecraft into lunar orbit and later boosted it back
- Right Side:** Includes a sidebar with 'Comments' and 'Share' buttons.

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



```
def main():
```

```
    canvas = Canvas(...)
```

```
    make_ball(canvas)
```

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

```
    canvas.create_rectangle( ... )
```

stack

heap

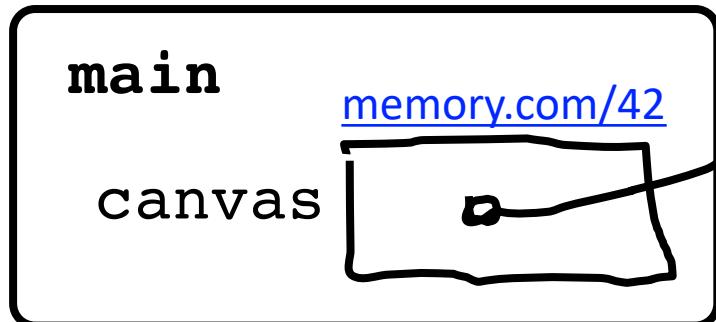
```
main
```



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

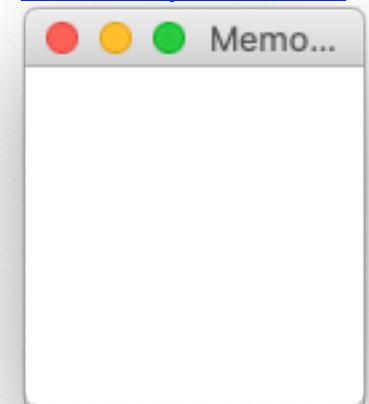
```
def make_ball(canvas):
    canvas.create_oval( ... )
```

stack



heap

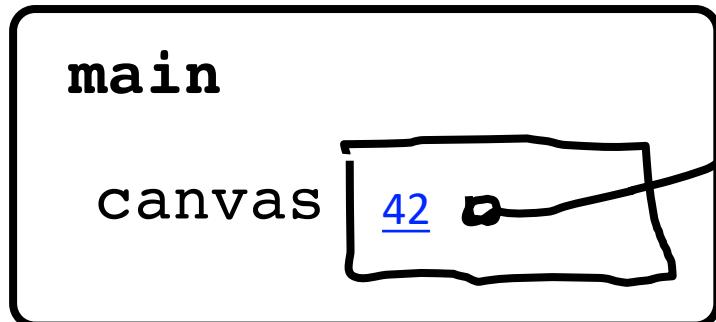
[memory.com/42](#)



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

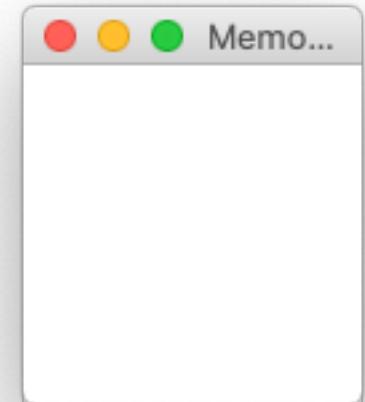
```
def make_ball(canvas):
    canvas.create_oval( ... )
```

stack



heap

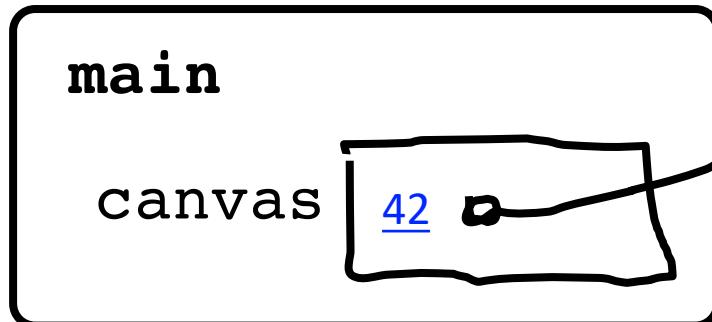
memory.com/42



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

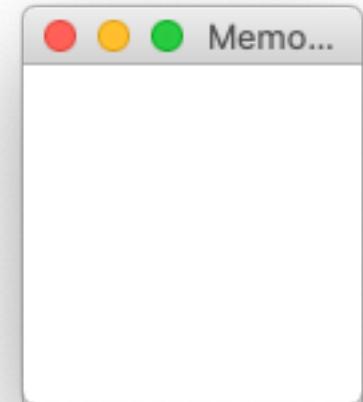
```
def make_ball(canvas):
    canvas.create_oval( ... )
```

