**Ground class:**

In the Ground class, we will add all the parameters which are needed to create a physics body, we will also create functions or methods for this Ground class. Create the class by using the **class** keyword and name of the class, as Ground. Remember, the **first letter** of the class name is always in **uppercase**.

In this class first, we need to create a **constructor**(). This function is executed when we create the object of the class. In the constructor function, we will pass the **arguments** which are to be given in sketch.js. These are x and y positions and the width and height of the body. When we will create the **object** of the Ground class, we can choose the position and the dimensions of the body. In the constructor() function we need to specify the **options**/properties for the physics body.

We want the ground to remain stationary, property we can use is

**isStatic: true**

In the Ground class create a **show**() function, functions inside the class are called methods, and we don’t need to add the function keyword in front of them.

If we want to give certain properties for one shape, such as color, stroke, and strokeWeight we can encapsulate these in-between push() and pop() statements . The properties added here do not affect other shapes on the canvas. The shape of the ground is a rectangle, so we will create a rectangle using the rect() function and pass the position and dimension of the physics body to it.

fill(“blue”) fills the shape with given color

stroke(“red”) is to give color to the boundary of shape

strokeWeight(7) mentions thickness of boundary

push() to add

pop() to delete or take out.

**Objects** :

In the setup function, we create the objects from the Ground class using the new keyword and assign them to the respective variable.

**applyForce**:

Just like in the real world, if you want to move a body (any object) you have to apply some kind of force to it. Either you push the object or you pull it. In the same way, if we want to move any object in the physics engine we can apply force on the object. We can apply force in a very specific direction such as x-direction then the body will only move in the horizontal direction. Similarly, a force applied in the Y direction will move the body in the vertical direction. We can join the 2 forces as well which will make the object move in both directions. It will be similar to how we throw a ball and make it have a parabolic trajectory

**Matter.js** offers us the function to apply the force on the body, where we can choose the body and points on which we want to apply the force on, direction, and the value of force. This function is called **Matter.Body.applyForce(body,position,force);**

We write as:

**Matter.Body.applyForce(ball,{x:0,y:0},{x:0,y:0});**

Here the first argument is the body, the second is the x & y positions which is the initial amount of force applied on the body and the third is the amount and direction of the force in x and y-direction.

If we apply +ve force in the x-direction, that will move the ball in the right direction and -ve force in the left direction.

Similarly, +ve value of y will move the ball in the downward direction

and -ve value will move in the upward direction.

**Create buttons:**

create buttons on the canvas and attach functions to the buttons.

The **buttons** we are creating are called **image buttons**, so arrow images we saw in the vsc will be displayed on the canvas, and they will work like buttons. First, you create two variables as var btn1, btn2;

Next, in the setup() function, we will create buttons.

Button is created using the **createImg**() function, in the arguments.

Then we will set the **position** and the **size** of the button. After that, we will attach the function with a button using the **mouseClicked**() function. We will pass the function name as an argument in this function.