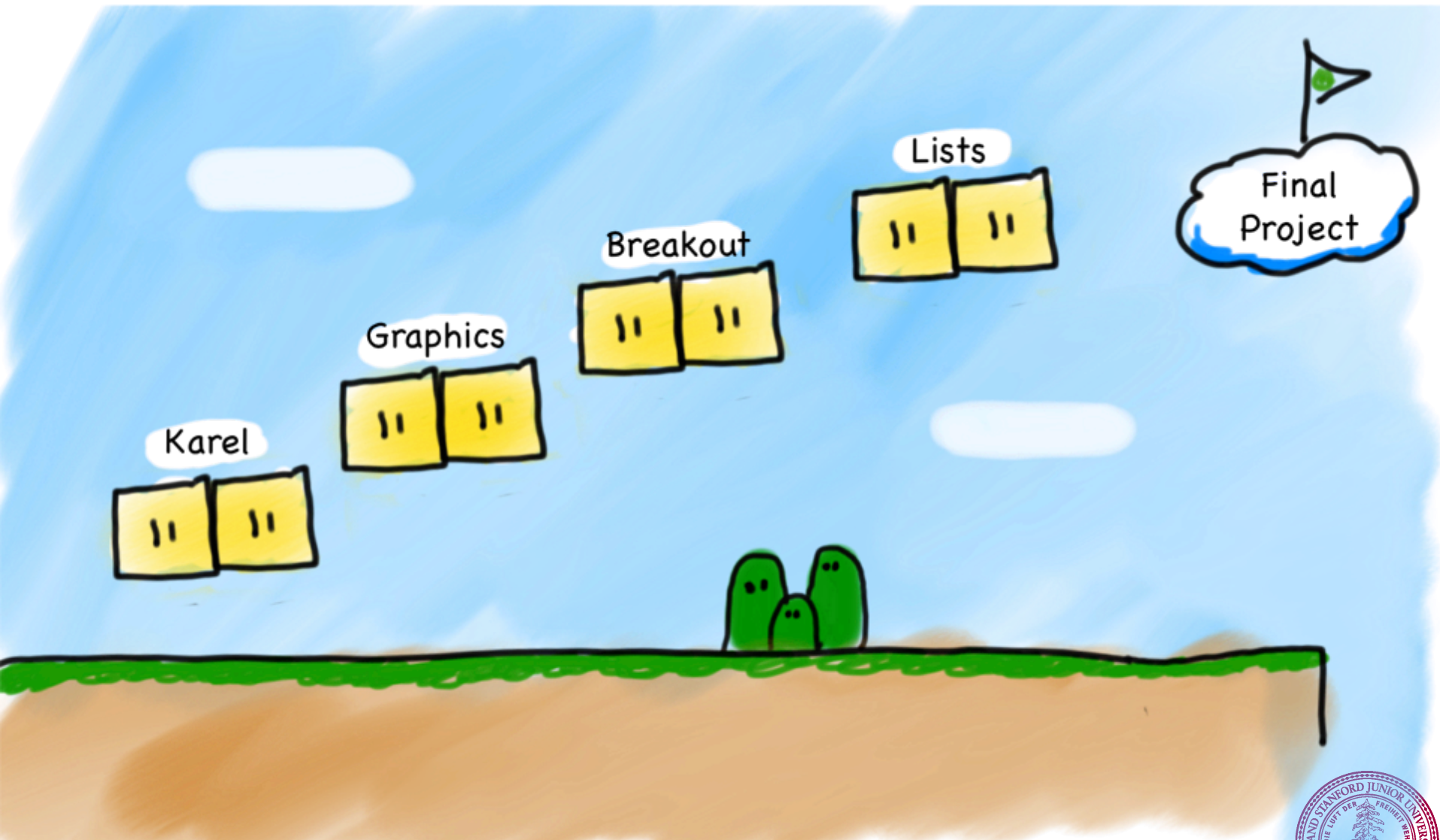


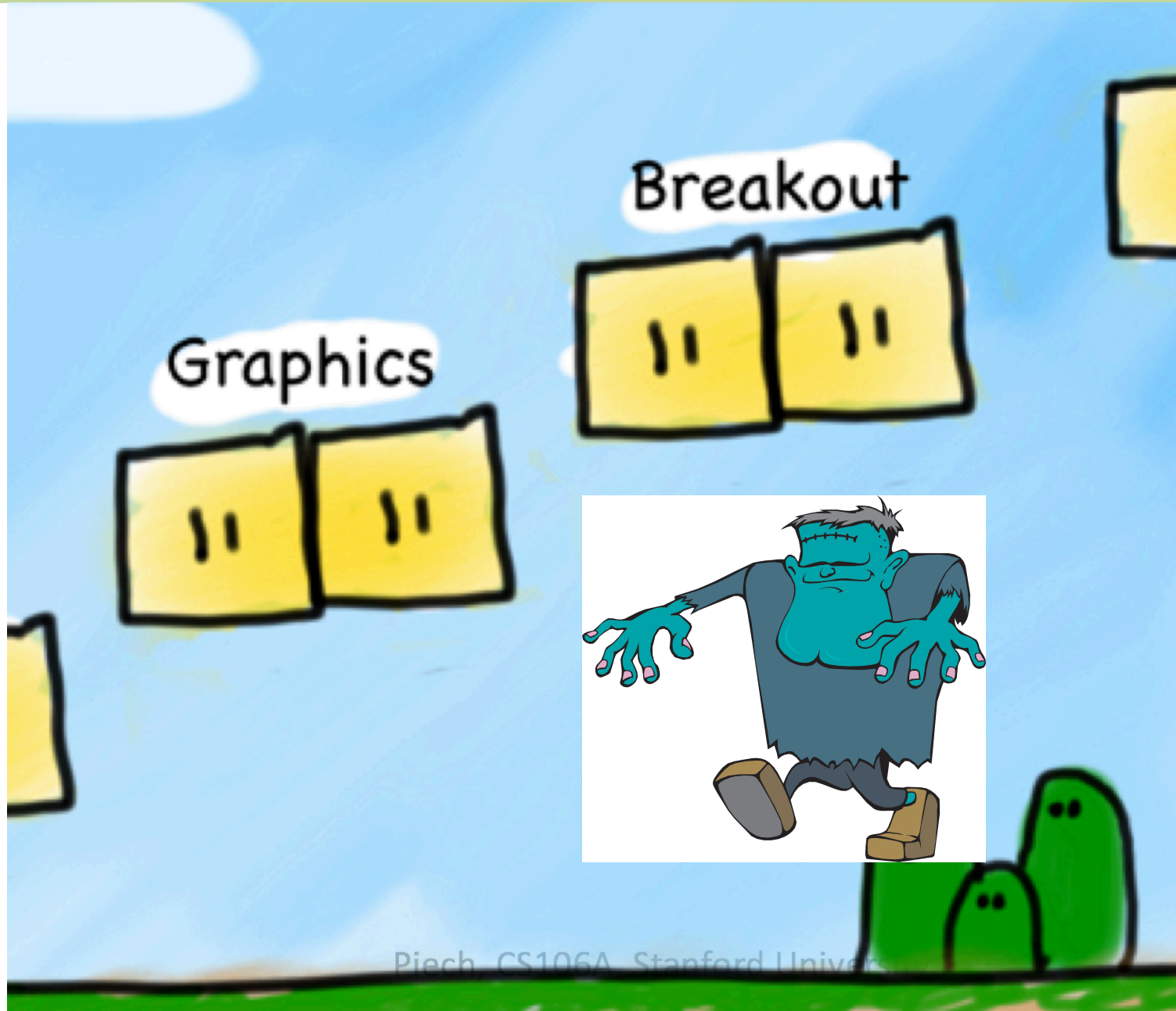


Animation

Our story so far...



Our story so far...



