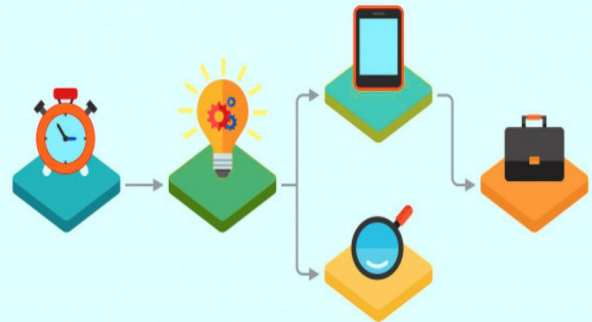


## Designing Algorithms



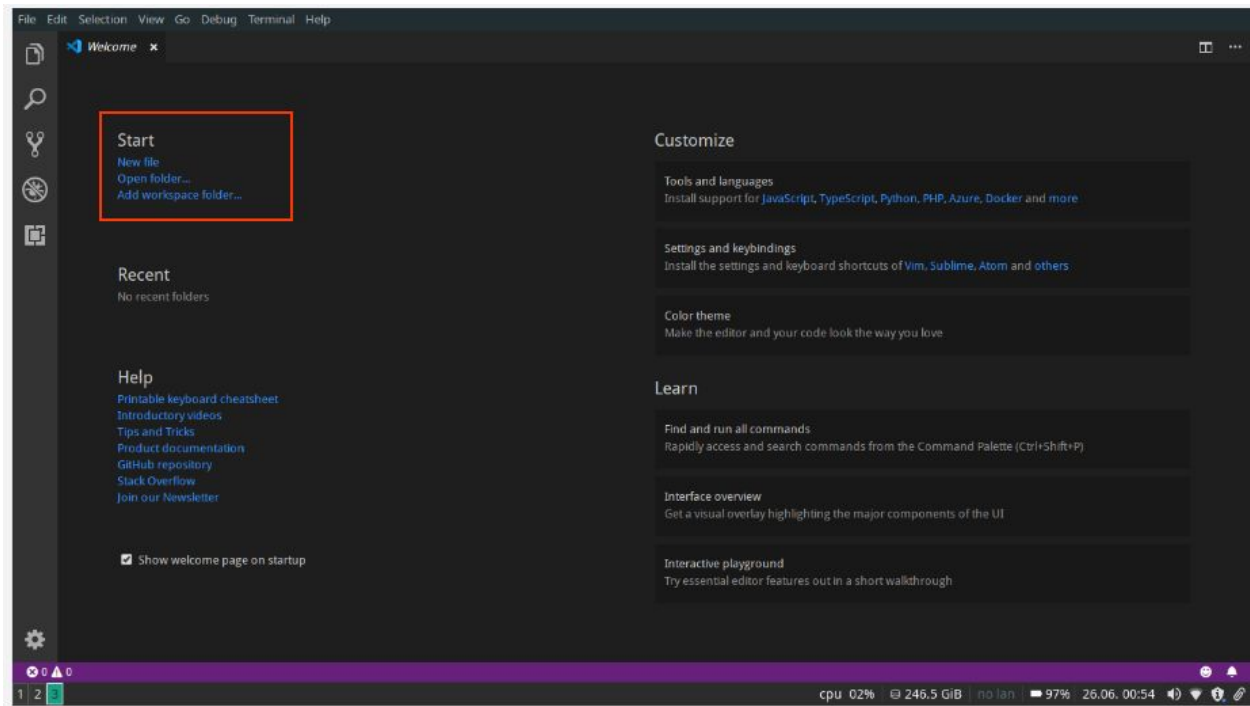
### What we did:

- Install Visual Code Studio
- Download Boilerplate code
- Design algorithm for collision detection, write code and test the program