stack

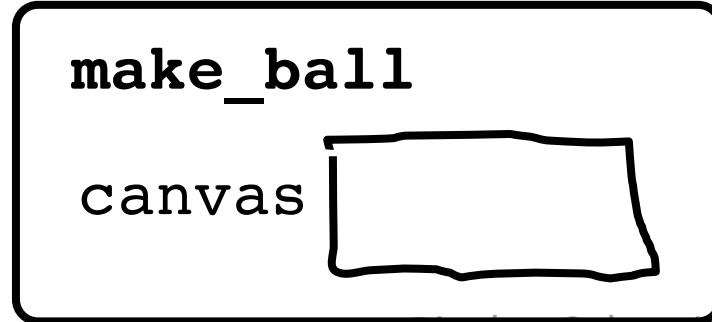


heap

memory.com/42



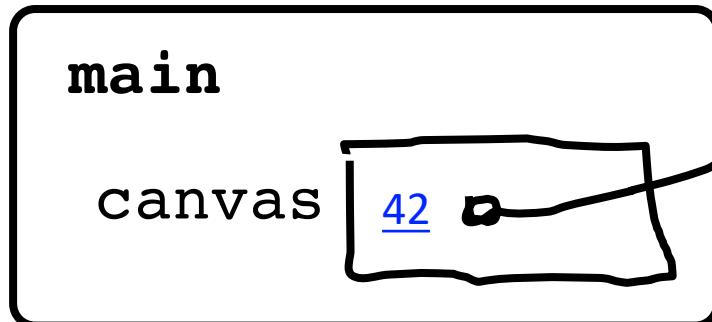
make_ball



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

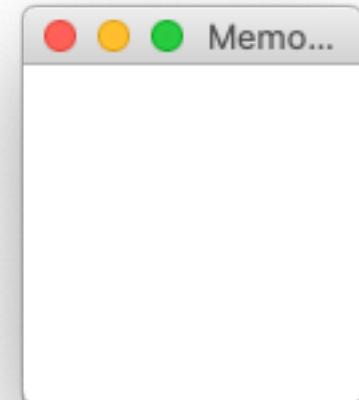
```
def make_ball(canvas):
    canvas.create_oval( ... )
```

stack

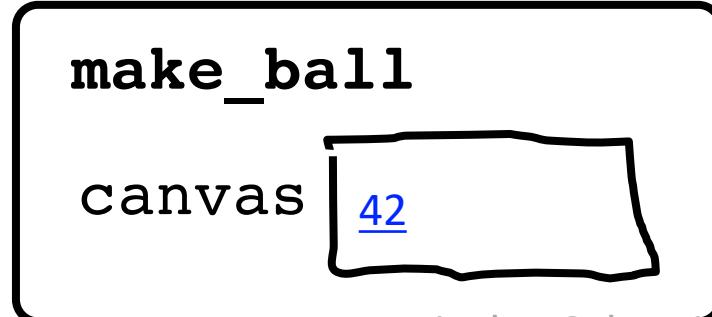


heap

memory.com/42



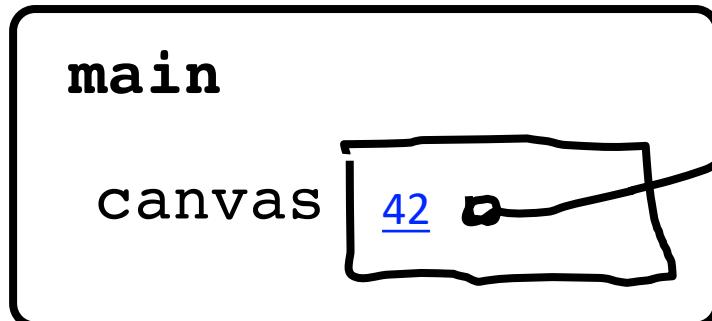
make_ball



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

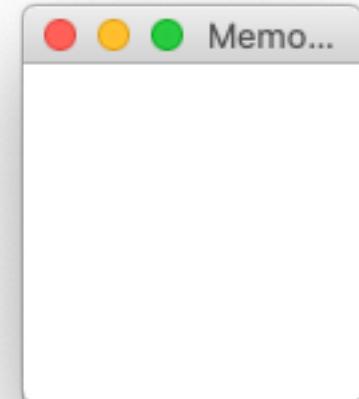
```
def make_ball(canvas):
    canvas.create_oval( ... )
```