Learning Goals

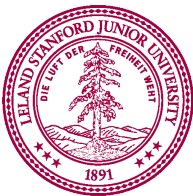
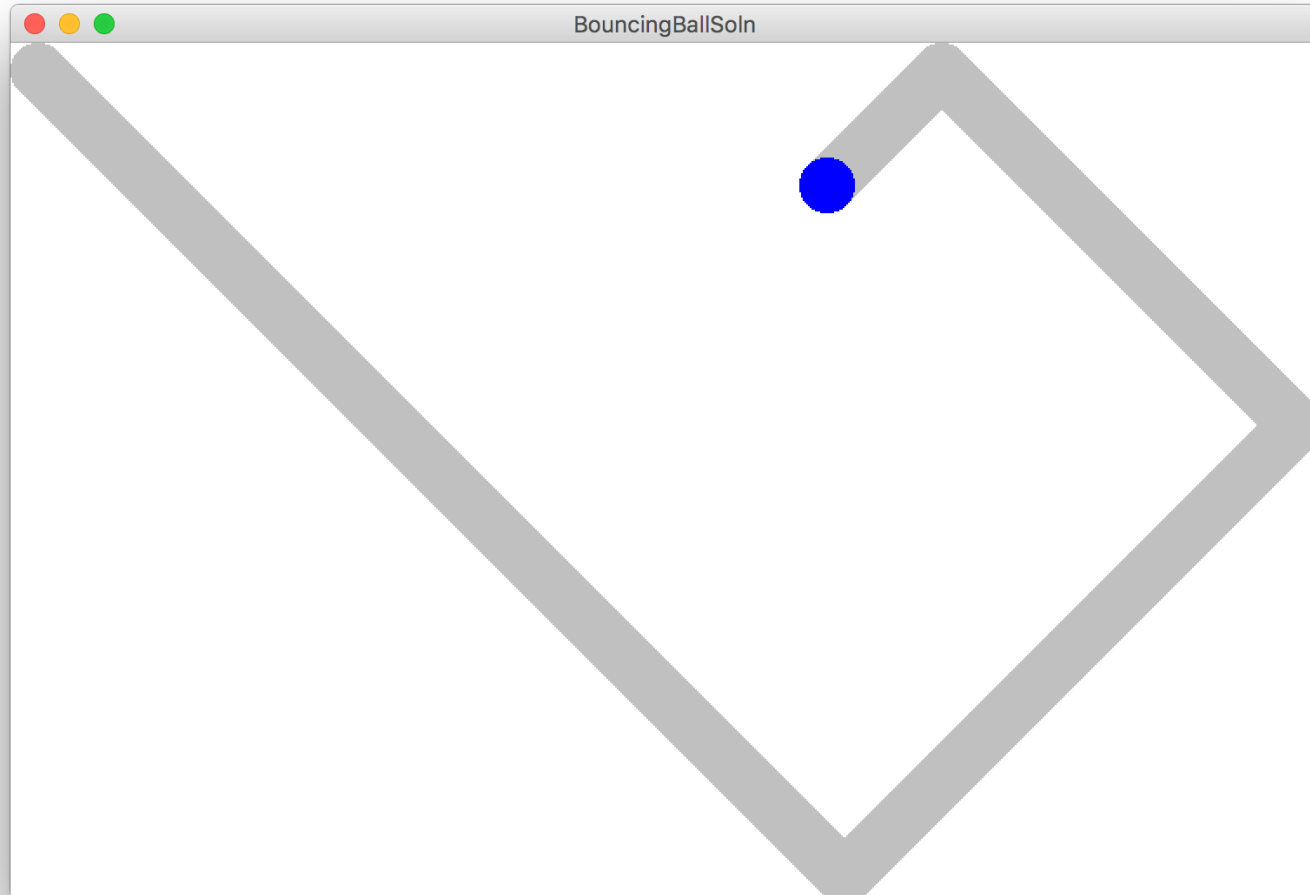
1. Write animated programs
2. Center an object



© Disney/Pixar



You will be able to write Bouncing Ball



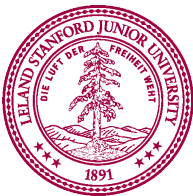
Learning Goals For Me

1. Speak slower

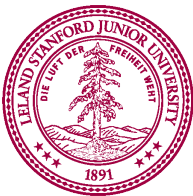
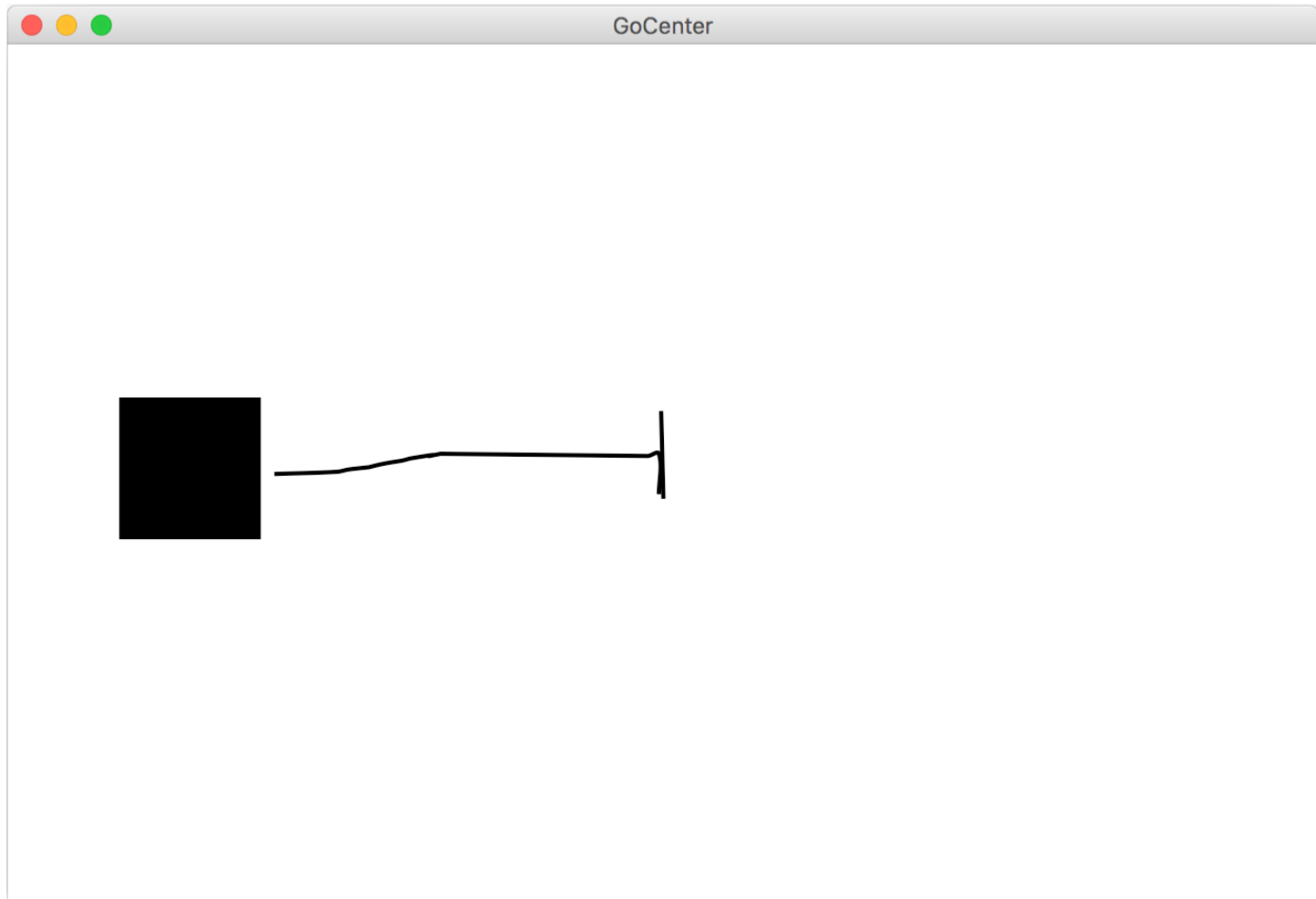


But First!

```
private void run() {  
    int x = 6 - 4 + 7 * 3;  
    println(x);  
  
    int y = (6 + 4 + 7) * 3;  
    println(y);  
  
    int z = 6 / 2 * 3;  
    println(z);  
  
}
```

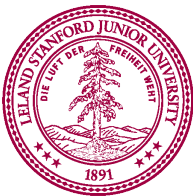


Move GRect



Animation Loop

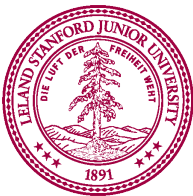
```
private void run() {  
    // setup  
  
    while(true) {  
        // update world  
  
        // pause  
        pause(DELAY);  
    }  
}
```



Animation Loop

```
private void run() {  
    // setup  
  
    while(true) {  
        // update world  
  
        // pause  
        pause(DELAY);  
    }  
}
```

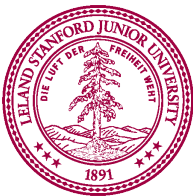
Make all the variables you need. Add graphics to the screen.