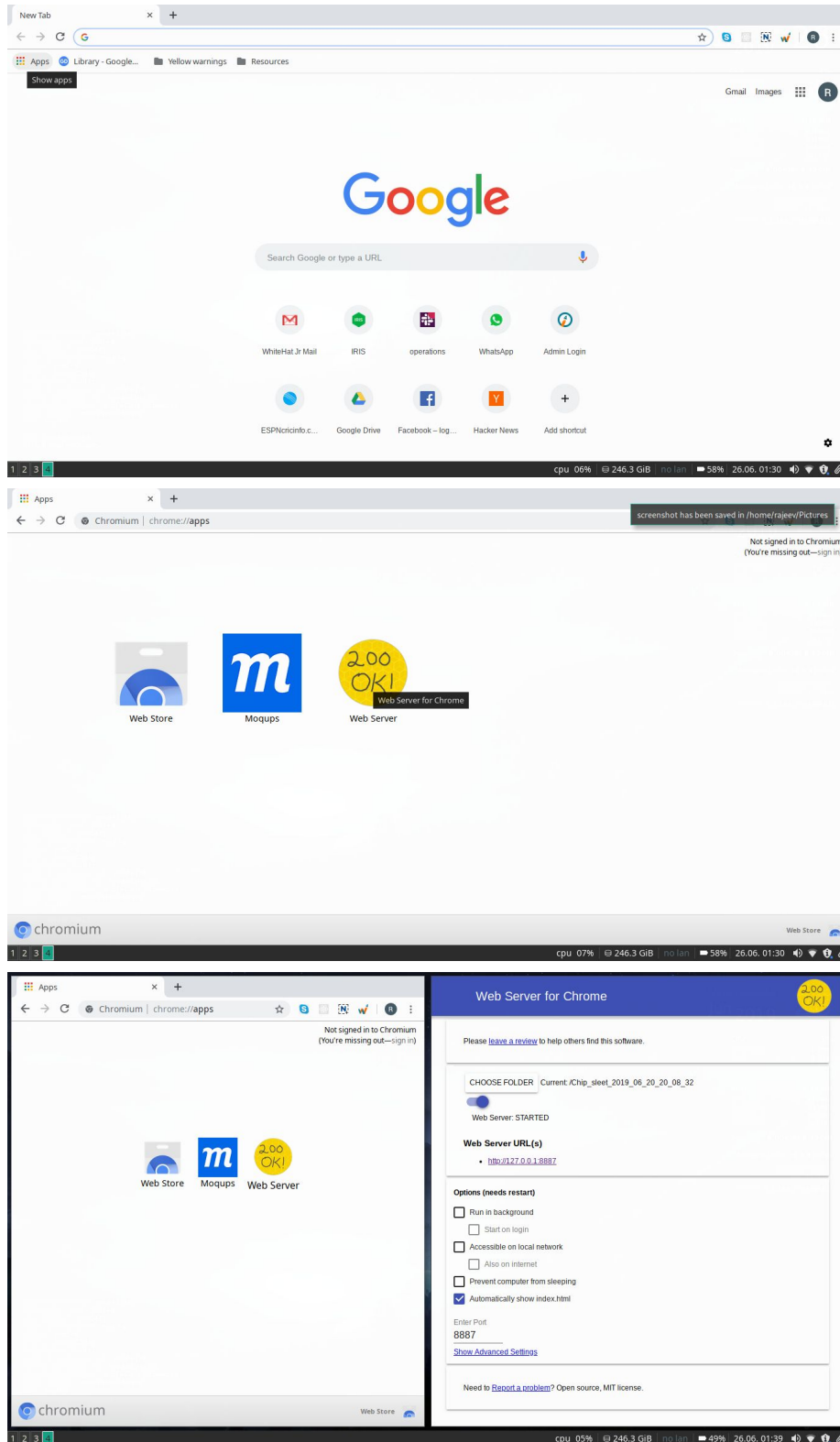
### How we did it:

