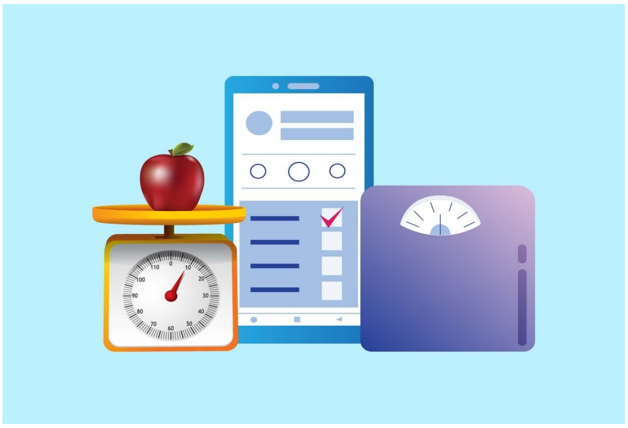
PRO-C23

WhiteHat Jr
Live Online Coding for Kids

CREATING BLUEPRINTS



What is our GOAL for this MODULE?

We used our knowledge about the physics engine and class concept to make the box toppling over another box.

What did we ACHIEVE in the class TODAY?

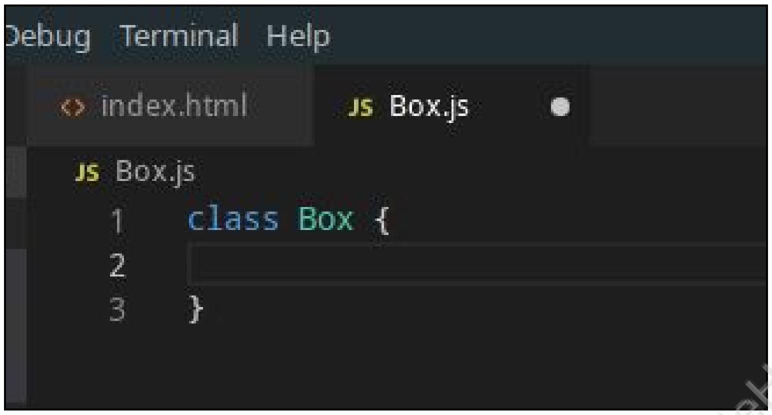
- Created a Box class which creates a template for new objects to be made using the physics engine.
- Created two box objects using the Box class template.
- Tuned the physics engine for properties like density, friction etc. for these objects so that they topple over each other.
- Displayed the rectangle so that it can draw with its orientation.

Which CONCEPTS/ CODING BLOCKS did we cover today?

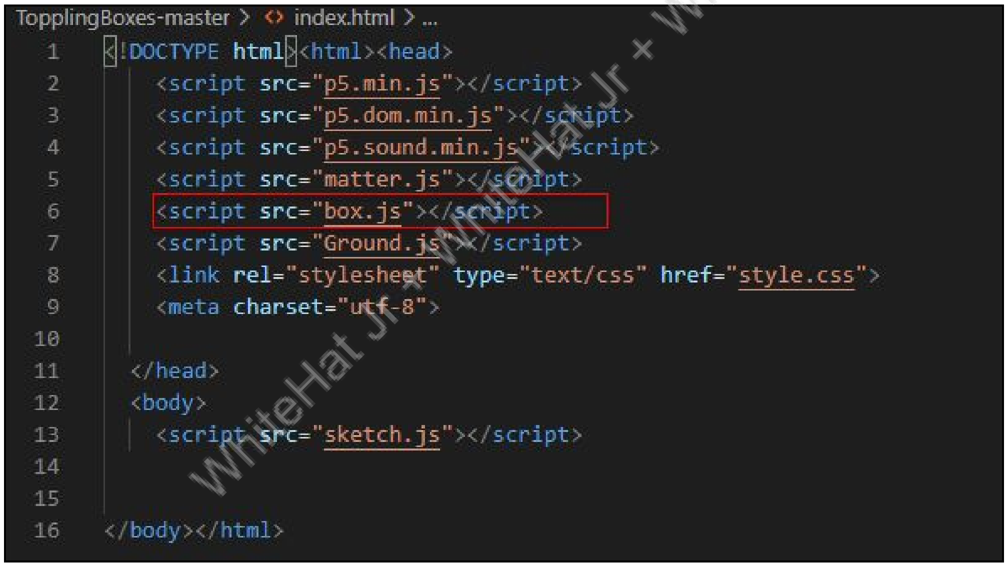
- Creating class object
- Tuning the physics engine to give objects their properties.