Animation Loop

```
private void run() {  
    // setup  
    while(true) {  
        // update world  
  
        // pause  
        pause(DELAY);  
    }  
}
```

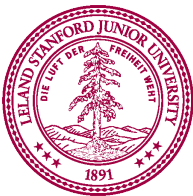
The animation loop is a repetition of heartbeats



Animation Loop

```
private void run() {  
    // setup  
  
    while(true) {  
        // update world  
        // pause  
        pause(DELAY);  
    }  
}
```

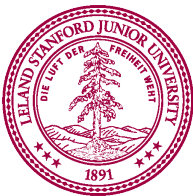
Each heart-beat, update
the world forward one
frame



Animation Loop

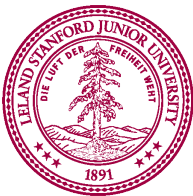
```
private void run() {  
    // setup  
  
    while(true) {  
        // update world  
  
        // pause  
        pause(DELAY);  
    }  
}
```

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

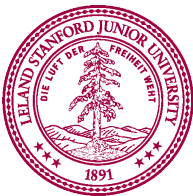
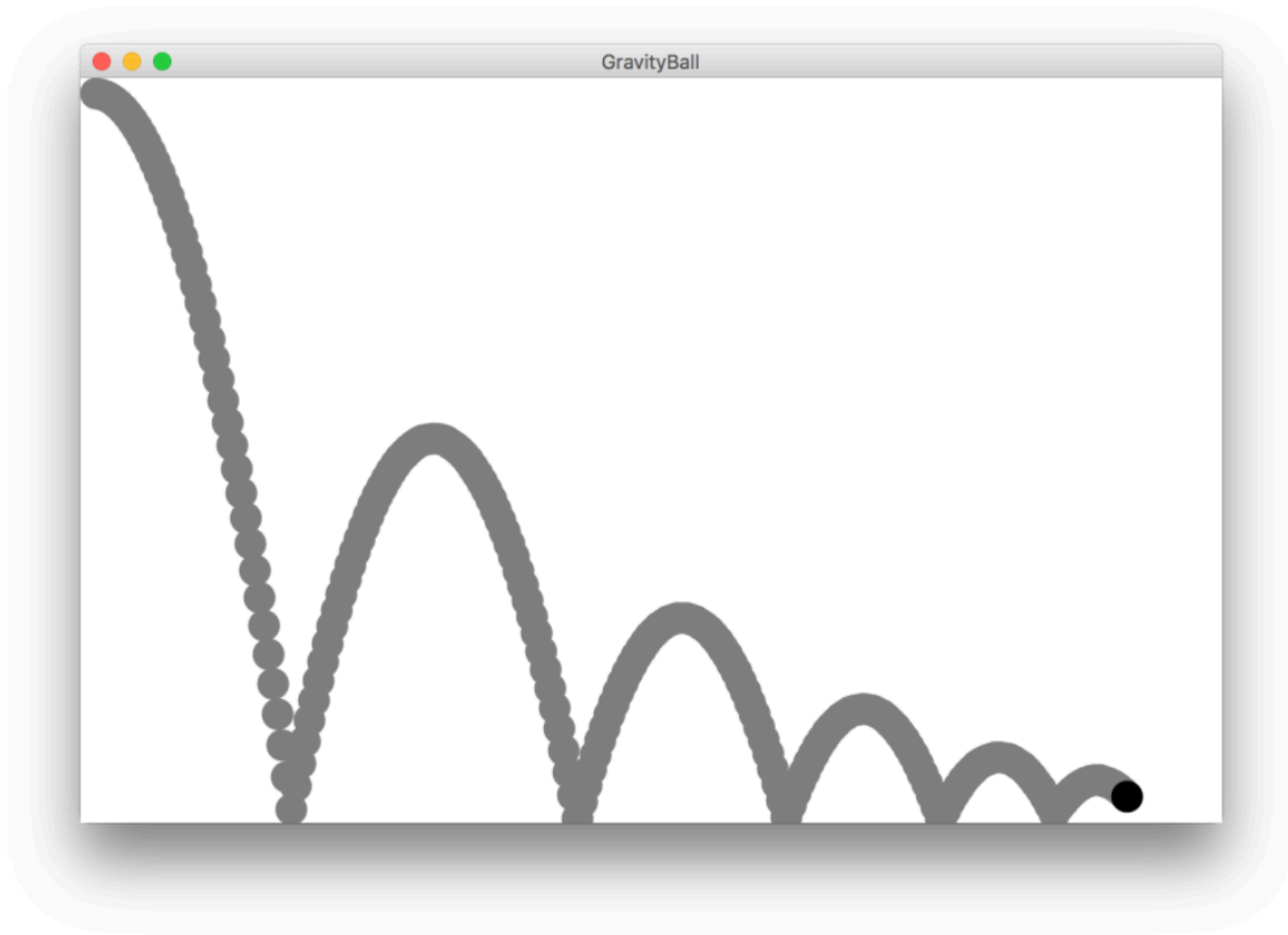


Move To Center

```
private void run() {  
    // setup  
    GRect r = new Grect(0, 250, 100, 100);  
    r.setFilled(true);  
    add(r);  
  
    while(true) {  
        // update world  
        r.move(1, 0);  
  
        // pause  
        pause(DELAY);  
    }  
}
```