**Step 1:** Install a code editor on our system: Visual Code Studio

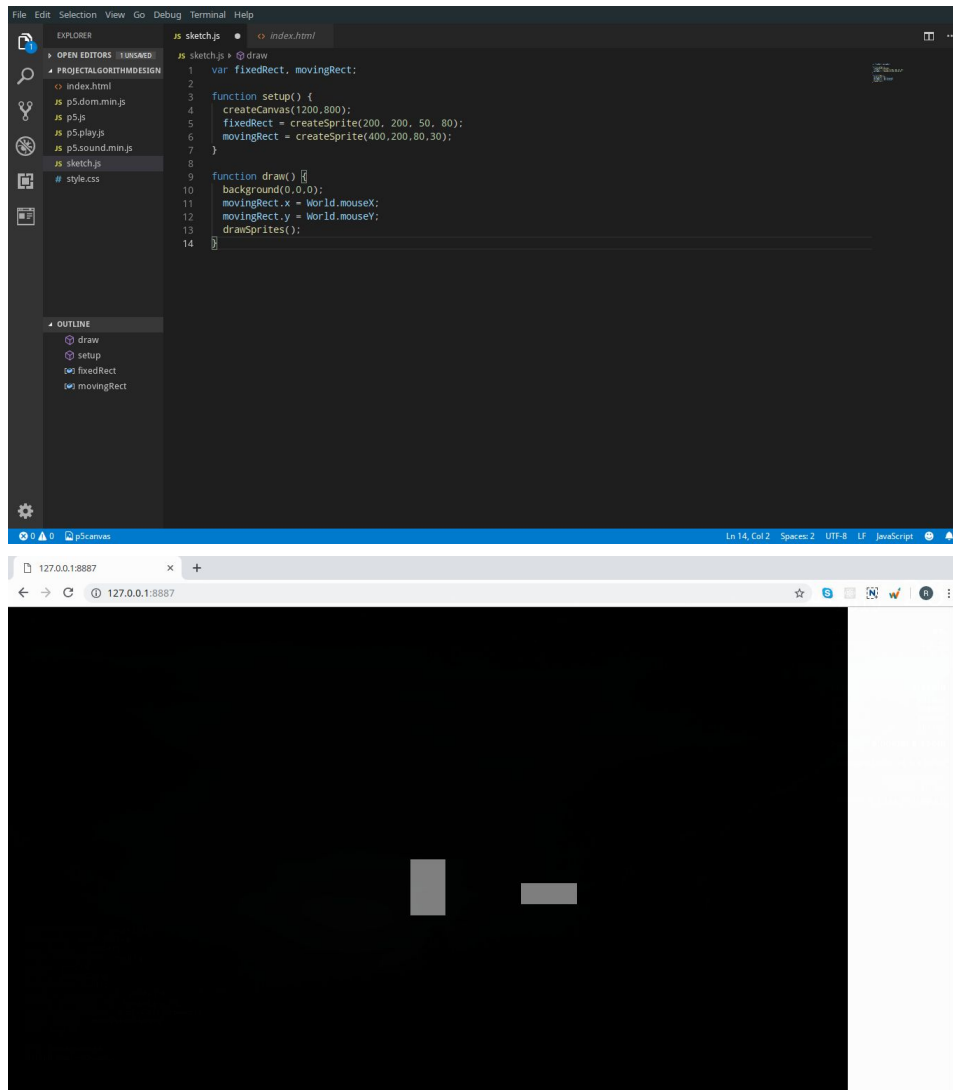
Download the boilerplate code for p5.play



