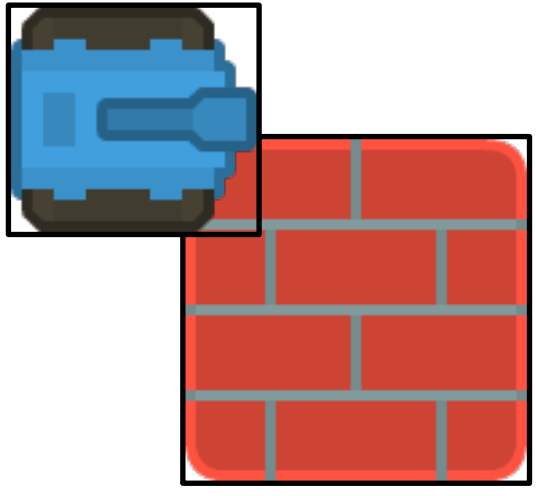
Introduction to Processing

Collision Detection

Rectangle-Rectangle Collision

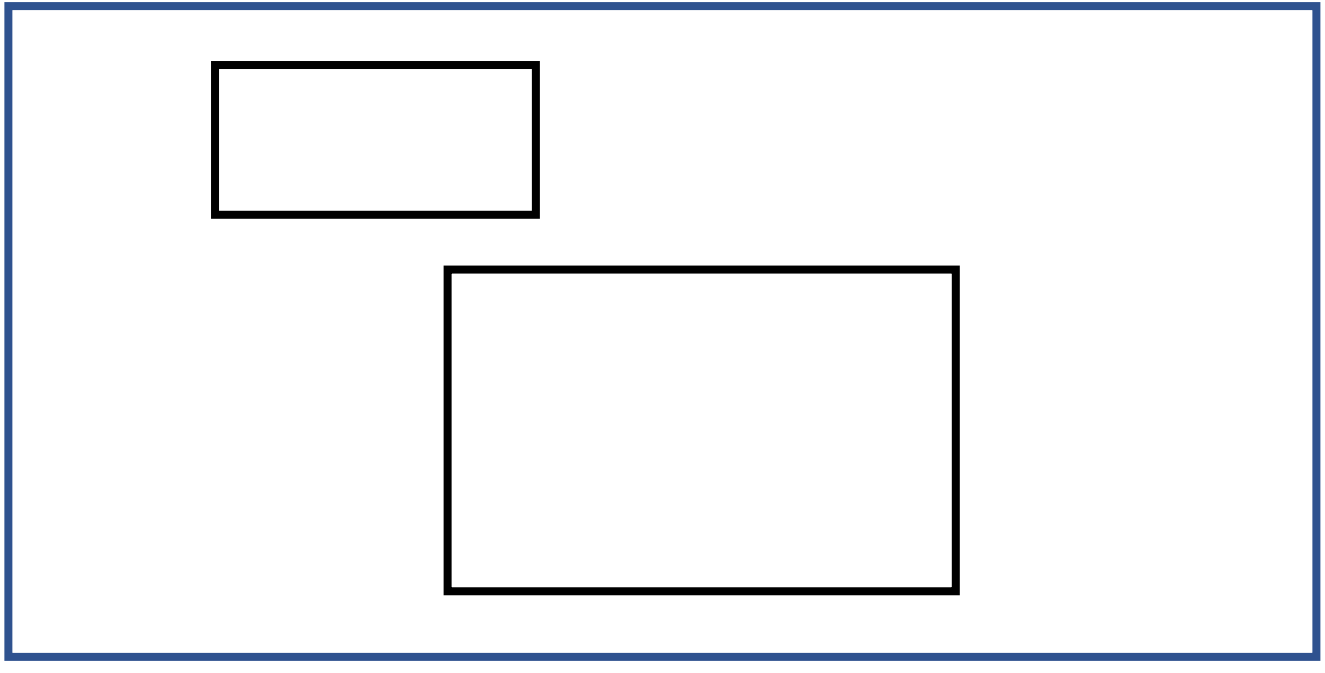
Since images are simply rectangular array of pixels, rectangle-rectangle collision is very useful for writing games.



Rectangle-Rectangle Collision

Rectangles below have a horizontal overlap but not a vertical one.