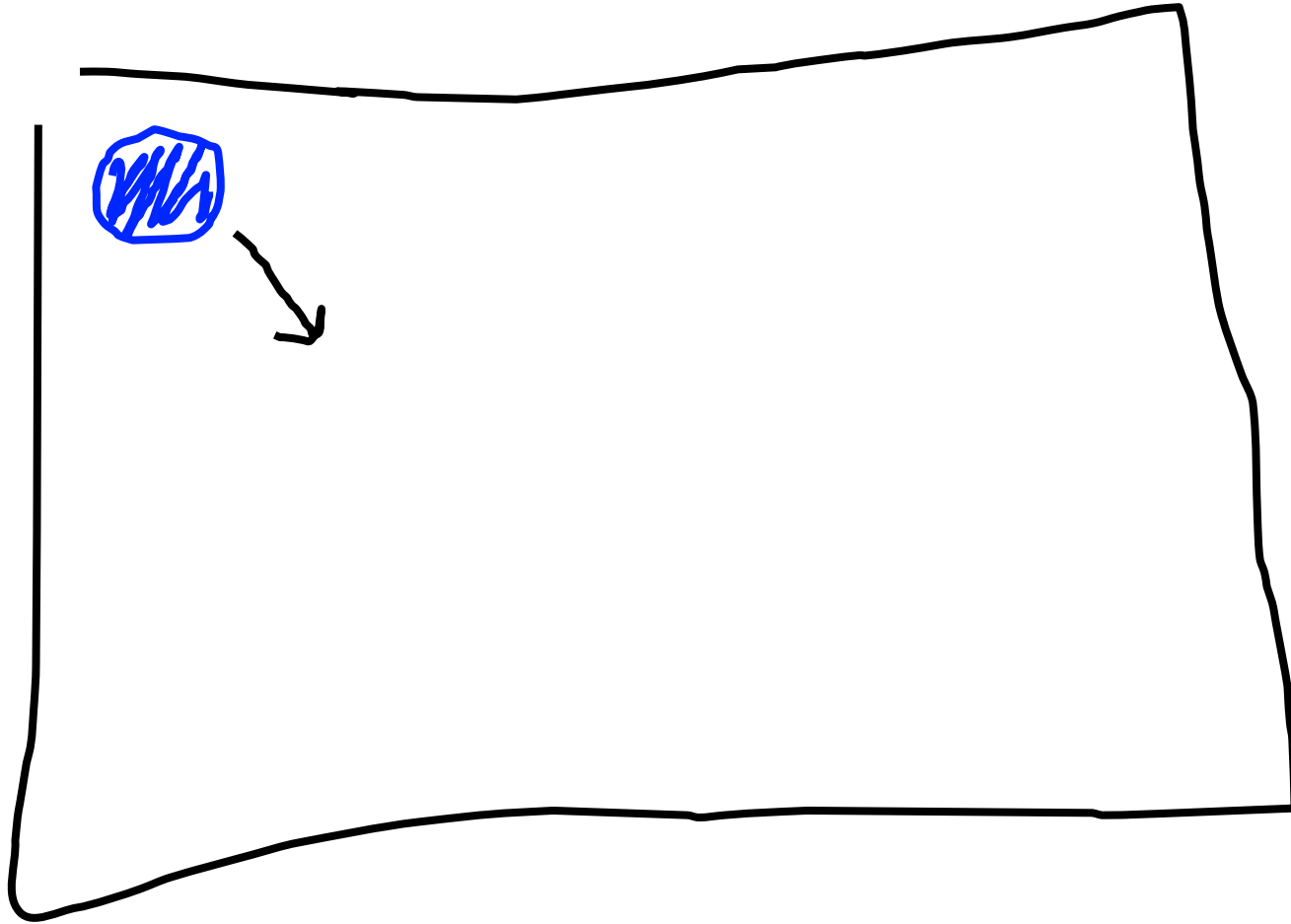


Gravity Ball

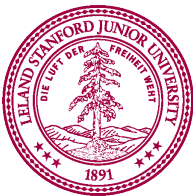


Gravity Ball

First heartbeat

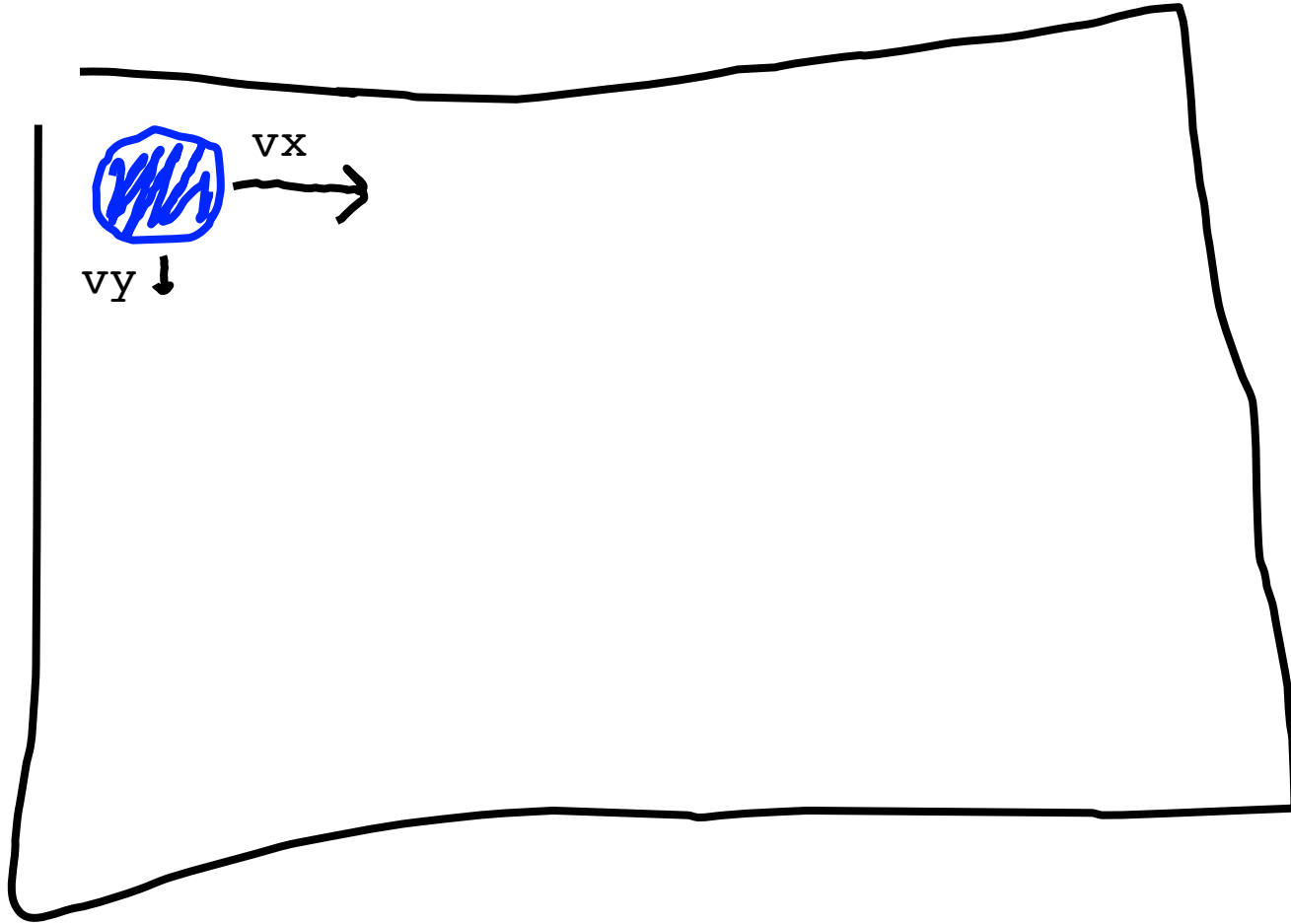


Velocity: how much the ball position changes each heartbeat

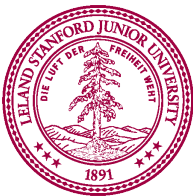


Gravity Ball

First heartbeat

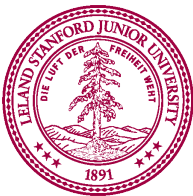
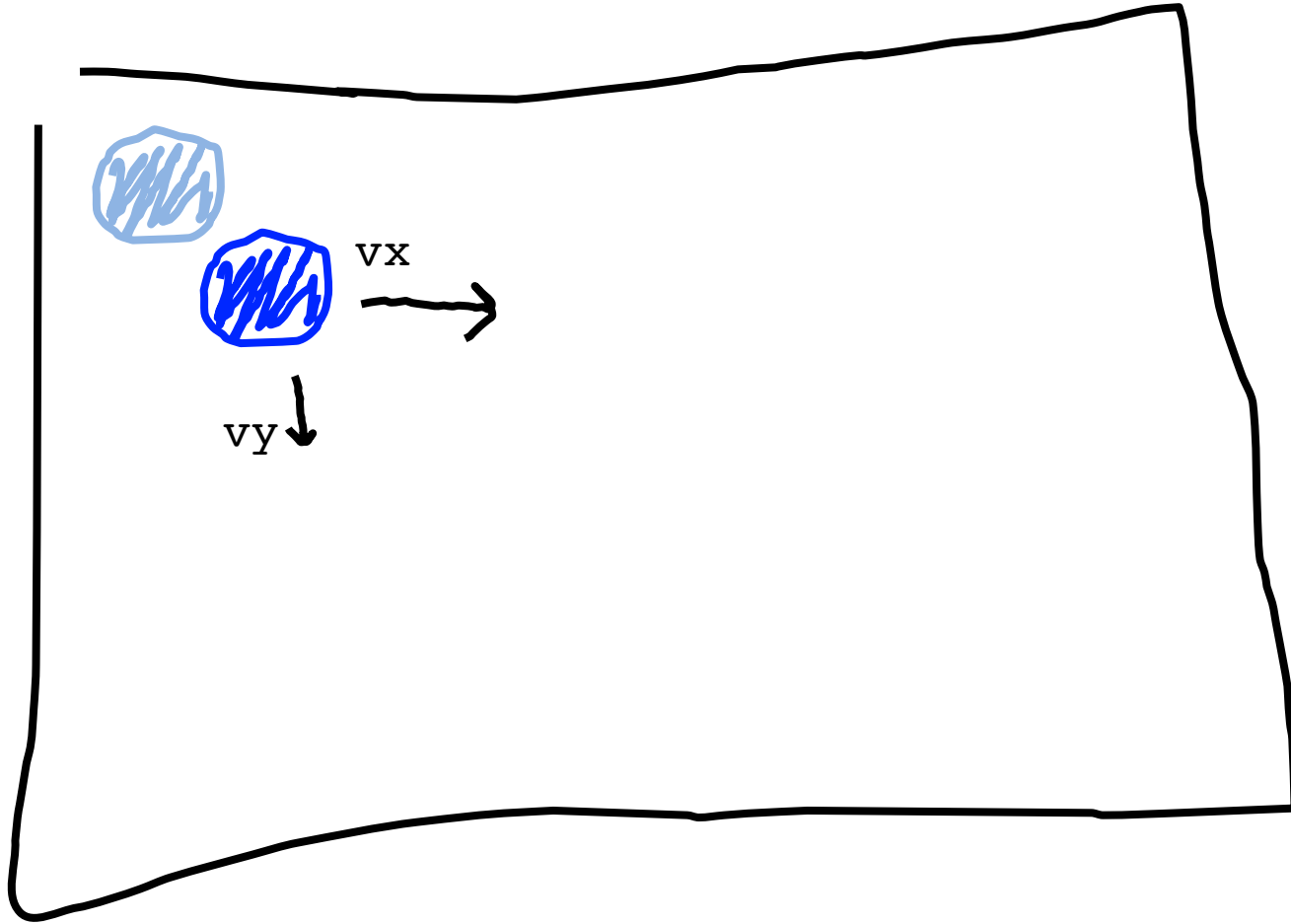


