Arrays and Bird Trajectory

What is our GOAL for this MODULE?

The goal for this module is to explore the different ways in which the data can be stored in javascript.

What did we ACHIEVE in the class TODAY?

We learned the different ways in which data can be stored in javascript. We explored the array data structure to design the bird's trajectory after it has been launched. We used the concept of game state to stop the bird from being draggable after the bird is launched.

Which CONCEPTS/ CODING BLOCKS did we cover today?

- Array data structure
- Design the bird’s trajectory
- Game states

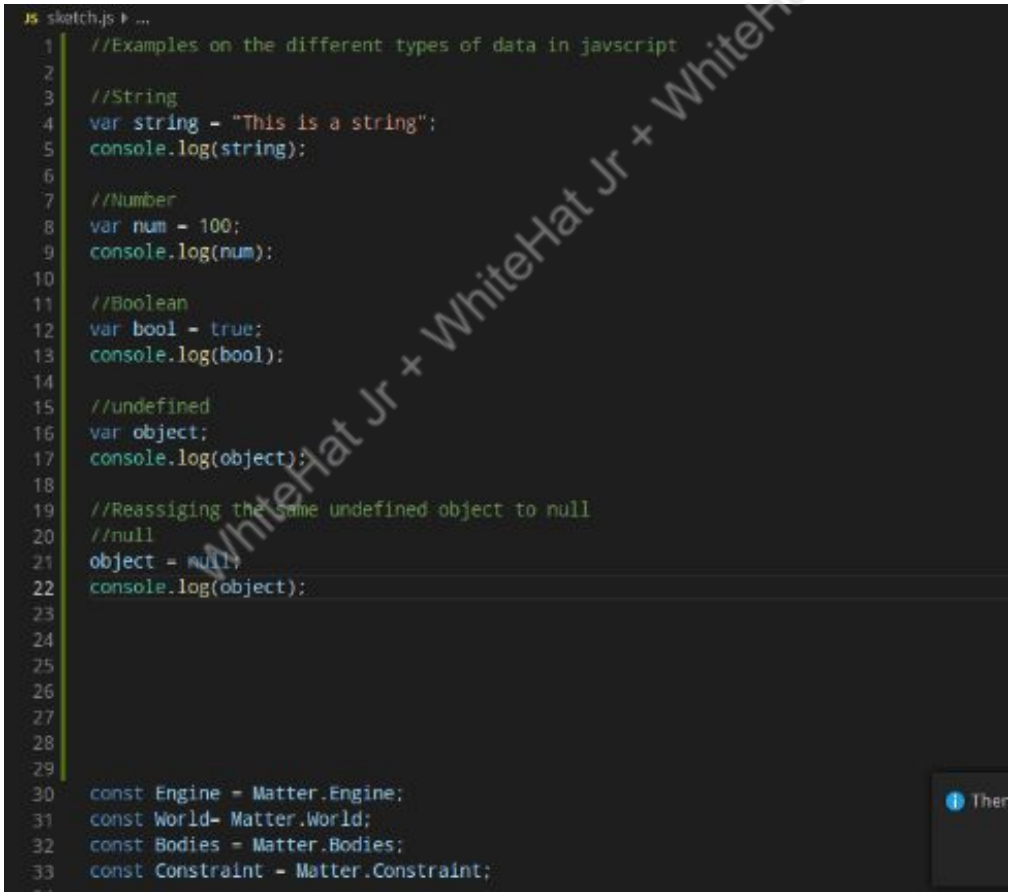
How did we DO the activities?