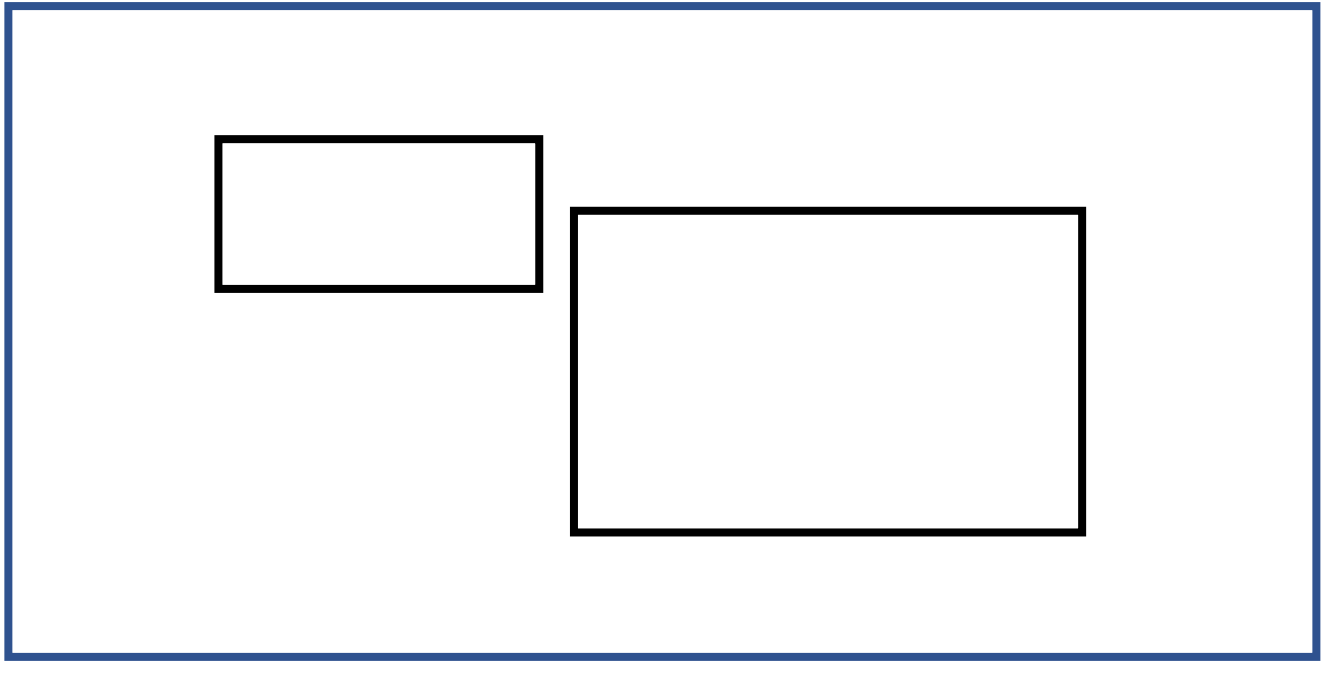
Origin (0,0)



Rectangle-Rectangle Collision

Rectangles below have a vertical overlap but not a horizontal one.

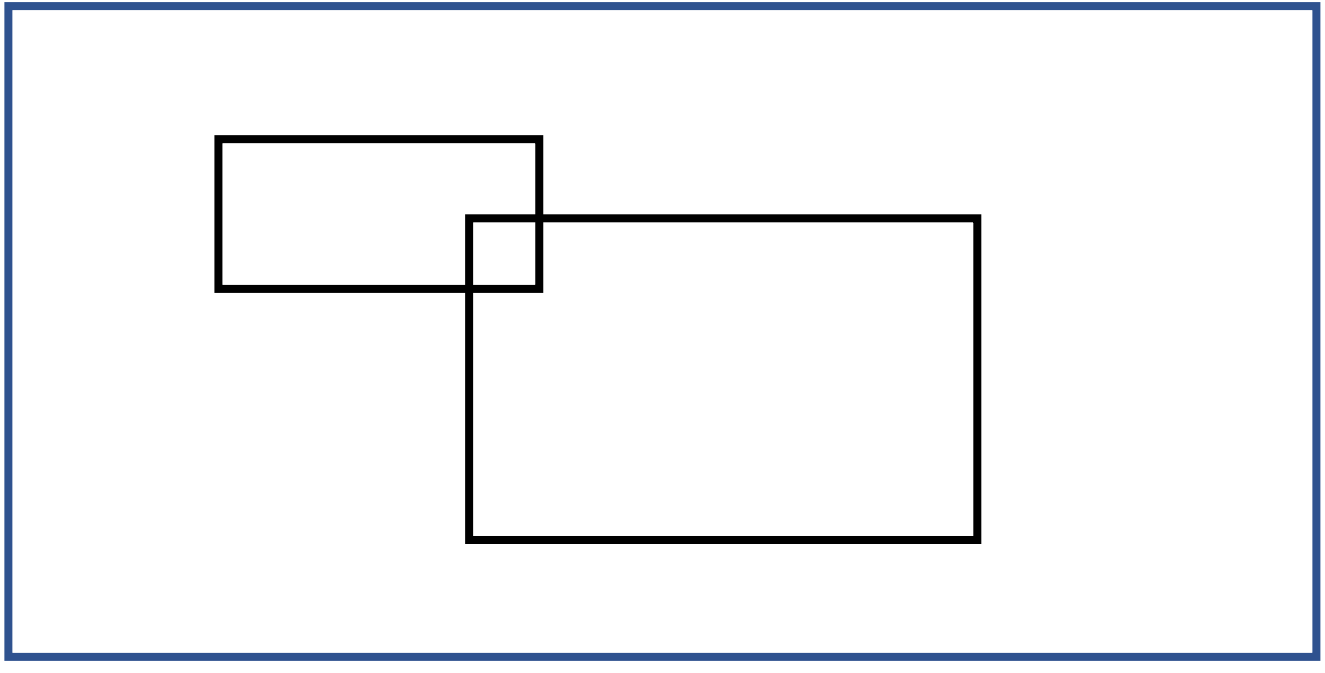
Origin (0,0)