The `GOval` **move** method takes in a change in x and a change in y



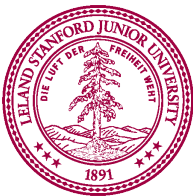
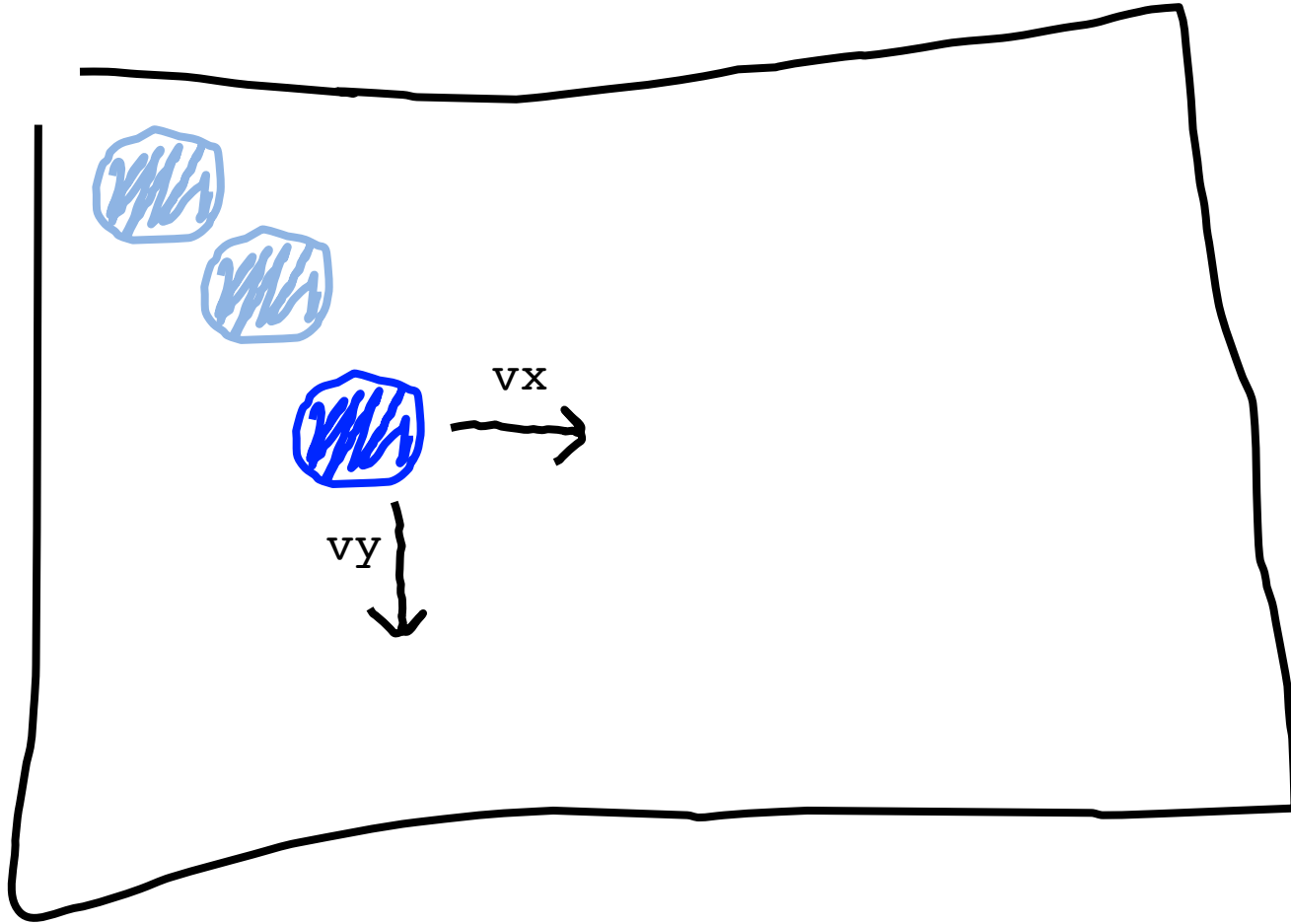
Gravity Ball

Second heartbeat



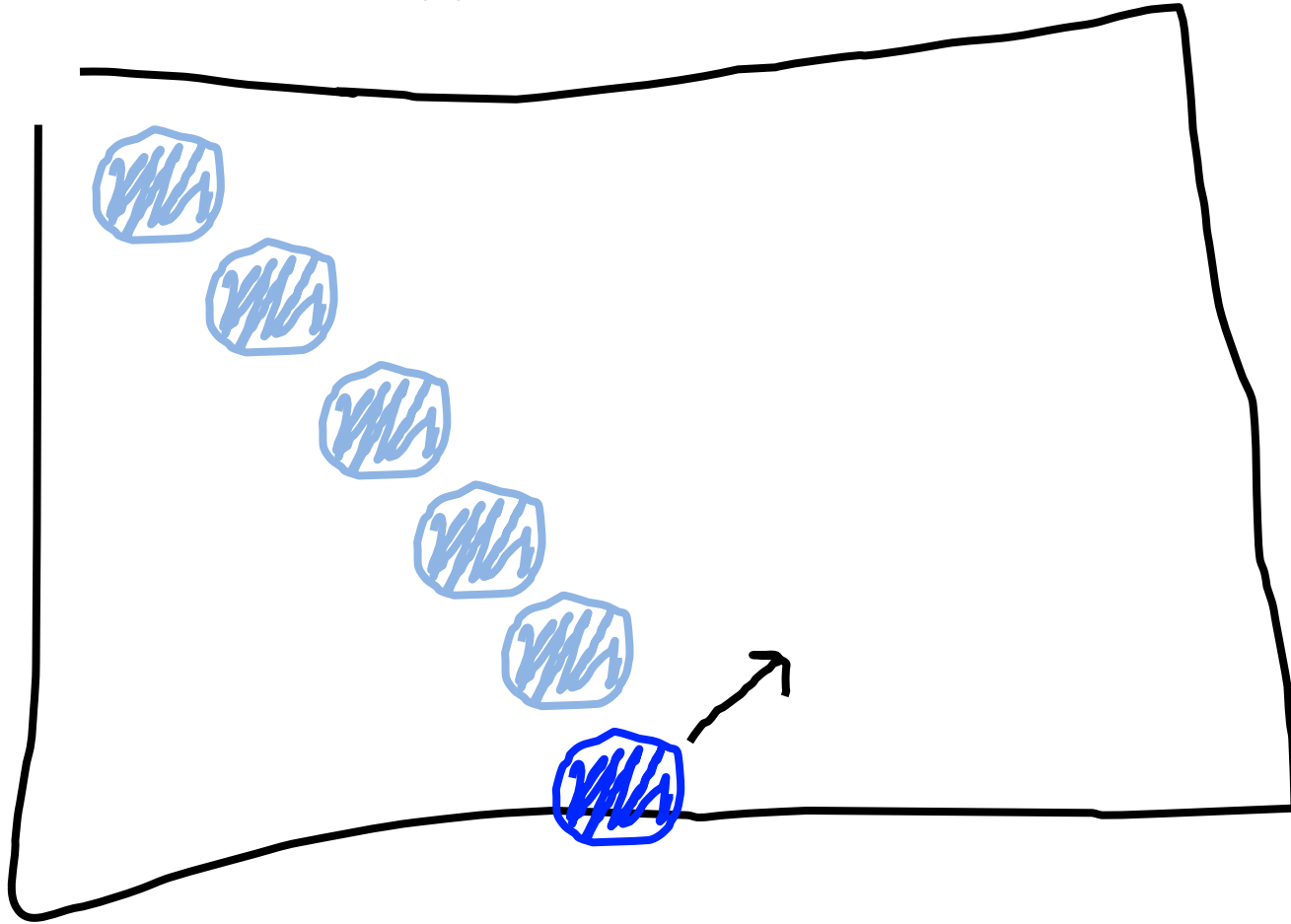
Gravity Ball

Third heartbeat



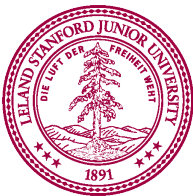
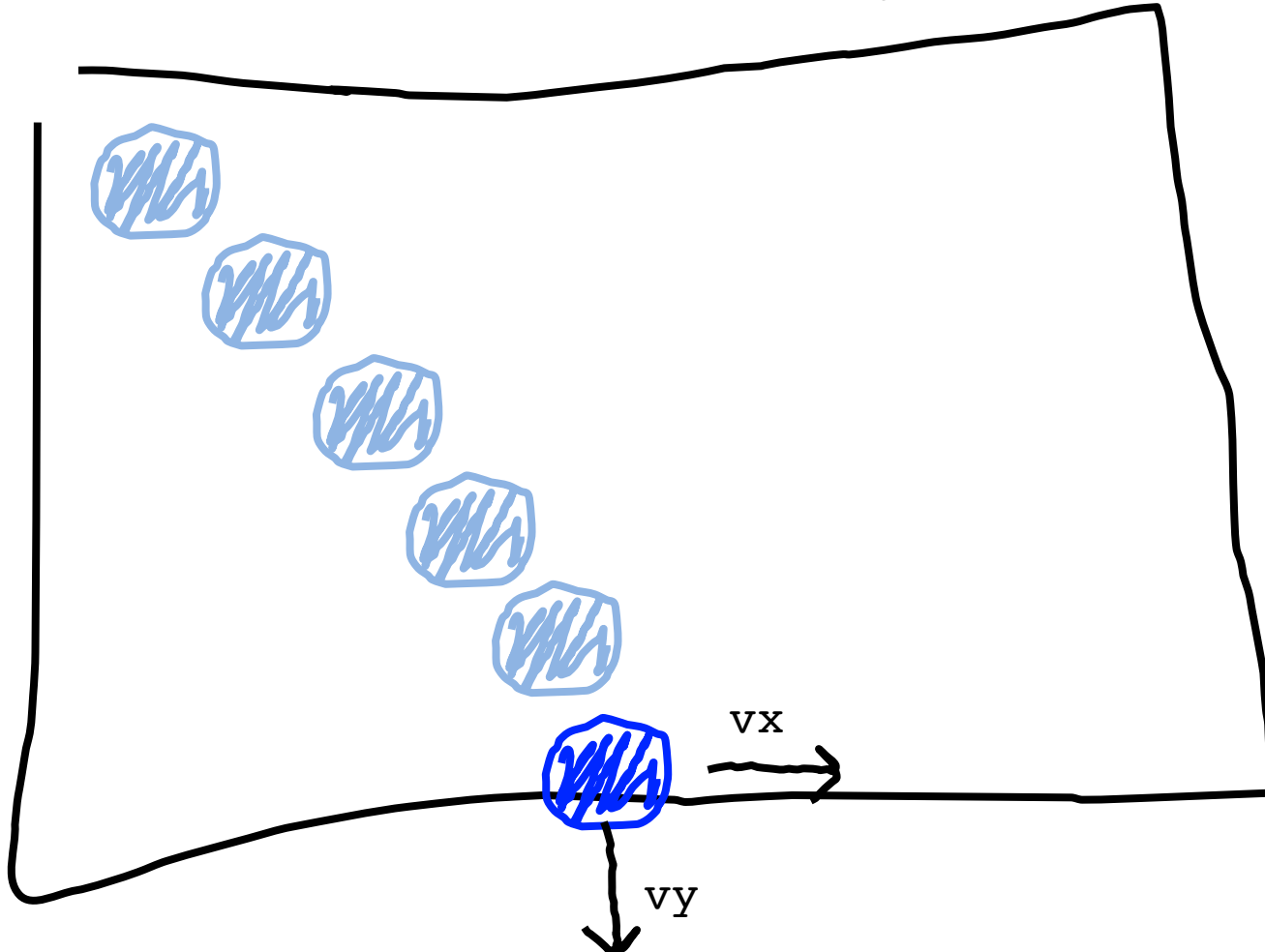
Gravity Ball

What happens when we hit a wall?