How did we DO the activities?

1. Create a new file in the same folder called Box.js.

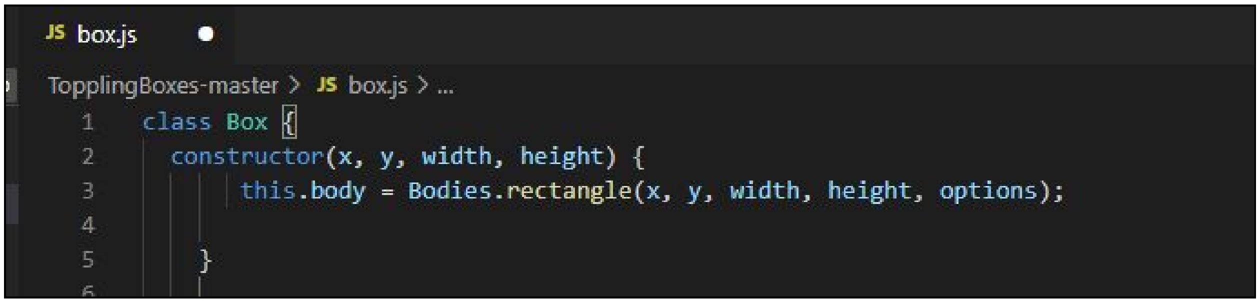


```
class Box {

}
```



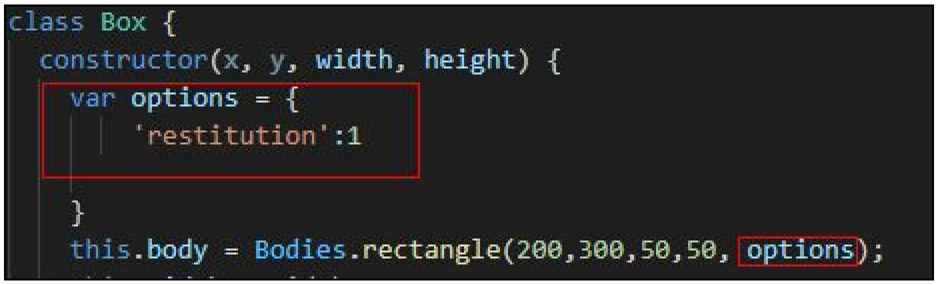
```
<!DOCTYPE html><html><head>
    <script src="p5.min.js"></script>
    <script src="p5.dom.min.js"></script>
    <script src="p5.sound.min.js"></script>
    <script src="matter.js"></script>
    <script src="box.js"></script>
    <script src="Ground.js"></script>
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta charset="utf-8">

  </head>
  <body>
    <script src="sketch.js"></script>


</body></html>
```

2. Create a rectangular body with x, y, width and height.



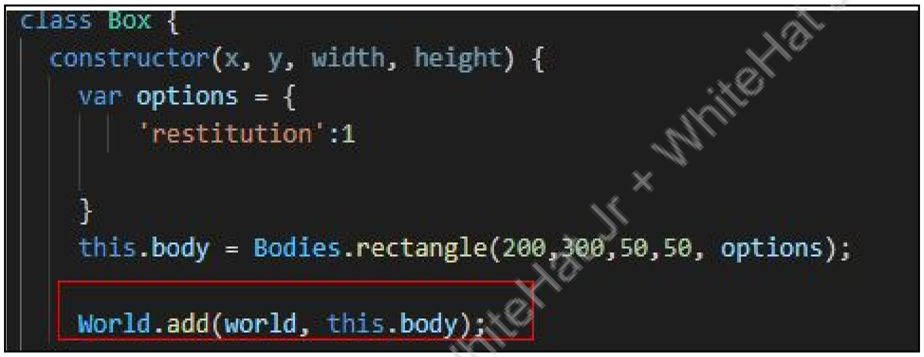
```
class Box {
  constructor(x, y, width, height) {
      this.body = Bodies.rectangle(x, y, width, height, options);

    }
```