Rectangle-Rectangle Collision

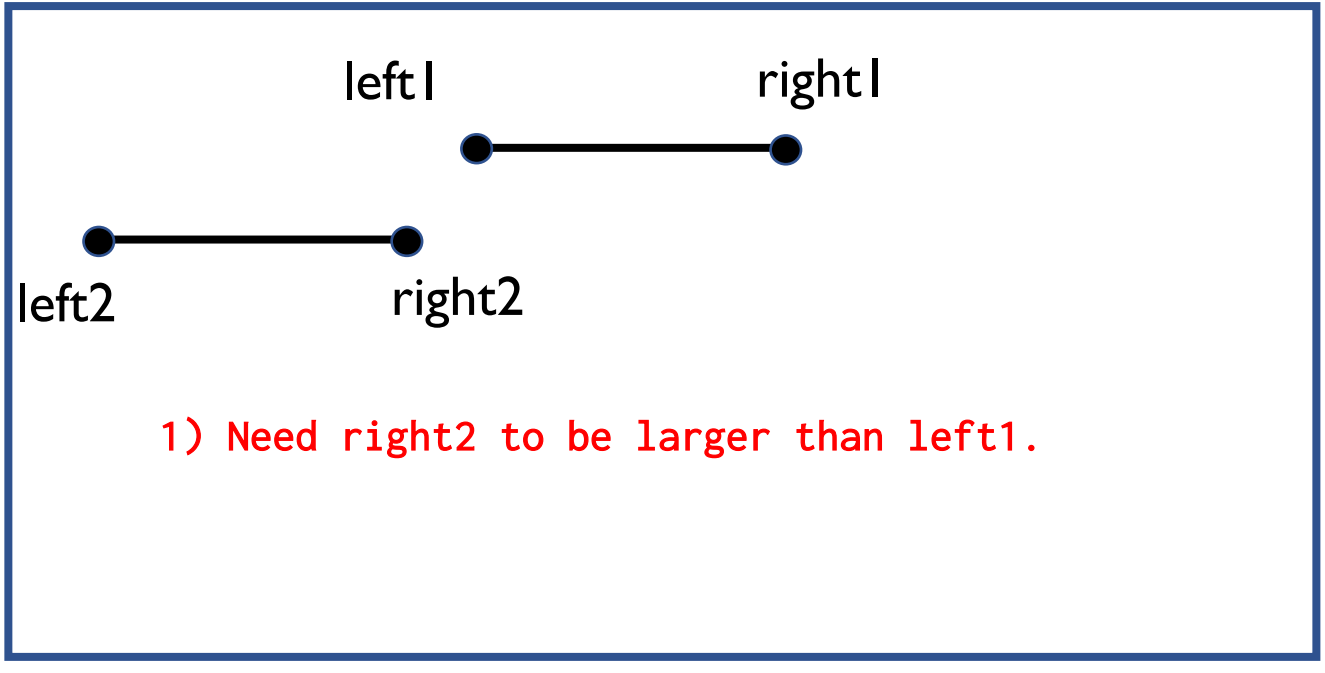
Rectangles below have overlaps in both directions.