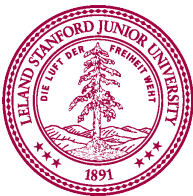
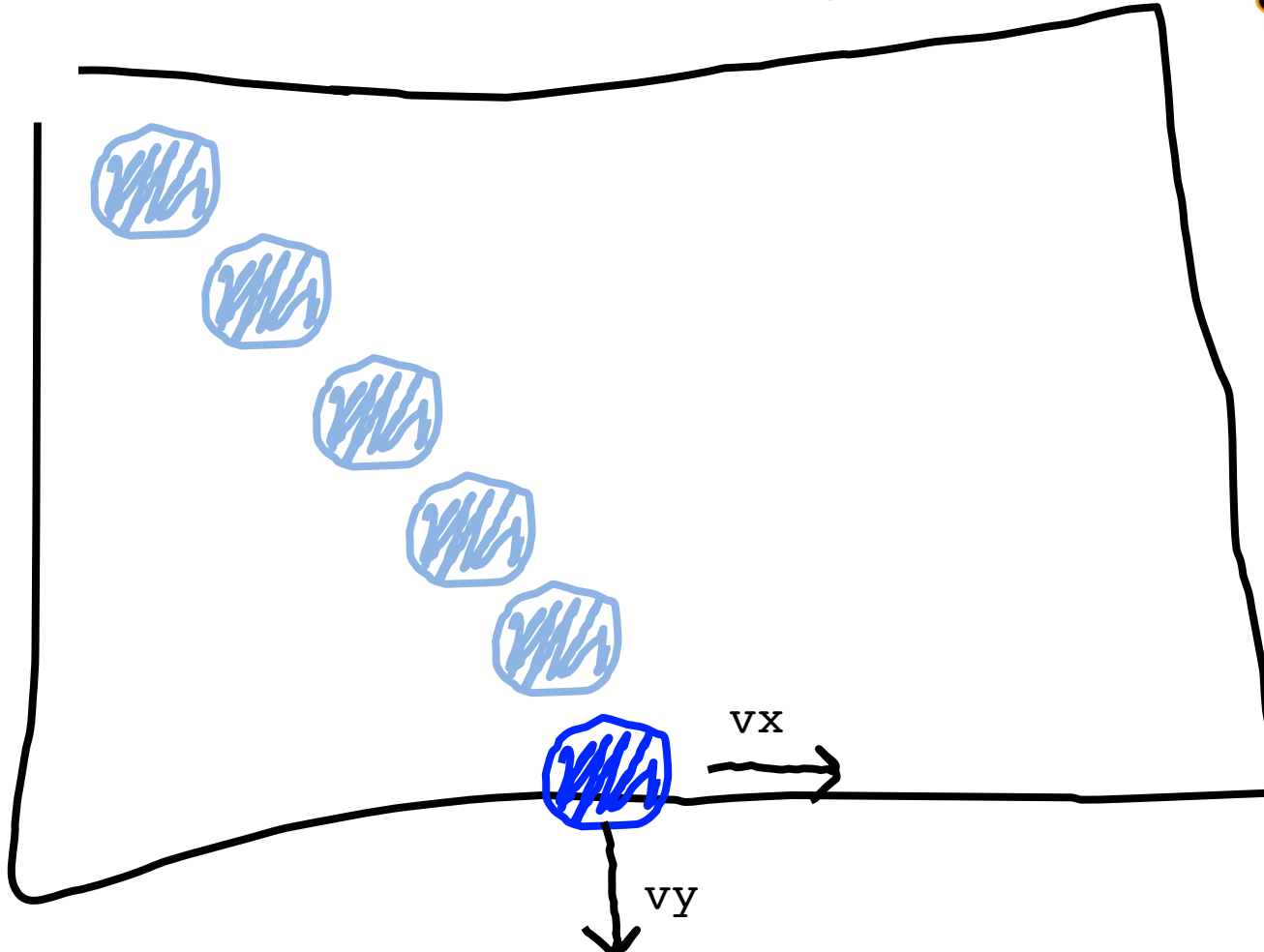
Gravity Ball

We have this velocity



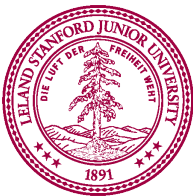
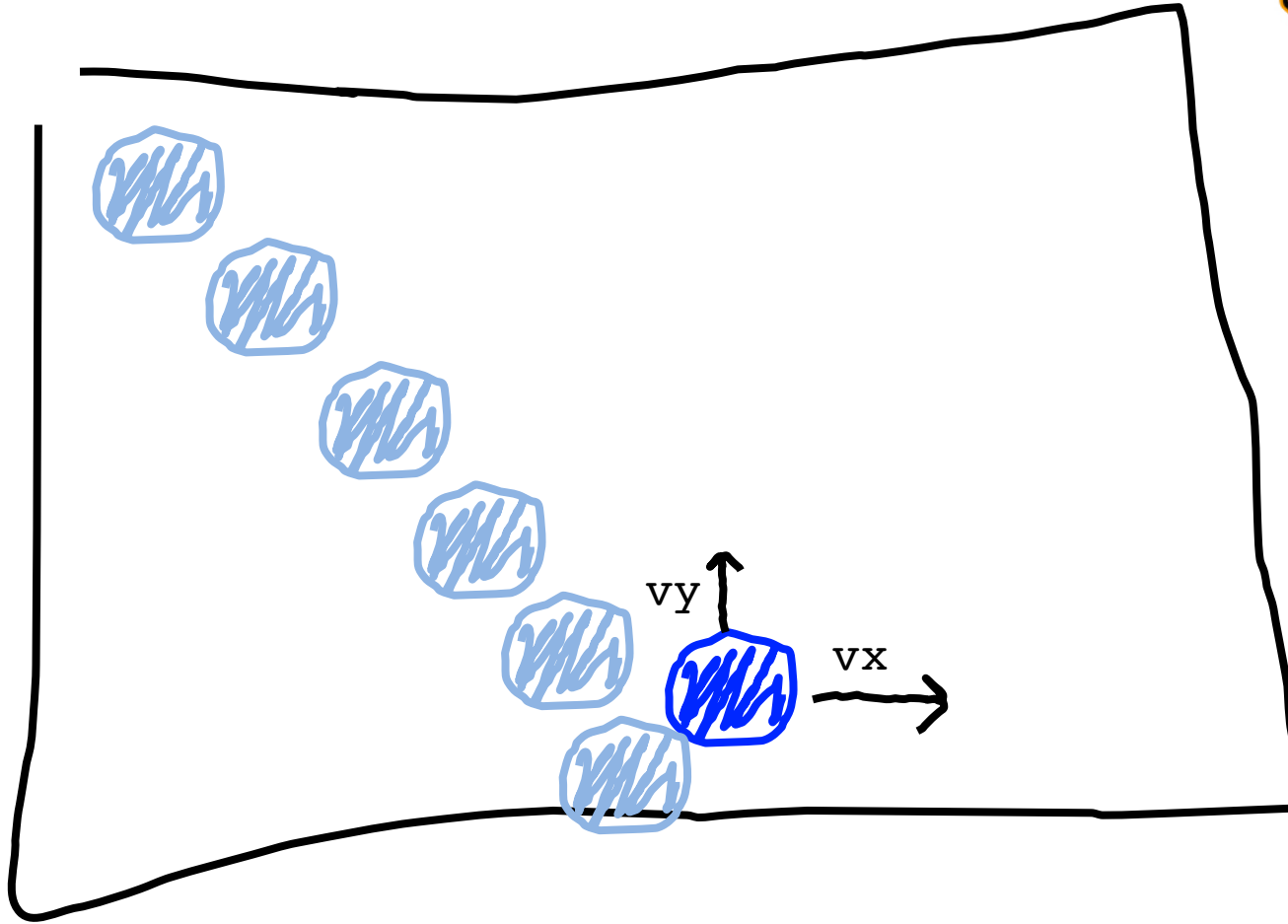
Gravity Ball