stack

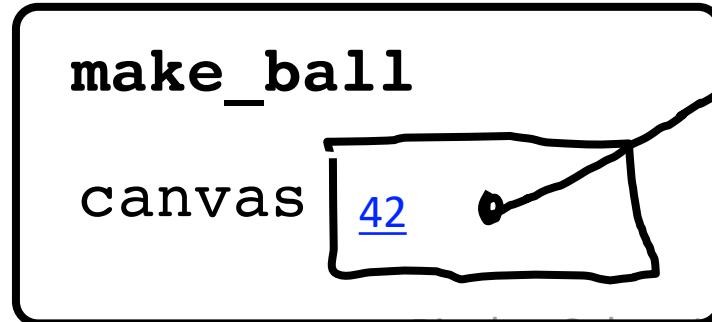


heap

memory.com/42



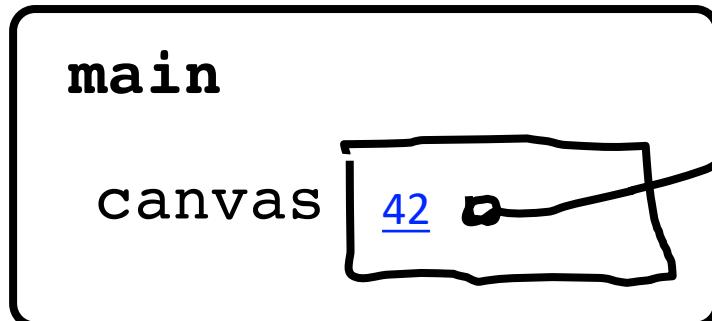
make_ball



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

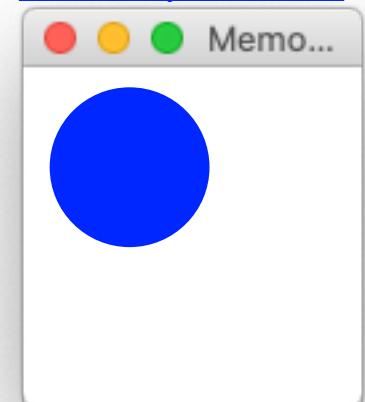
```
def make_ball(canvas):
    canvas.create_oval( ... )
```

stack

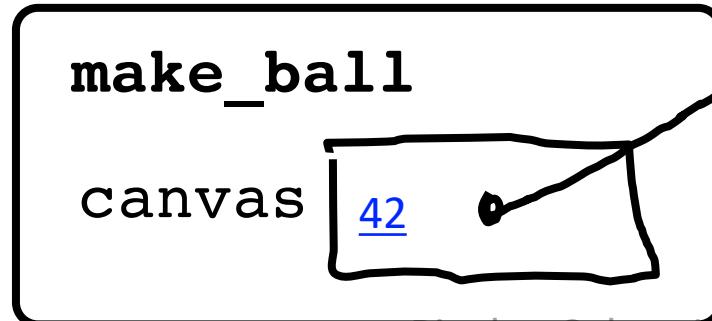


heap

memory.com/42



make_ball

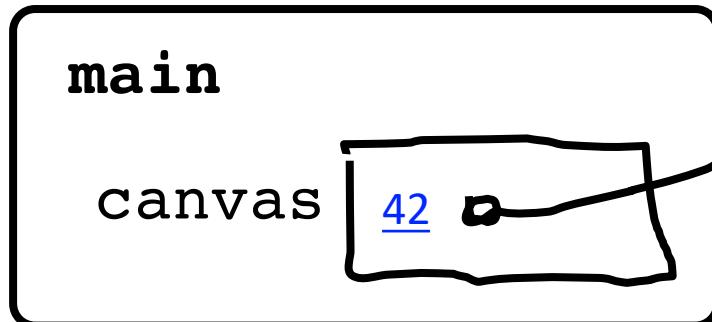


```
def main():
    canvas = Canvas(...)
    make_ball(canvas)
```