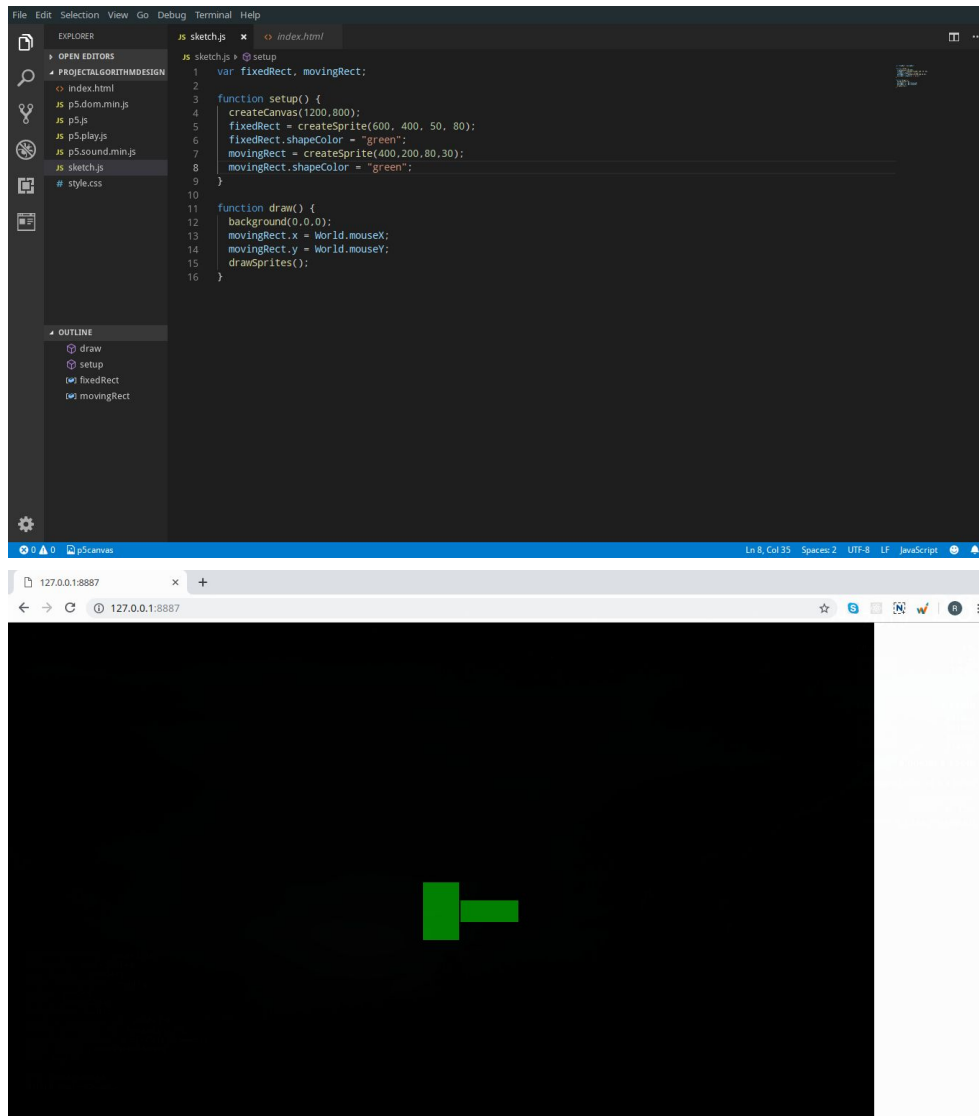
**Step 2:** Click on the webserver link which opens on the browser.



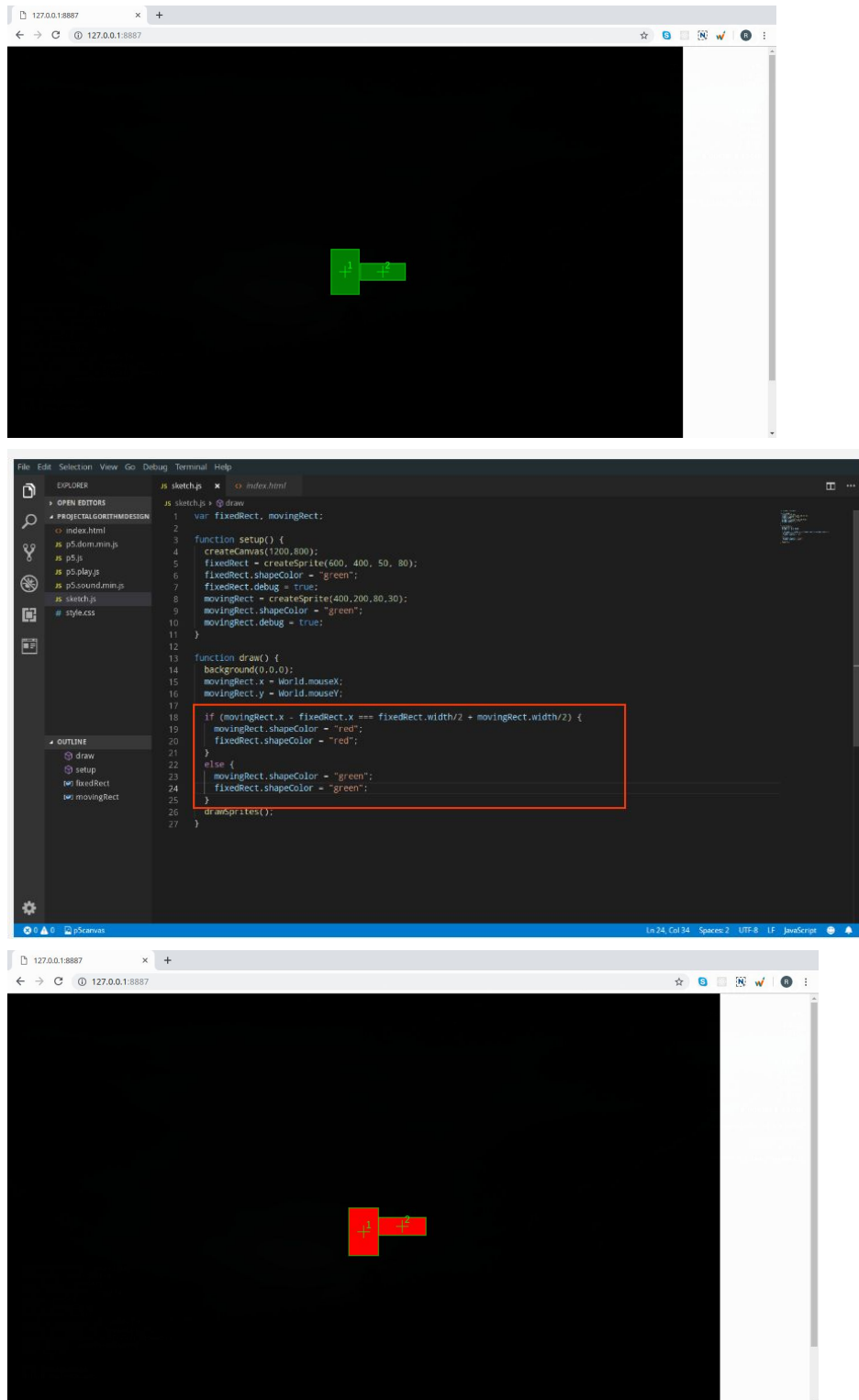
**Step 3:** Create two rectangles with different widths and heights- one rectangle should be fixed, the other should be moving. Add controls to the moving rectangle so that it moves with the mouse.