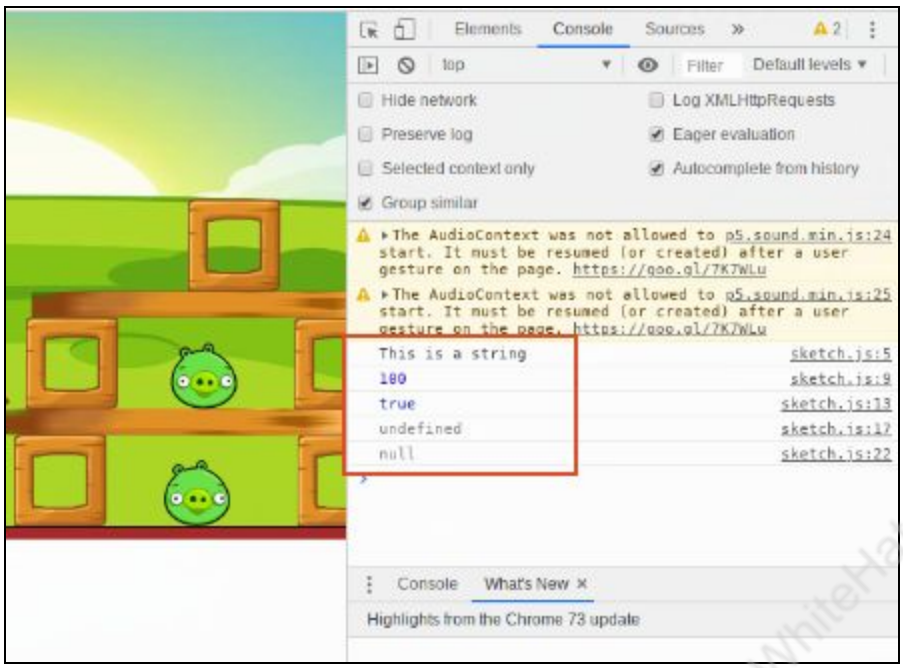
We stored data most commonly inside variables. In Javascript, a variable holds different types of data:

- String: These are sequence of characters stored inside quotes
- Number: Any mathematical number
- Boolean values: true and false values

There are also two special types of data:
null -> it means nothing or empty
undefined -> it means that no value has been assigned to a variable.

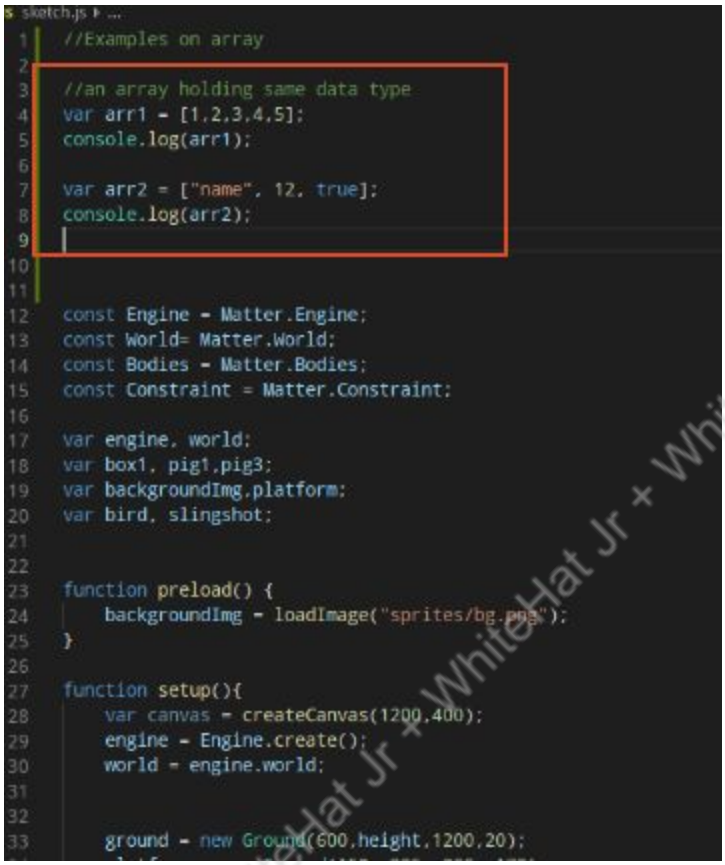
A single variable javascript can hold any of these types of data. Different data can also be reassigned to a single variable.





The disadvantage of storing data in a variable is, we can store only one value at a time. Thus, it is too cumbersome to create a variable for each value. It would also make our code unreadable.