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

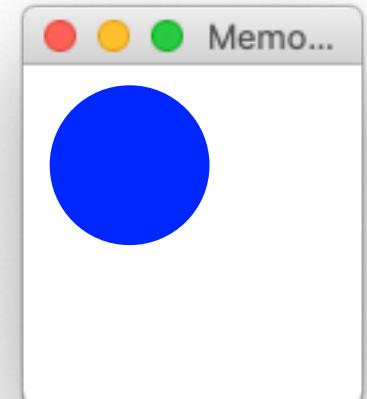


stack



heap

memory.com/42



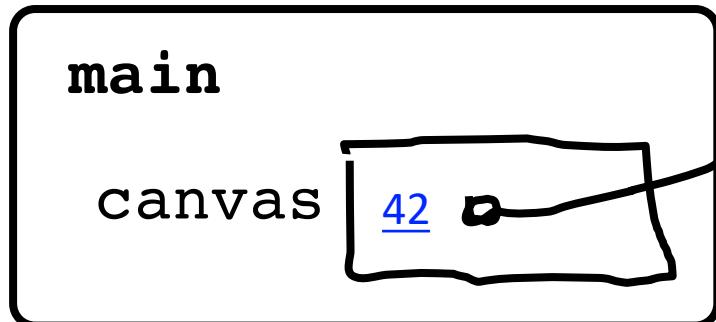
make_ball



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

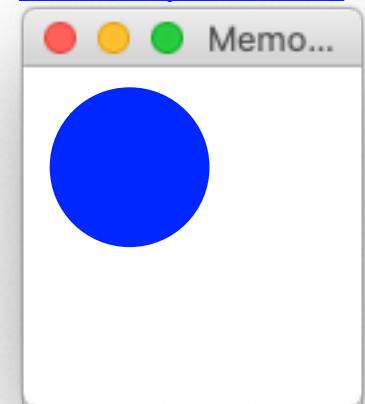
```
def make_ball(canvas):
    canvas.create_oval( ... )
```

stack



heap

memory.com/42





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

That allows functions to
modify the variable



Learning Goals

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



Special Graphics Functions

```
# move shape to some new coordinates
```

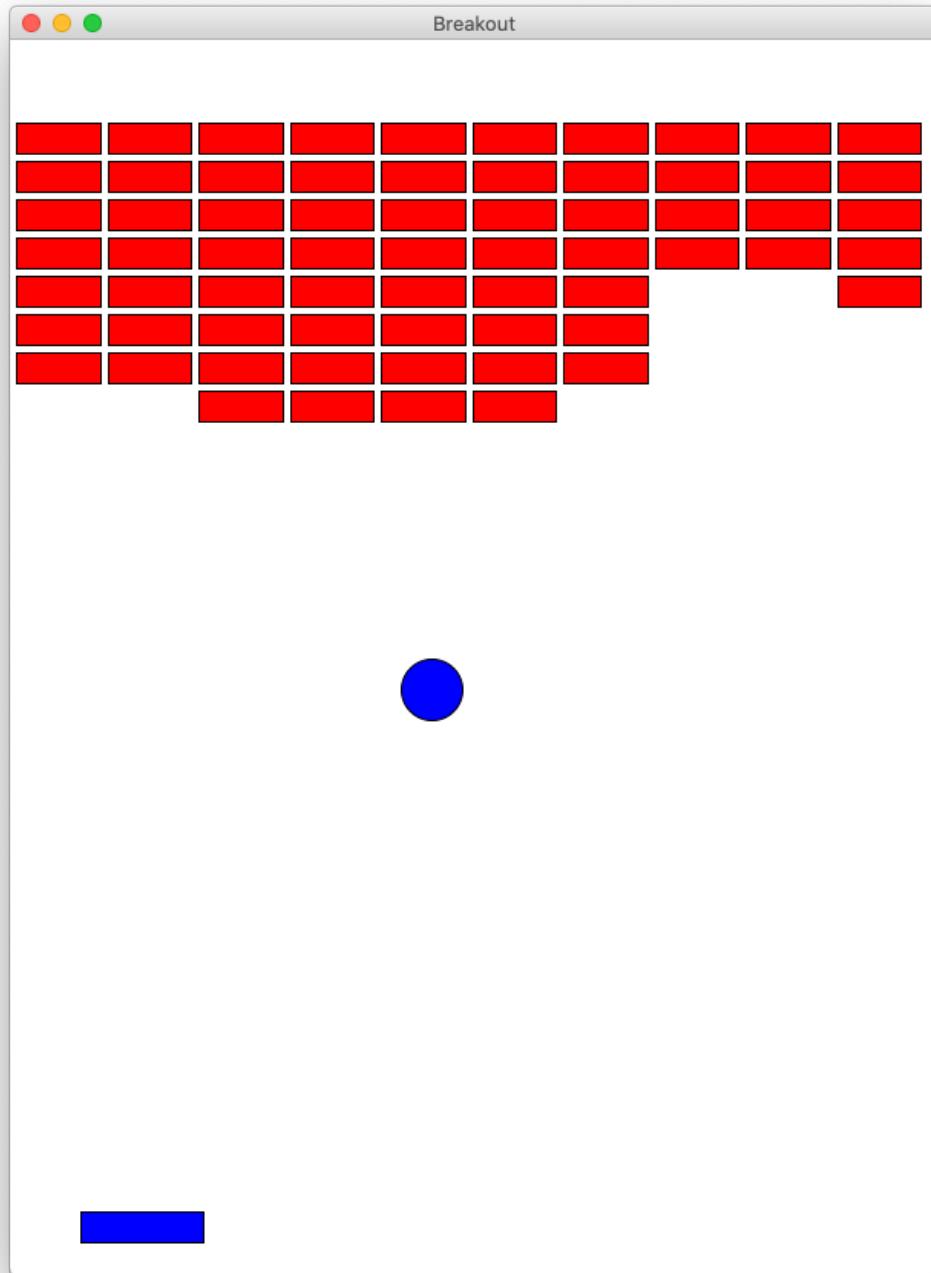
```
canvas.moveto(shape, new_x, new_y)
```

```
# move shape by a given change_x and change_y
```