3. Add an option, which will finetune the physics engine for the object.



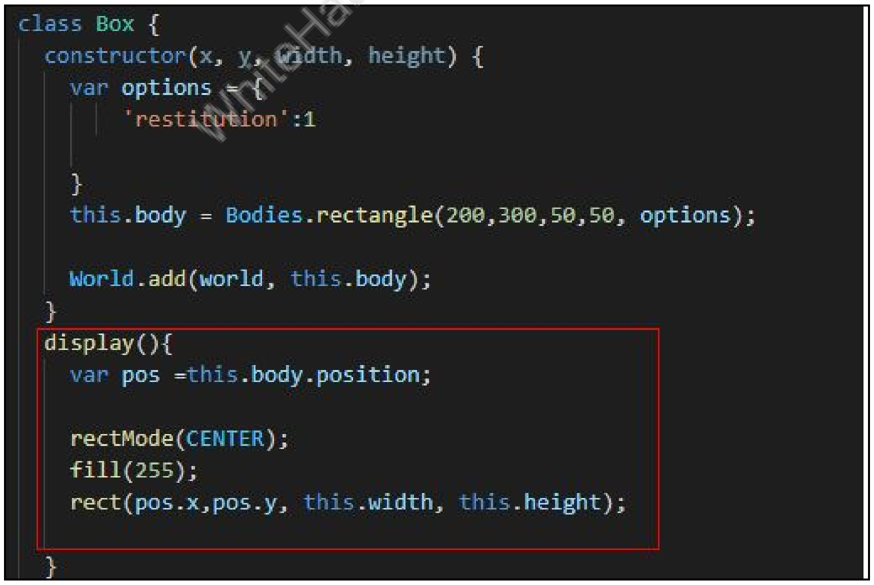
```
class Box {
  constructor(x, y, width, height) {
    var options = {
        'restitution':1

    }
    this.body = Bodies.rectangle(200,300,50,50, options);
```

4. Add this object to the world.



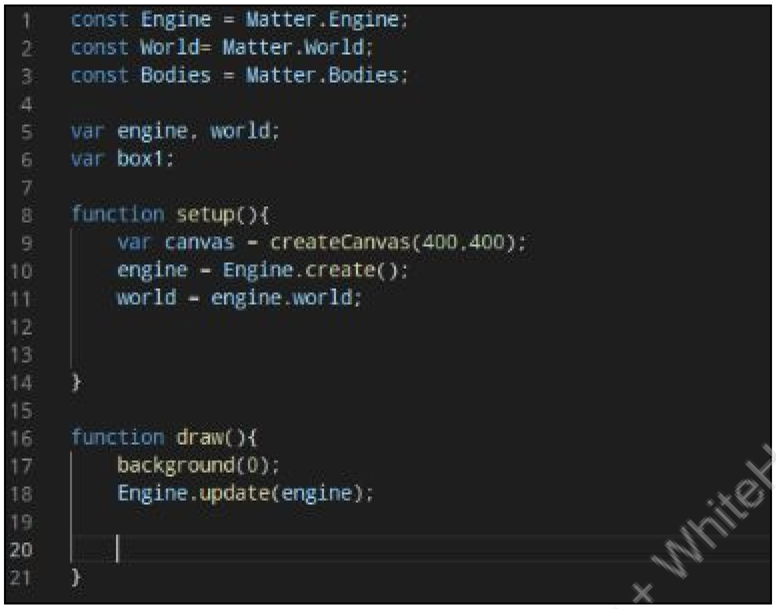
```
class Box {
  constructor(x, y, width, height) {
    var options = {
        'restitution':1

    }
    this.body = Bodies.rectangle(200,300,50,50, options);

    World.add(world, this.body);
```

5. Display this object using a **display()** function.



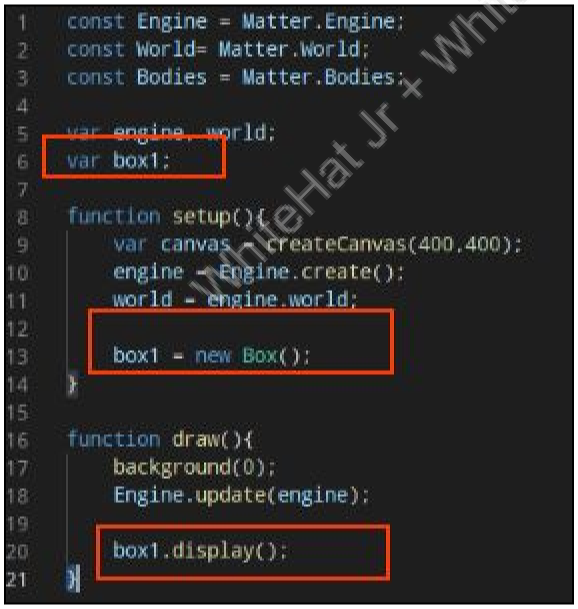
```
class Box {
  constructor(x, y, width, height) {
    var options = {
        'restitution':1

    }
    this.body = Bodies.rectangle(200,300,50,50, options);

    World.add(world, this.body);
  }
  display(){
    var pos =this.body.position;

    rectMode(CENTER);
    fill(255);
    rect(pos.x,pos.y, this.width, this.height);

  }
```