To avoid this problem, Javascript and most other languages have different data structures to hold multiple values. One popular data structure to hold multiple values is called an "array". An array is created inside square brackets and can store a list of the same or different types of data separated by a comma. Example:



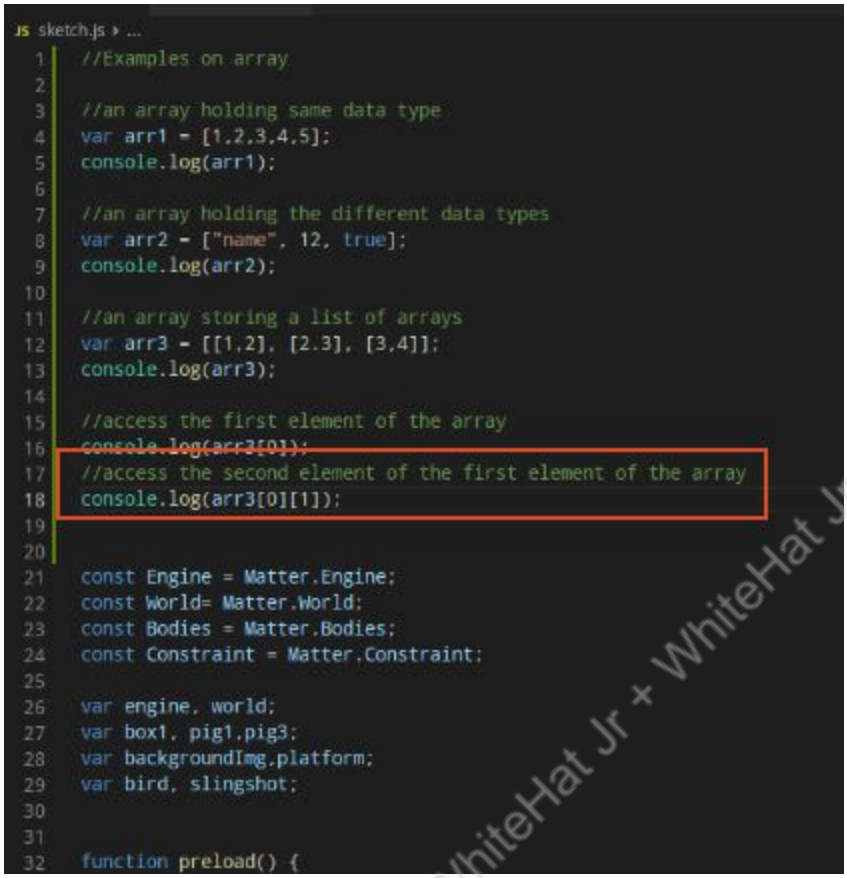
An array stores a list of arrays! Example:

```
sketch.js ▸ ...
1  //Examples on array
2
3  //an array holding same data type
4  var arr1 = [1,2,3,4,5];
5  console.log(arr1);
6
7  //an array holding the different data types
8  var arr2 = ["name", 12, true];
9  console.log(arr2);
10
11 //an array storing a list of arrays
12 var arr3 = [[1,2], [2.3], [3,4]];
13 console.log(arr3);
14
15
16 const Engine = Matter.Engine;
17 const World= Matter.World;
18 const Bodies = Matter.Bodies;
19 const Constraint = Matter.Constraint;
20
21 var engine, world;
22 var box1, pig1,pig3;
23 var backgroundImg,platform;
24 var bird, slingshot;
25
26
27 function preload() {
```

Each value in the array is indexed by a number. The first value has an index of '0', the second value has an index of '1' and so on. (Counting always starts from 0 in most computer languages). If we want to access the first element of our arr3 array, we can access it by using arr3[0].