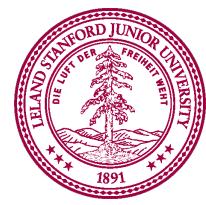
```
canvas.move(shape, change_x, change_y)
```

Come back tomorrow to learn about lists!

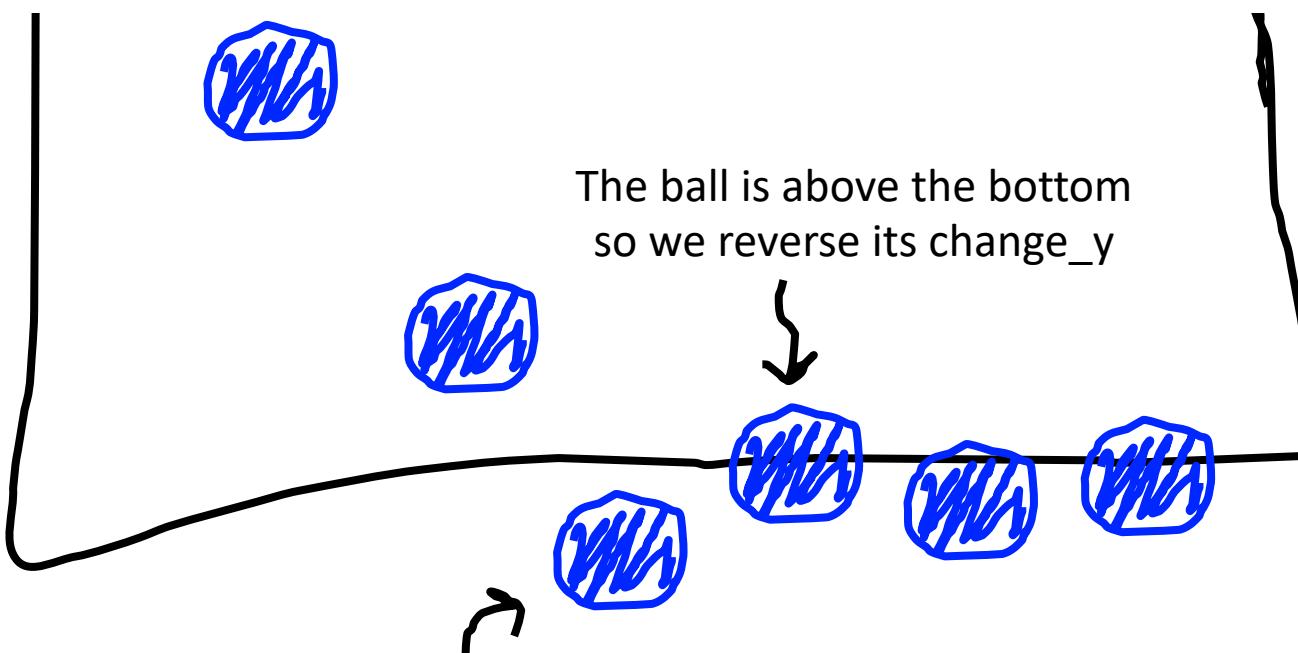




Piech + Sahami, CS106A, Stanford University



A Sticky Situation



The ball is bellow the bottom
so we reverse its change_y