6. In the sketch.js file, remove all the statements associated with creating the bodies.

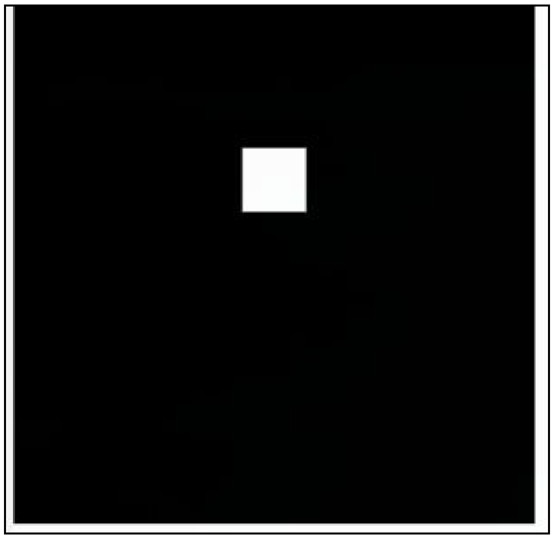


```
const Engine = Matter.Engine;
const World= Matter.World;
const Bodies = Matter.Bodies;

var engine, world;
var box1;

function setup(){
    var canvas = createCanvas(400,400);
    engine = Engine.create();
    world = engine.world;


}

function draw(){
    background(0);
    Engine.update(engine);

    
}
```

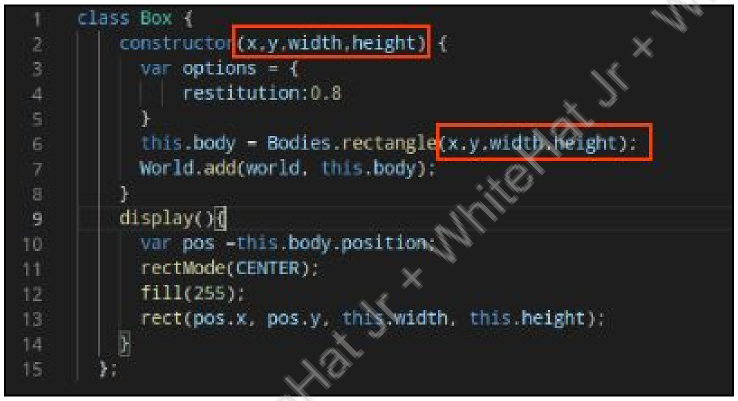
7. Create a new object and display it with just two statements.



```
const Engine = Matter.Engine;
const World= Matter.World;
const Bodies = Matter.Bodies;

var engine, world;
var box1;

function setup(){
    var canvas = createCanvas(400,400);
    engine = Engine.create();
    world = engine.world;

    box1 = new Box();
}

function draw(){
    background(0);
    Engine.update(engine);

    box1.display();
}
```

