SCOPE OF VARIABLES

What is our GOAL for this MODULE?

We used our knowledge of variables, functions, loops, game states, etc to reset the game and set up a local environment to run the Trex code locally.

What did we ACHIEVE in the class TODAY?

- Created objects by passing different values in a constructor.
- Changed the scope of variable from local to global for Reset & GameOver Sprite
- Reset game when the reset icon is pressed.

Which CONCEPTS/ CODING BLOCKS did we cover today?

- Passing the values from one function to another.
- Scope of variables.
- Changing game state.

How did we DO the activities?

1. Create a **constructor()** function in the **box.js** file where you can pass values such as position, size, and velocity.

```
class Box
  {
    constructor(x,y,w,h,vx)
    {
      this.x =x;
      this.y =y;
      this.w =w;
      this.h = h;
      this.vx = vx;
    }

  }
```

2. Define the functions to display and move the box object on the canvas.

```
show()
{
  rect(this.x,this.y,this.w,this.h)
}

  move()
{
  this.x = this.x+this.vx;
}

moveup()
{
  this.y = this.y - this.vy;
}

}
```

3. Create an object of the **box** class in the **setup()** function and pass the positions, size, and velocity values by using the **Box()** class.

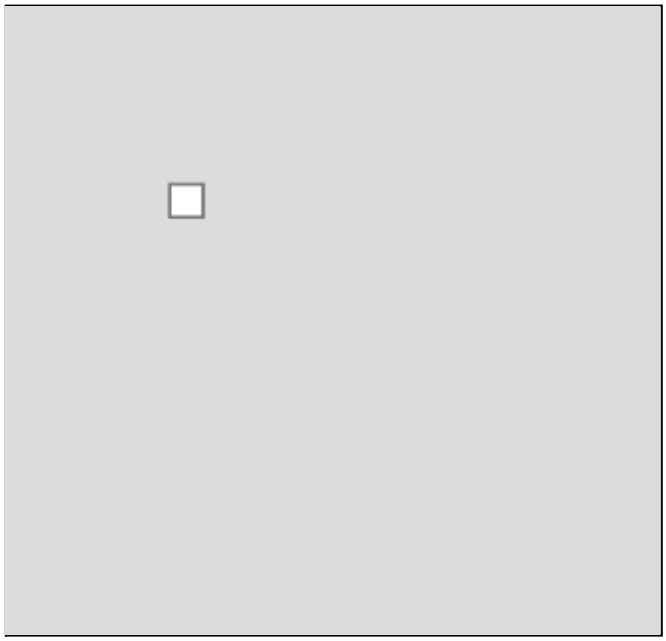
```
var box;

function setup()
{
  createCanvas(400, 400);
  box = new Box(100,200,20,20,2,1);
}
```

4. Call the **show()** and **moveup()** functions using the **box** object in the **draw()** function, which will display our moving box on the canvas.

```
function draw()
{
  background(220);
  box.show();
  box.moveup();
}
```

Output



5. Experiment with the scope of the variable and the correct place to declare them as per their role in the game. When the variables are used out of scope, it gives a reference error. Thus variables have a scope and they live and die inside the scope.

```
var PLAY = 1;
var END = 0;
var gameState = PLAY;

var trex, trex_running, trex_collided;
var ground, invisibleGround, groundImage;

var cloudsGroup, cloudImage;
var obstaclesGroup, obstacle1, obstacle2, obstacle3,
obstacle4, obstacle5, obstacle6;

var score;
var gameOverImg,restartImg
var jumpSound , checkPointSound, dieSound
```

6. Add a reset icon and add functionality to it. Use the **mousePressedOver()** instruction to detect if the mouse is pressed over the reset sprite and print a message when it is pressed.


```
    obstaclesGroup.setVelocityXEach(0);
    cloudsGroup.setVelocityXEach(0);
  }

  //stop trex from falling down
  trex.collide(invisibleGround);

  if(mousePressedOver(restart)) {
      console.log("Restart the Game")
    }


  drawSprites();
}
```

7. Create a reset() function which resets everything in the game to its original state.

```
  //stop trex from falling down
  trex.collide(invisibleGround);

  if(mousePressedOver(restart)) {
      console.log("Restart the Game")
    }


  drawSprites();
}

function reset(){

}
```

8. Write code for the reset function.

```
function reset(){
  gameState = PLAY;
  gameOver.visible = false;
  restart.visible = false;
}


function spawnObstacles(){
 if (frameCount % 60 === 0){
   var obstacle = createSprite(600,165,10,40);
   obstacle.velocityX = -(6 + 3*score/100);
```

9. Destroy the obstacles when the game is reset.

```
function reset(){
  gameState = PLAY;
  gameOver.visible = false;
  restart.visible = false;

  obstaclesGroup.destroyEach();
  cloudsGroup.destroyEach();
}


function spawnObstacles(){
 if (frameCount % 60 === 0){
   var obstacle = createSprite(600,165,10,40);
```

10. Change the trex collided animation to Trex running.

```
function reset(){
  gameState = PLAY;
  gameOver.visible = false;
  restart.visible = false;

  obstaclesGroup.destroyEach();
  cloudsGroup.destroyEach();

  trex.changeAnimation("running", trex_running);
}
```

11. Move the condition where we checked the mouse pressed over reset icon inside game end state to fix the restarting game issue even when the reset icon was

invisible. Reset the score.

```
  drawSprites();
}

function reset(){
  gameState = PLAY;
  gameOver.visible = false;
  restart.visible = false;
  trex.changeAnimation("running", trex_running);

  obstaclesGroup.destroyEach();
  cloudsGroup.destroyEach();
  score = 0;

}
```

12. Update the score count based on the frame rate to fix the issue of score resetting to 0 temporarily and then starting from the old score.

```
if(gameState === PLAY){
  //move the
  gameOver.visible = false;
  restart.visible = false;

  ground.velocityX = -(4 + 3* score/100)
  //scoring
  score = score + Math.round(getFrameRate()/60);

  if(score>0 && score%100 === 0){
    checkPointSound.play()
  }

  if (ground.x < 0){
    ground.x = ground.width/2;
  }
```

What's next?

We'll learn to write code on the local machine.

Extend Your Learning:

1. Learn more about the scope of variables.