Origin (0,0)



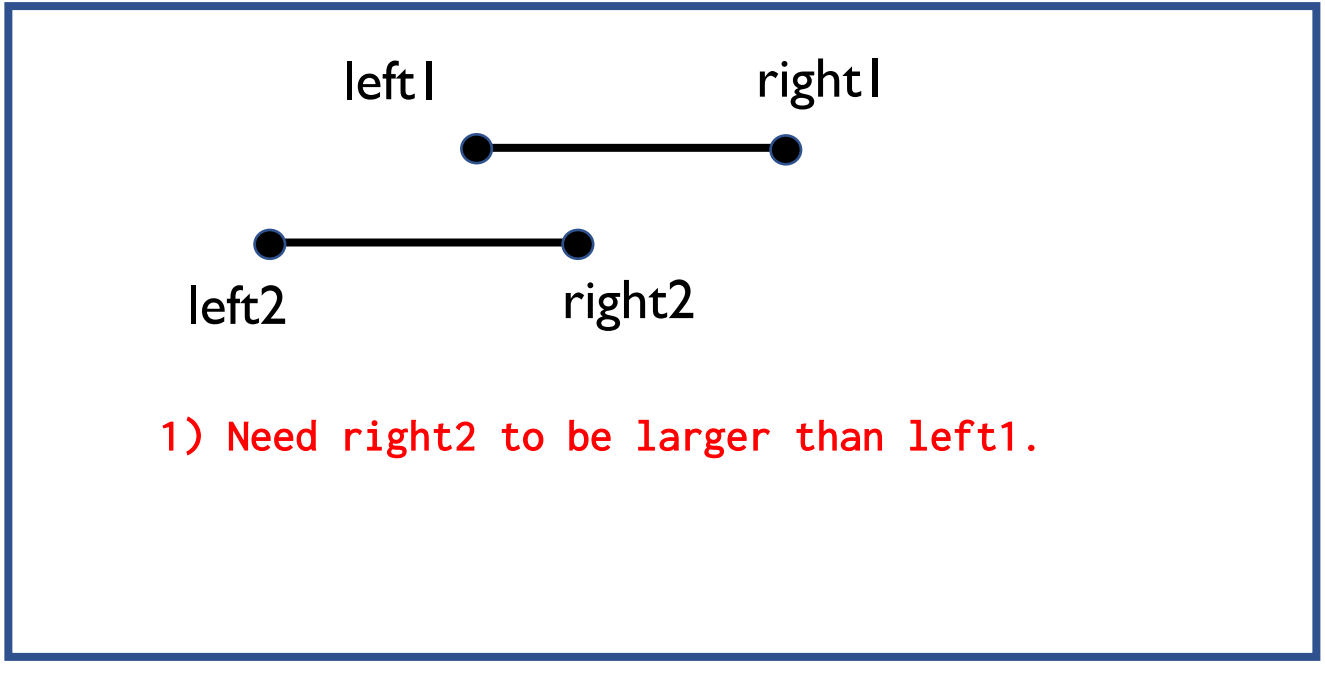
Checking Overlap

Origin (0,0)