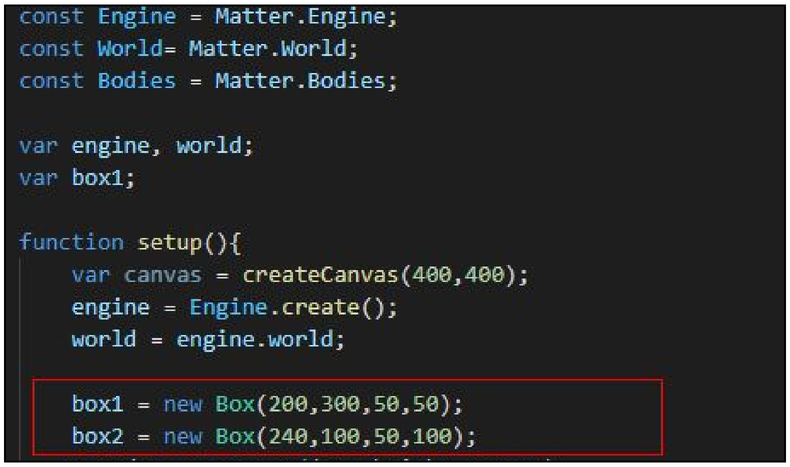


8. Pass the x, y, width and height coordinates to the constructor in the Box class.



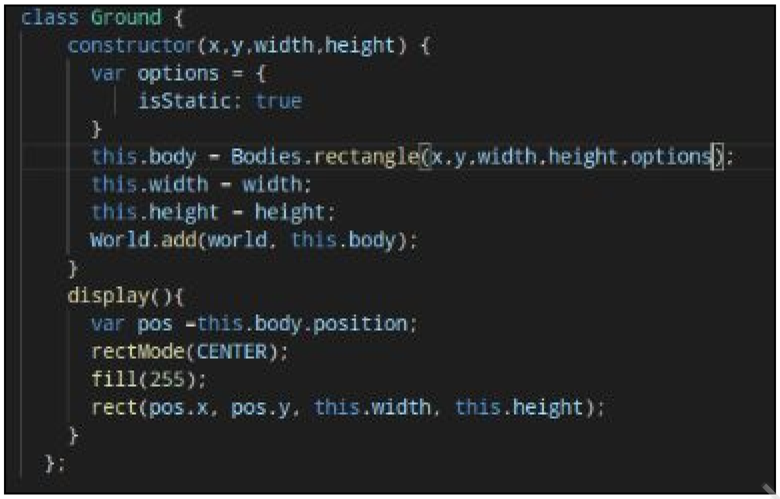
```
class Box {
  constructor(x,y,width,height) {
    var options = {
      restitution:0.8
    }
    this.body = Bodies.rectangle(x,y,width,height);
    World.add(world, this.body);
  }
  display(){
    var pos =this.body.position;
    rectMode(CENTER);
    fill(255);
    rect(pos.x, pos.y, this.width, this.height);
  }
};
```

9. Create the second box object using the Box class.

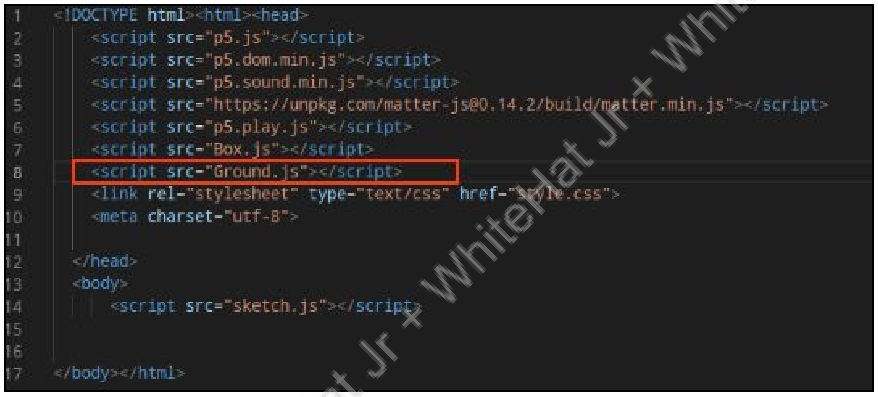


```
const Engine = Matter.Engine;
const World= Matter.World;
const Bodies = Matter.Bodies;

var engine, world;
var box1;

function setup(){
    var canvas = createCanvas(400,400);
    engine = Engine.create();
    world = engine.world;

    box1 = new Box(200,300,50,50);
    box2 = new Box(240,100,50,100);
```