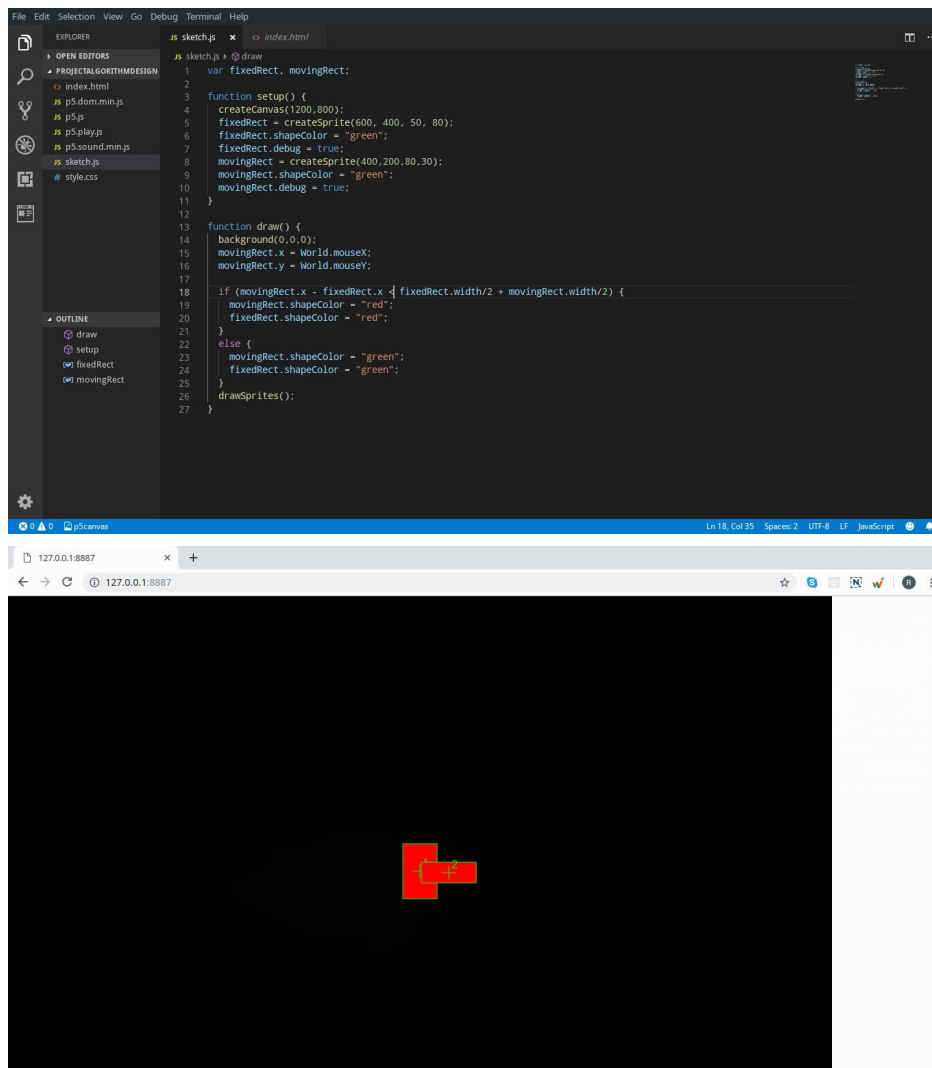
**Step 4:** Make the colour of both the rectangles green; write a collision detection algorithm so that when the two objects/rectangles collide, their colour changes to green.