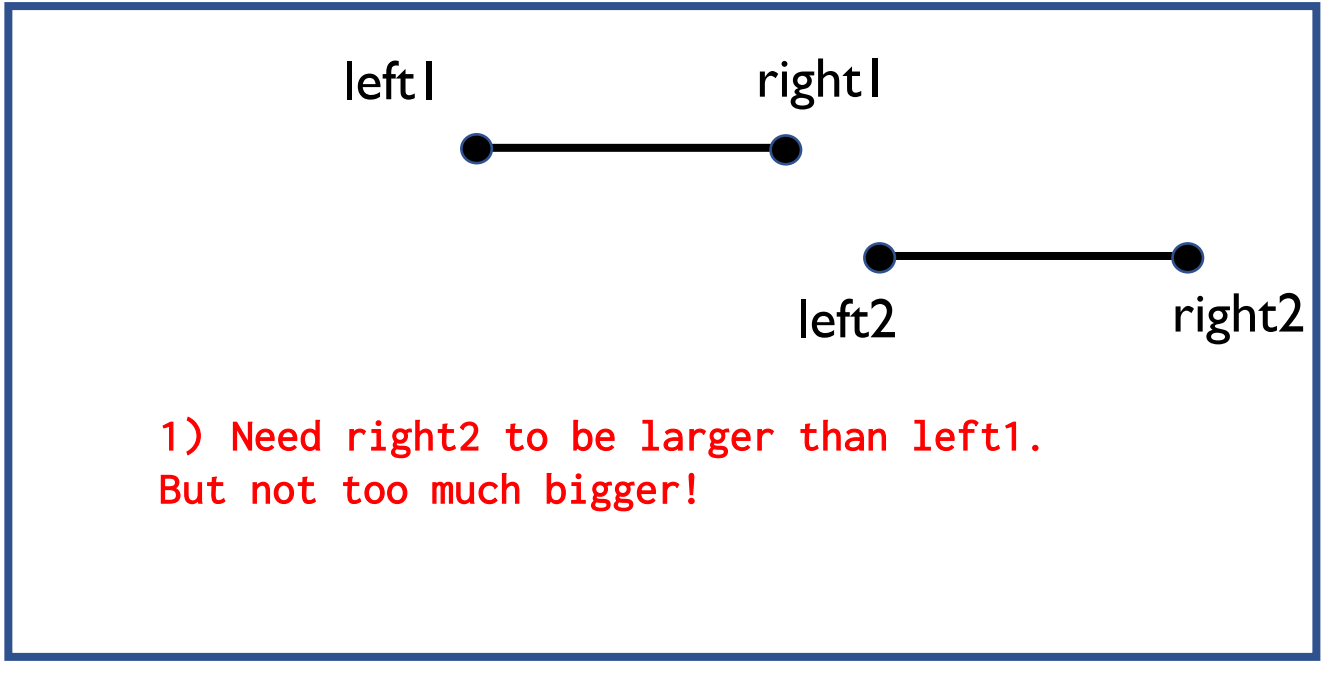
Checking Overlap

How do we check for overlap?



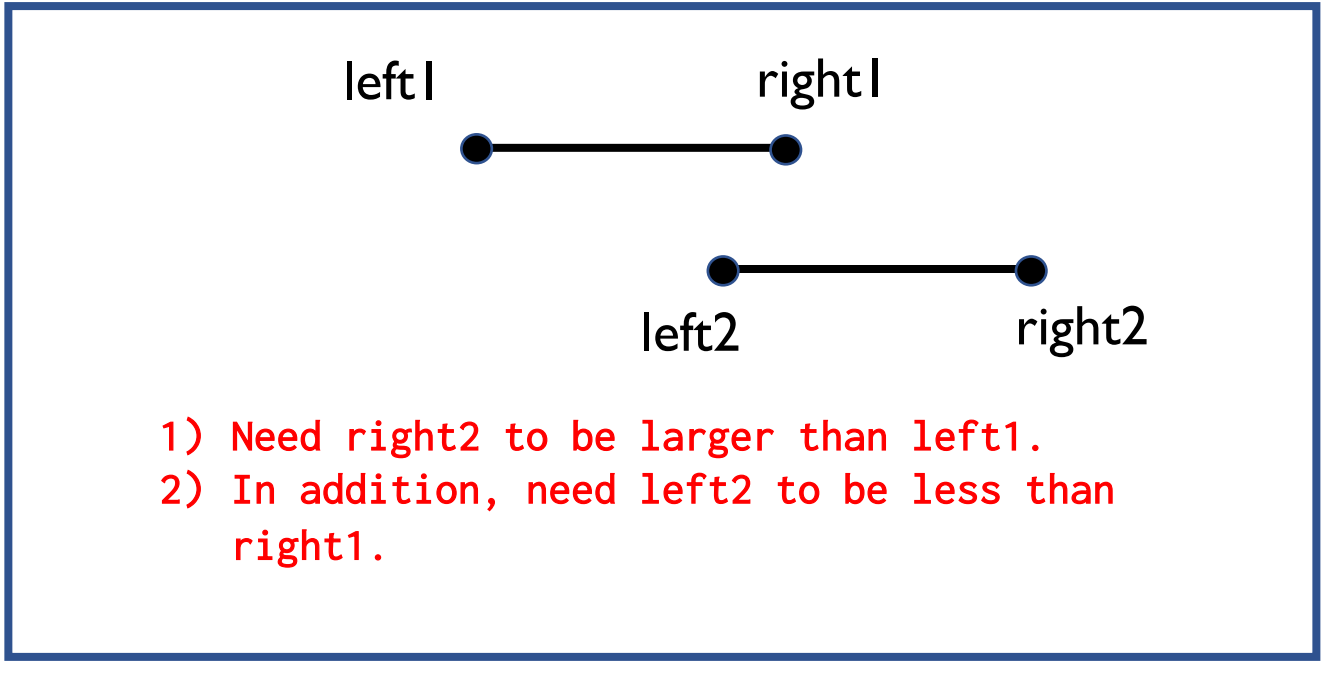
Checking Overlap

How do we check for overlap?