Our new velocity

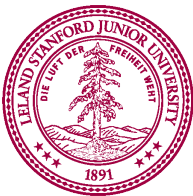


Gravity Ball

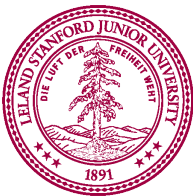
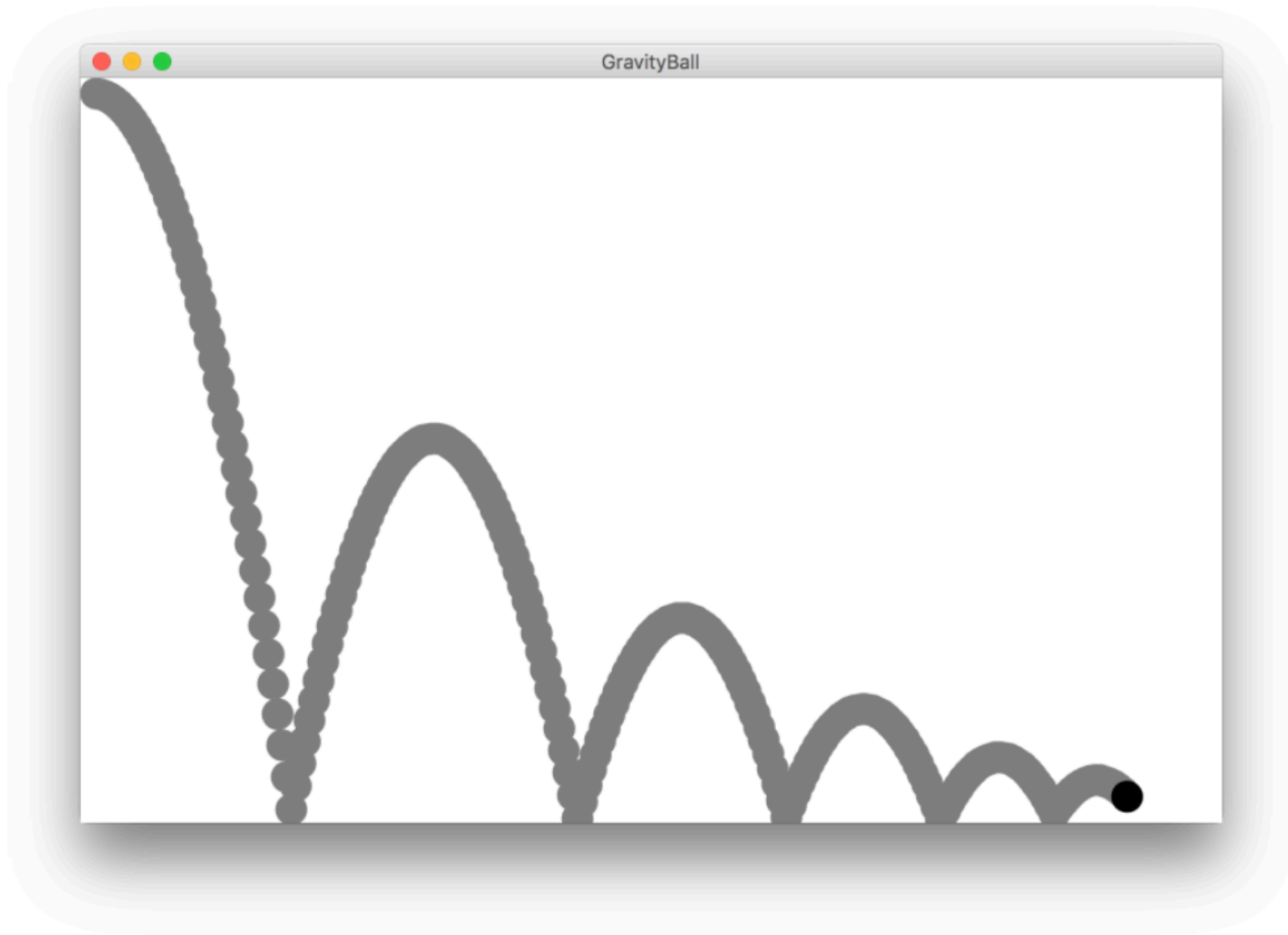
Seventh Heartbeat



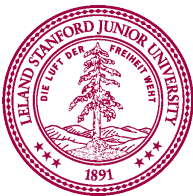
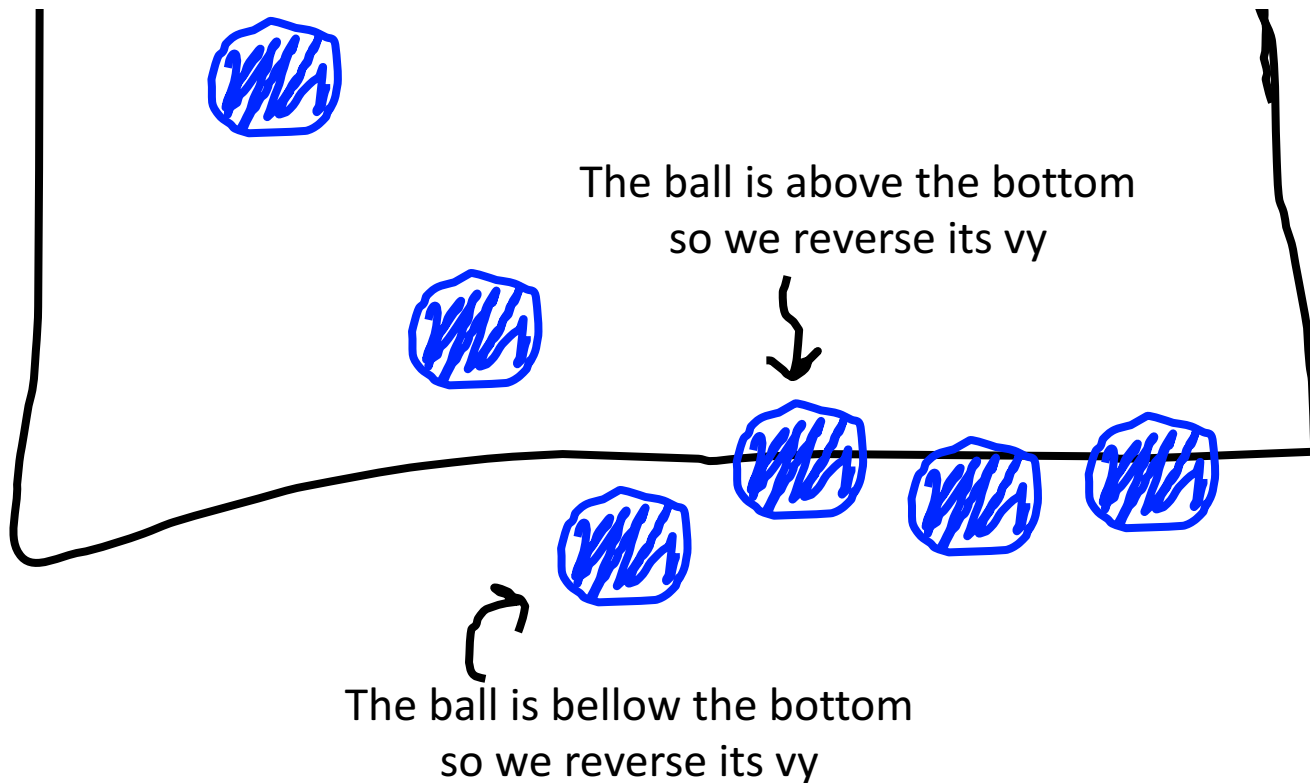
Questions?



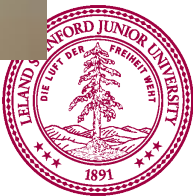
Gravity Ball



A Sticky Situation

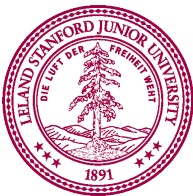


Centering



A Variable love story

By Chris

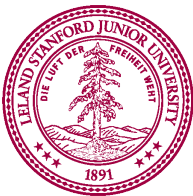


Once upon a time..

x was looking for love...

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x



x was looking for love...

```
int x = 5;
```

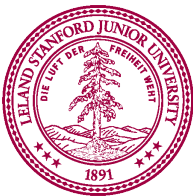
```
if(lookingForLove()) {
```

```
    int y = 5;
```

```
}
```

```
println(x + y);
```

A hand-drawn diagram consisting of a large, hand-drawn bracket shape. Inside the top part of the bracket is the number '5'. Below the bottom part of the bracket is the letter 'x'. This diagram represents the state of the variable x after the code execution, where it holds the value 5.



x was looking for love...

```
int x = 5;
```

x was definitely
looking for love

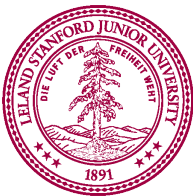
```
if(lookingForLove()) {
```

```
    int y = 5;
```

```
}
```

```
println(x + y);
```

A hand-drawn diagram consisting of a large, hand-drawn bracket shape. Inside the top part of the bracket is the number '5'. Below the bottom part of the bracket is the letter 'x'. This diagram represents the state of the variable x, which holds the value 5.