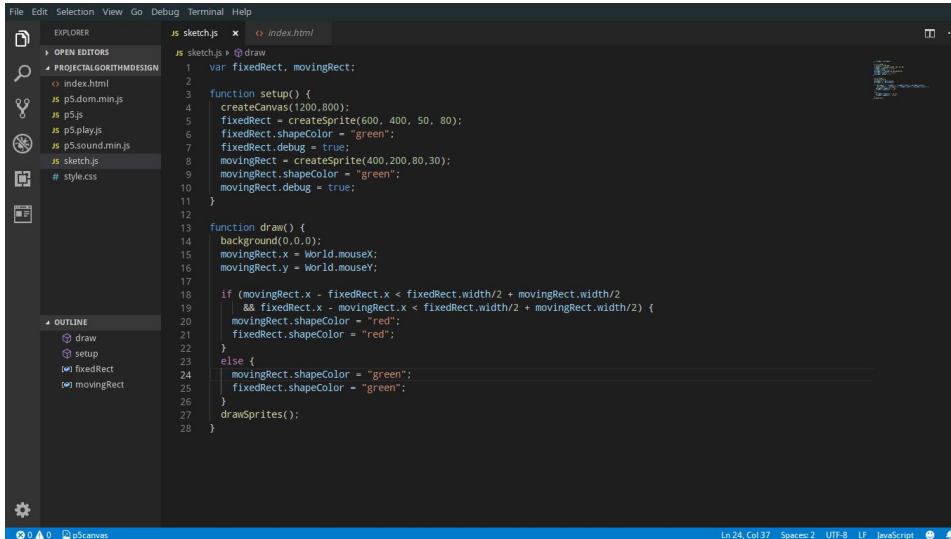
**Step 5:** Add this condition to our code and change the colour of rectangles to red when the object touches each other, else keep it to green. One of the conditions, when the objects collide, should be:  $\text{movingRect.x} - \text{fixedRect.x} = \text{fixedRect.width}/2 + \text{movingRect.width}/2$ . If the distance is greater than this, then the objects have not collided.



**Step 6:** The rectangles turn red exactly at the point of touch when moving rectangle is to the LEFT. Change "===" sign in our condition to "<", we can detect when the moving rectangle has crossed over to the fixed rectangle.