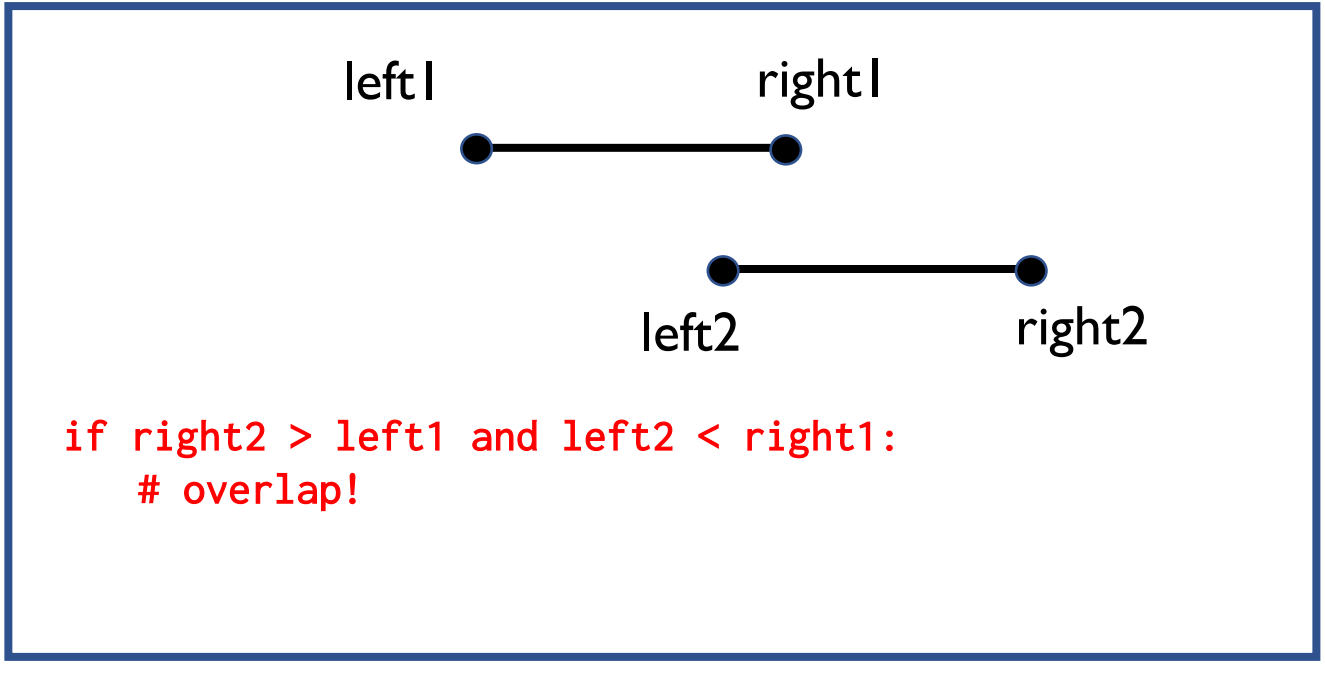
Checking Overlap

How do we check for overlap?