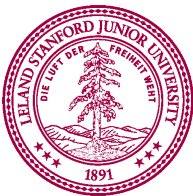


And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y



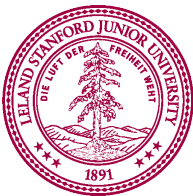
And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y

Hi, I'm y



“Wow!”

And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

Wow

5
x

5
y



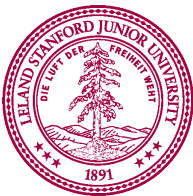
And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y

We have so much
in common



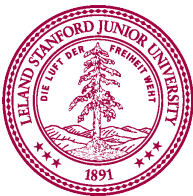
And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y

We both have
value 5!



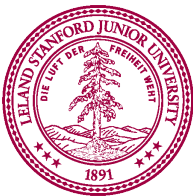
And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y

Maybe one day
we can...



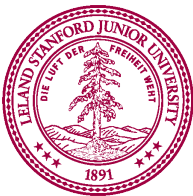
And met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y

println together?

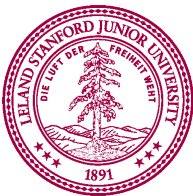


They got along

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y



It was a beautiful match...

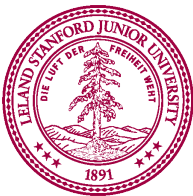
But then tragedy struck.

Tragedy Struck

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

5
x

5
y



Tragedy Struck

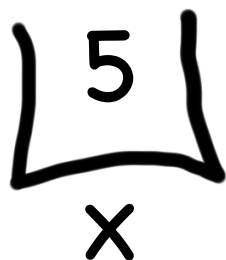
```
int x = 5;
```

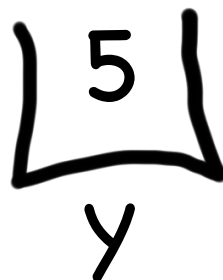
```
if(lookingForLove()) {
```

```
    int y = 5;
```

```
}  
}
```

```
println(x + y);
```

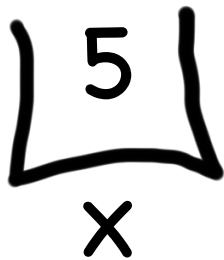
A hand-drawn diagram showing the number 5 inside a bracket-like shape, with the letter x written below it.

A hand-drawn diagram showing the number 5 inside a bracket-like shape, with the letter y written below it.



Tragedy Struck

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```

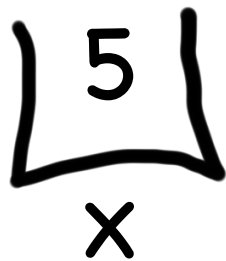


Nooooooooooooooooooooo!

You see...

When a program exits the code block...

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



Where a variable was declared...

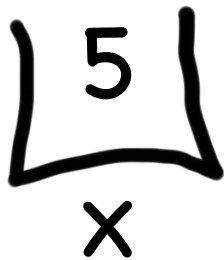
```
int x = 5;
```

```
if(lookingForLove()) {
```

```
    int y = 5;
```

```
}  
}
```

```
println(x + y);
```



It gets deleted from memory!

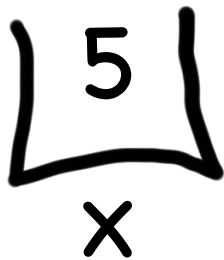
```
int x = 5;
```

```
if(lookingForLove()) {
```

```
    int y = 5;
```

```
}  
}
```

```
println(x + y);
```

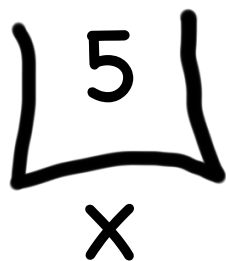


Since y was declared inside the if

```
int x = 5;
```

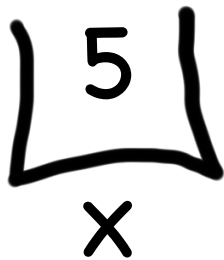
```
if(lookingForLove()) {  
    int y = 5;  
}
```

```
println(x + y);
```



It gets deleted from memory here

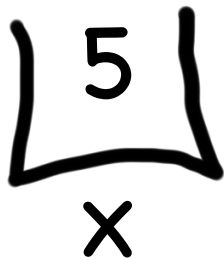
```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}  
println(x + y);
```



And doesn't exist here.

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
}
```

```
println(x + y);
```

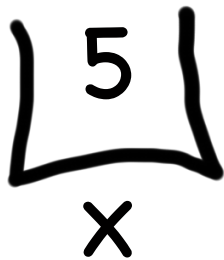


And doesn't exist here.

```
int  
if(l  
}  
}
```

**Error. Undefined
variable y.**

```
println(x + y);
```

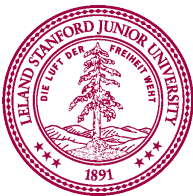


The End

Or is it?

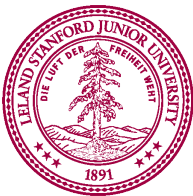
Variables have a lifetime

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```



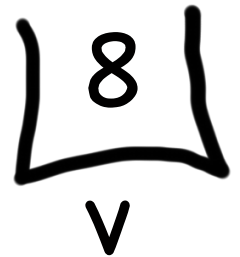
Variables have a lifetime

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```

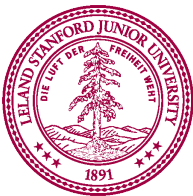


Come to existence when declared

```
public void run() {  
    double v = 8; ← Comes to life here  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```



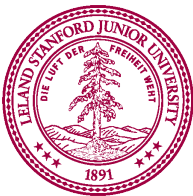
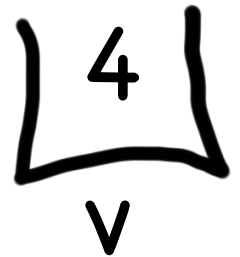
A hand-drawn diagram consisting of a large right-facing curly bracket. Inside the top part of the bracket is the number 8. Below the bottom part of the bracket is the letter v.



Live Until End of Code-Block

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```

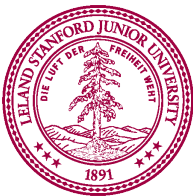
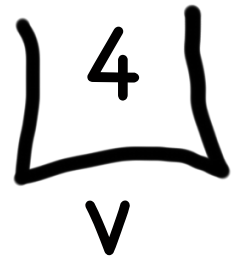
← This is the inner most code block in which it was declared....



Variables have a lifetime

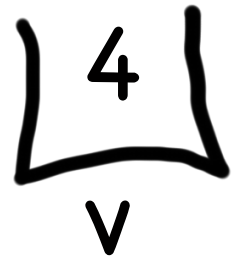
```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```

Still alive here...

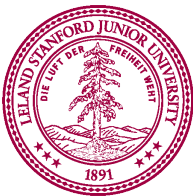


Live Until End of Code-Block

```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```



← It dies here (at the end of its code block)



Live Until End of Code-Block

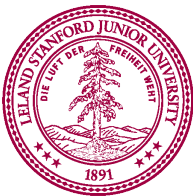
```
public void run() {  
    double v = 8;  
    if (condition) {  
        v = 4;  
        ... some code  
    }  
    ... some other code  
}
```

← It dies here (at the end of its code block)



A Variable love story

Chapter 2

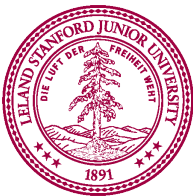


The programmer fixed her bug

x was looking for love...

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
    println(x + y);  
}
```

5
x



x was looking for love...

```
int x = 5;
```

x was definitely
looking for love

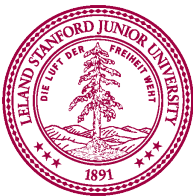
```
if(lookingForLove()) {
```

```
    int y = 5;
```

```
    println(x + y);
```

```
}
```

A hand-drawn diagram consisting of a large, hand-drawn bracket shape. Inside the top part of the bracket is the number '5'. Below the bottom part of the bracket is the letter 'x'. This represents the state of the variable x after the first line of code is executed.

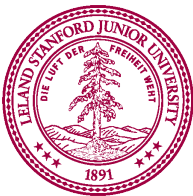


x met y

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
    println(x + y);  
}
```

5
x

5
y

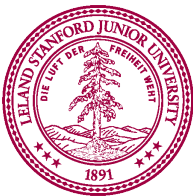


Since they were both in scope...

```
int x = 5;  
if(lookingForLove()) {  
    int y = 5;  
    println(x + y);  
}
```

5
x

5
y



The story had a happy ending!