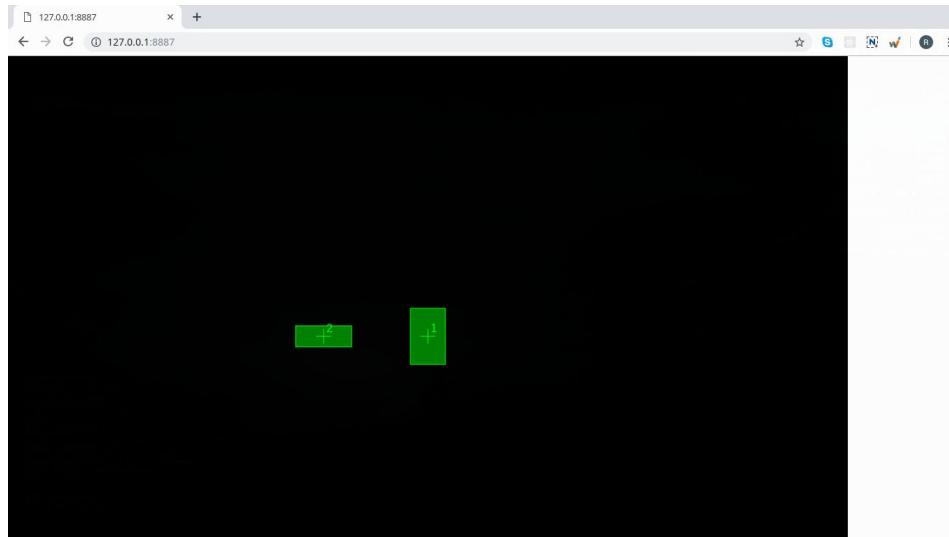
**Step 7:** Add additional conditions to complete our collision algorithm.  
 $\text{fixedRect.x} - \text{movingRect.x} < \text{fixedRect.width}/2 + \text{movingRect.width}/2$



```

1  var fixedRect, movingRect;
2
3  function setup() {
4    createCanvas(1200,800);
5    fixedRect = createSprite(600, 400, 50, 80);
6    fixedRect.shapeColor = "green";
7    fixedRect.debug = true;
8    movingRect = createSprite(400,200,80,30);
9    movingRect.shapeColor = "green";
10   movingRect.debug = true;
11 }
12
13 function draw() {
14   background(0,0,0);
15   movingRect.x = World.mouseX;
16   movingRect.y = World.mouseY;
17
18   if (movingRect.x - fixedRect.x < fixedRect.width/2 + movingRect.width/2
19       && fixedRect.x - movingRect.x < fixedRect.width/2 + movingRect.width/2) {
20     movingRect.shapeColor = "red";
21     fixedRect.shapeColor = "red";
22   }
23   else {
24     movingRect.shapeColor = "green";
25     fixedRect.shapeColor = "green";
26   }
27   drawSprites();
28 }
  
```

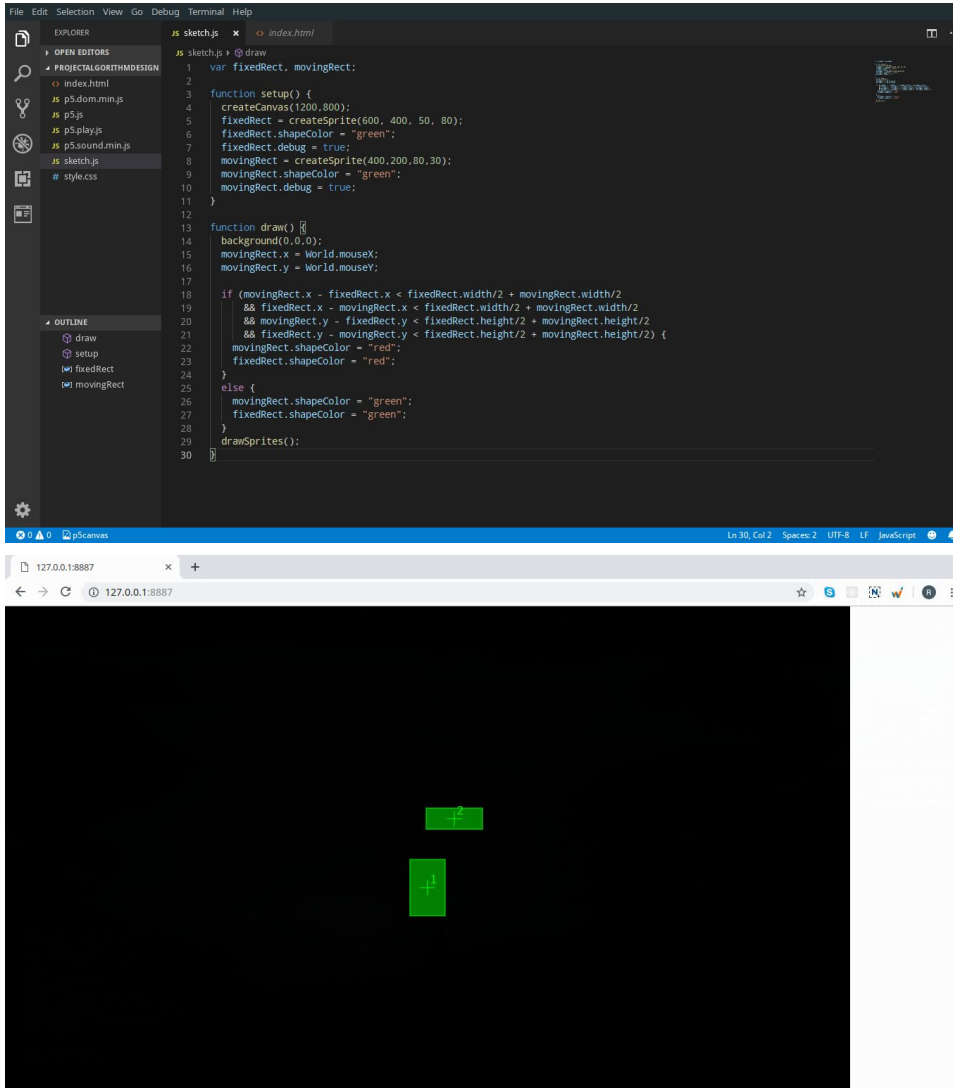
When the rectangle is moving from the left to the right, it is green and when it touches the fixed rectangle, it becomes red.