10. Create a Ground class blueprint and then create a ground object using it.

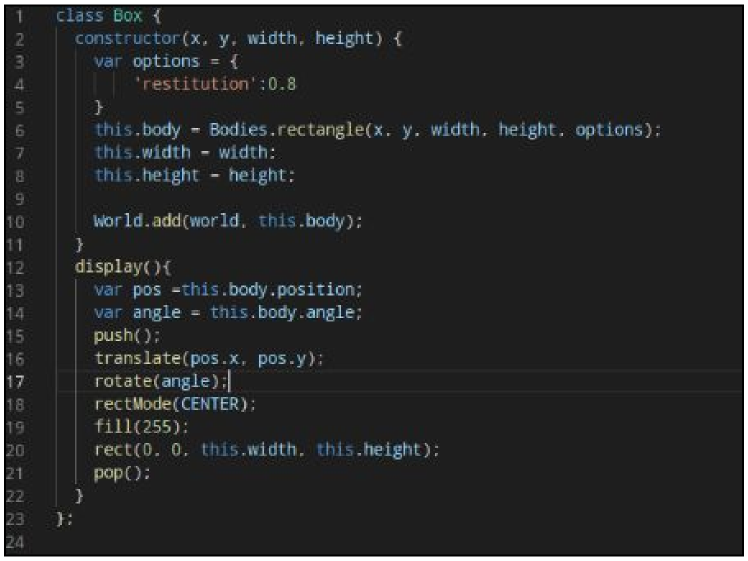


```
class Ground {
    constructor(x,y,width,height) {
      var options = {
        isStatic: true
      }
      this.body = Bodies.rectangle(x,y,width,height,options);
      this.width = width;
      this.height = height;
      World.add(world, this.body);
    }
    display(){
      var pos =this.body.position;
      rectMode(CENTER);
      fill(255);
      rect(pos.x, pos.y, this.width, this.height);
    }
  };
```

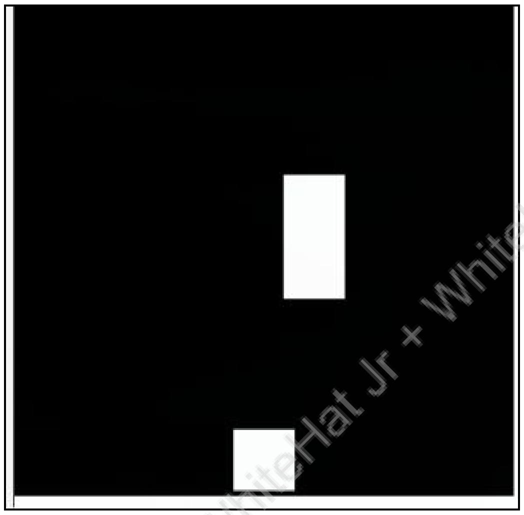


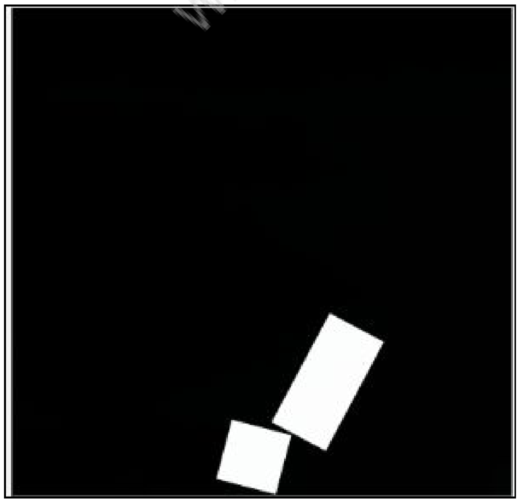
```
<!DOCTYPE html><html><head>
    <script src="p5.js"></script>
    <script src="p5.dom.min.js"></script>
    <script src="p5.sound.min.js"></script>
    <script src="https://unpkg.com/matter-js@0.14.2/build/matter.min.js"></script>
    <script src="p5.play.js"></script>
    <script src="Box.js"></script>
    <script src="Ground.js"></script>
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta charset="utf-8">

  </head>
  <body>
    <script src="sketch.js"></script>

</body></html>
```

11. Store the new translation and rotation setting and then revert back to the old setting when the object is drawn. This is done using push() and pop().
    - push() -> captures the new setting.
    - pop() -> reverts back to the old setting.
    - translate() -> to change the 0 of the axis to a given x and y position.



```
class Box {
  constructor(x, y, width, height) {
    var options = {
        'restitution':0.8
    }
    this.body = Bodies.rectangle(x, y, width, height, options);
    this.width = width;
    this.height = height;

    World.add(world, this.body);
  }
  display(){
    var pos =this.body.position;
    var angle = this.body.angle;
    push();
    translate(pos.x, pos.y);
    rotate(angle);
    rectMode(CENTER);
    fill(255);
    rect(0, 0, this.width, this.height);
    pop();
  }
};
```





12. You can play around with more properties of objects like restitution, density, friction etc.

What's next?

Using what we have learned in this class, we will create the stack of obstacles for the pig in the Angry Birds game.

Extend your knowledge:

1. Go through the following link to know more about classes: https://www.w3schools.com/js/js_object_classes.asp