Checking Overlap

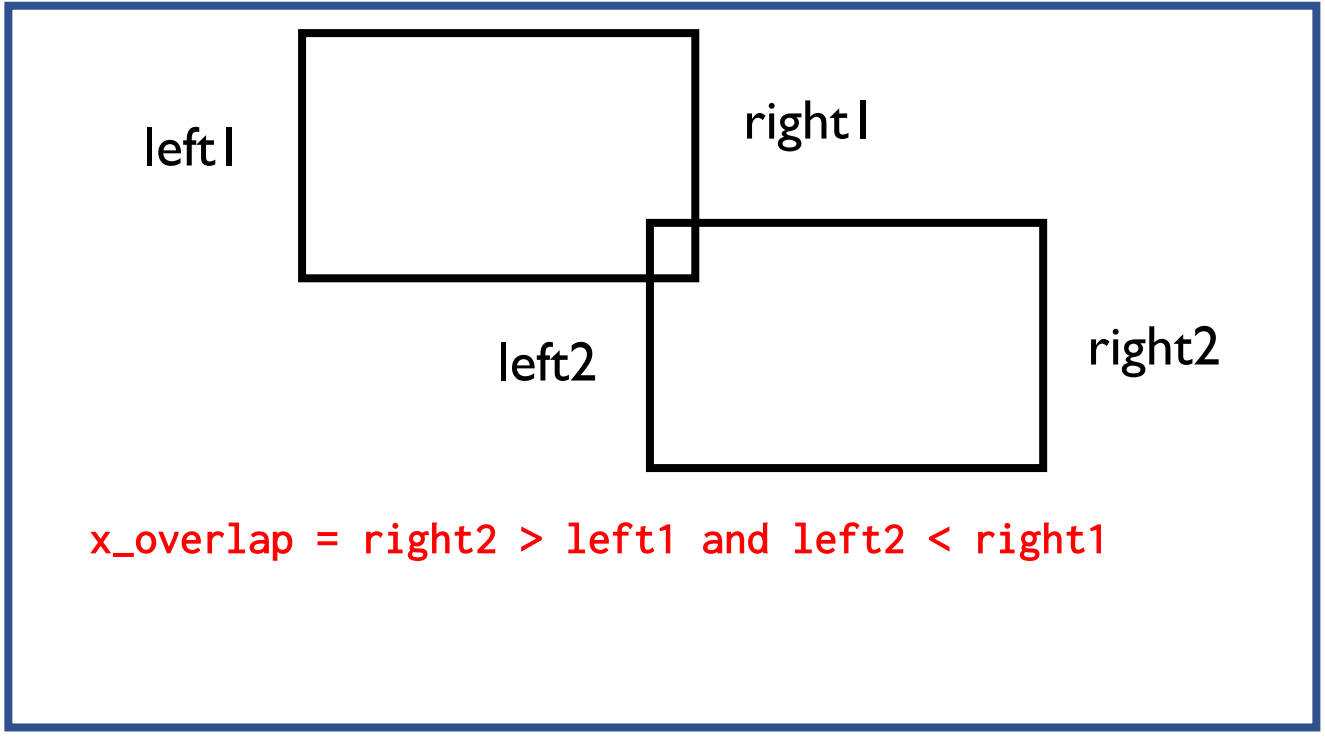
How do we check for overlap?




```
if right2 > left1 and left2 < right1:
    # overlap!
```