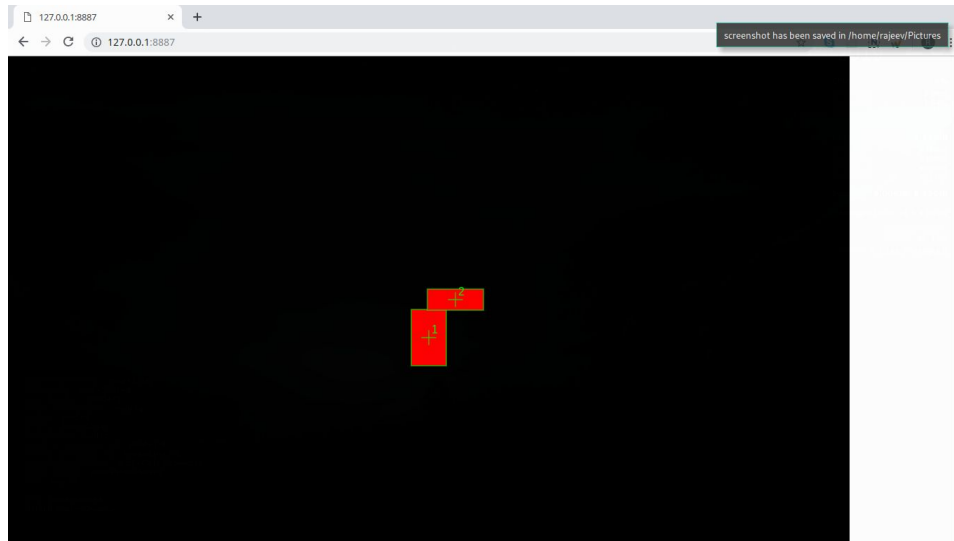


**Step 8:** Add an additional two conditions to check for the vertical distance between the two rectangles to be 0.



```

1  var fixedRect, movingRect;
2
3  function setup() {
4    createCanvas(1200,800);
5    fixedRect = createSprite(600, 400, 50, 80);
6    fixedRect.shapeColor = "green";
7    fixedRect.debug = true;
8    movingRect = createSprite(400,200,80,30);
9    movingRect.shapeColor = "green";
10   movingRect.debug = true;
11 }
12
13 function draw() {
14   background(0,0,0);
15   movingRect.x = World.mouseX;
16   movingRect.y = World.mouseY;
17
18   if (movingRect.x - fixedRect.x < fixedRect.width/2 + movingRect.width/2
19     && fixedRect.x - movingRect.x < fixedRect.width/2 + movingRect.width/2
20     && movingRect.y - fixedRect.y < fixedRect.height/2 + movingRect.height/2
21     && fixedRect.y - movingRect.y < fixedRect.height/2 + movingRect.height/2) {
22     movingRect.shapeColor = "red";
23     fixedRect.shapeColor = "red";
24   }
25   else {
26     movingRect.shapeColor = "green";
27     fixedRect.shapeColor = "green";
28   }
29   drawSprites();
30 }
  
```



**What's next?:** Write more algorithms and put them together to create your own library like p5