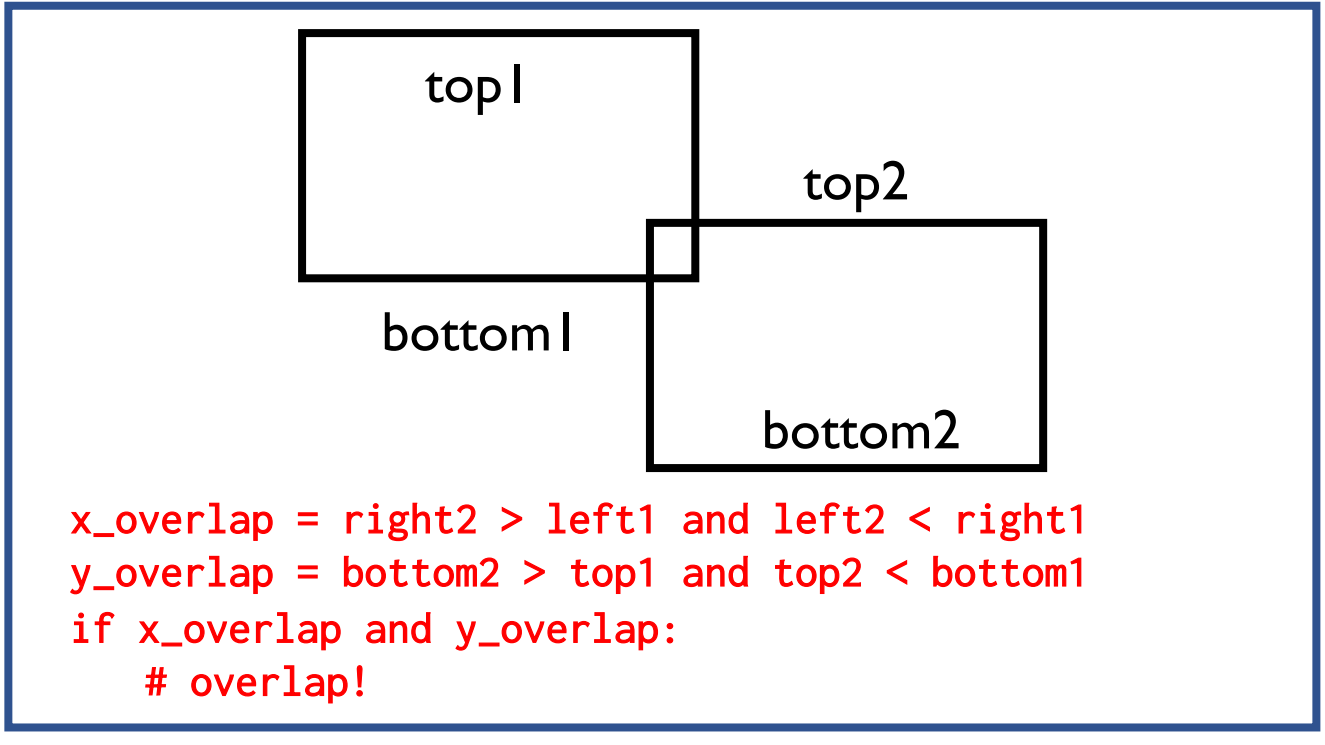
Rectangle-Rectangle Collision

Rectangles below have overlaps in both directions.



Rectangle-Rectangle Collision

Rectangles below have overlaps in both directions.



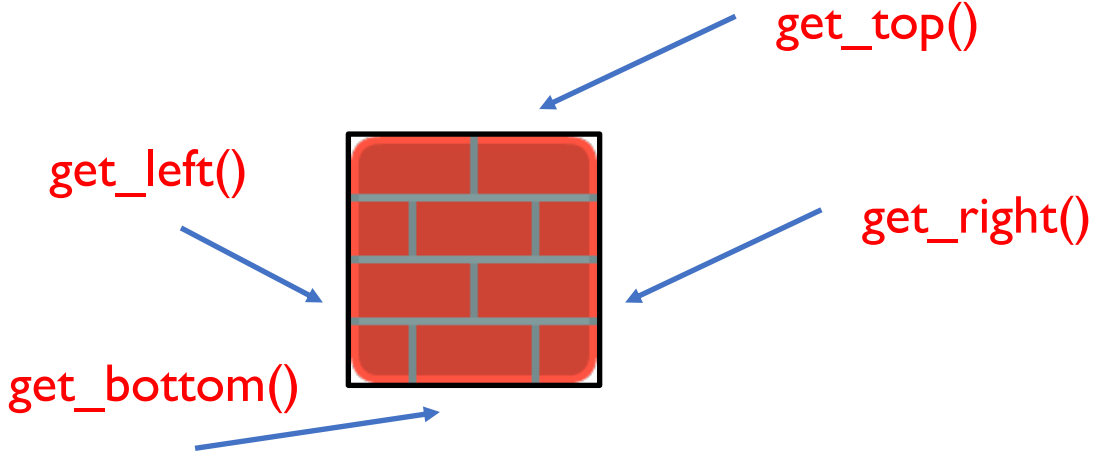

```
x_overlap = right2 > left1 and left2 < right1
y_overlap = bottom2 > top1 and top2 < bottom1
if x_overlap and y_overlap:
    # overlap!
```

check_for_collision(sprite1, sprite2)

We'll write the check_for_collision method which accepts two parameters: sprite1 and sprite2 and returns whether they intersect.

```
def check_for_collision(self, sprite1, sprite2):
    # returns whether sprite1 and sprite2 intersects
```

Use the get_left, get_right, get_top and get_bottom methods to get the respective boundaries of the sprite!



check_for_collision_list(sprite, sprite_list)

Another useful method is the check_for_collision_list which accepts two parameters: `sprite` and `sprite_list` and returns a list of sprites in `sprite_list` which intersects with `sprite`.

```
def check_for_collision_list(self, sprite, sprite_list):
    #returns list of sprites in sprite_list which
    #intersects with sprite.
    # remember to call check_for_collision! use self and
    # the dot notation.
    # if self.check_for_collision(sp1, sp2):
```

Pick Up Coins Lab

In the previous lab, you are now able to control a sprite with the keyboard.

In this lab, implement check_for_collision and check_for_collision_list. Then implement on_update so that as the tank moves about, it picks up coins and coins are removed from the screen appropriately.

Display the text which shows the coin count. For example, "Coins: 10" and update appropriately.