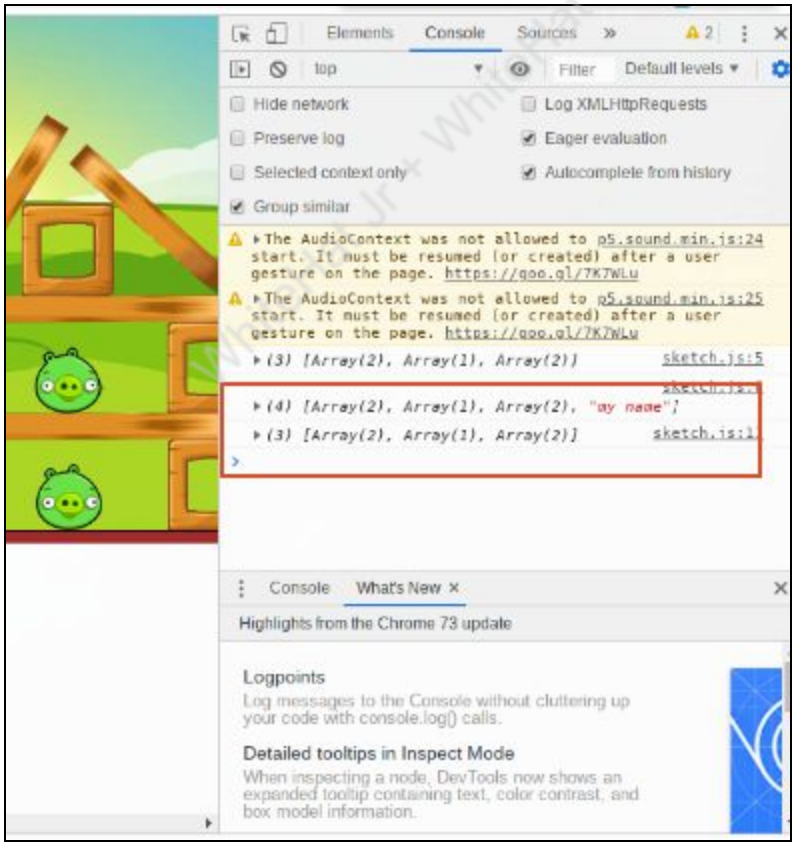
```
//Examples on array

//an array holding same data type
var arr1 = [1,2,3,4,5];
console.log(arr1);

//an array holding the different data types
var arr2 = ["name", 12, true];
console.log(arr2);

//an array storing a list of arrays
var arr3 = [[1,2], [2.3], [3,4]];
console.log(arr3);

//access the first element of the array
console.log(arr3[0]);

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

var engine, world;
var box1, pig1,pig3;
var backgroundImg,platform;
var bird, slingshot;
```

To access the first element inside the first element of the array. We did this by adding a sub-index like this arr3[0][0]. The second element of the first element in the array can be accessed with arr3[0][1].



A new value can be pushed inside an array by using array.push(). Similarly, the last values can be pooped out of the arrays using array.pop().

```
sketch.js ▸ ...
1  //Examples on array
2
3  //an array storing a list of arrays
4  var arr3 = [[1,2], [2,3], [3,4]];
5  console.log(arr3);
6
7  arr3.push("my name");
8  console.log(arr3);
9
10 arr3.pop();
11 console.log(arr3);
12
13
14 const Engine = Matter.Engine;
15 const World= Matter.World;
16 const Bodies = Matter.Bodies;
17 const Constraint = Matter.Constraint;
18
19 var engine, world;
20 var box1, pig1,pig3;
21 var backgroundImg,platform;
22 var bird, slingshot;
23
24
25 function preload() {
26     backgroundImg = loadImage("sprites/bg.png");
27 }
28
```

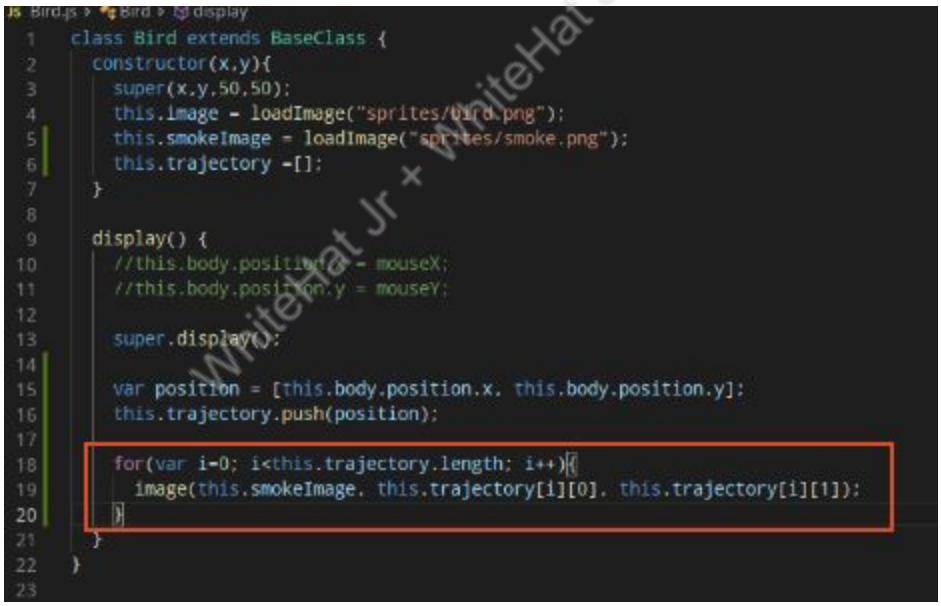


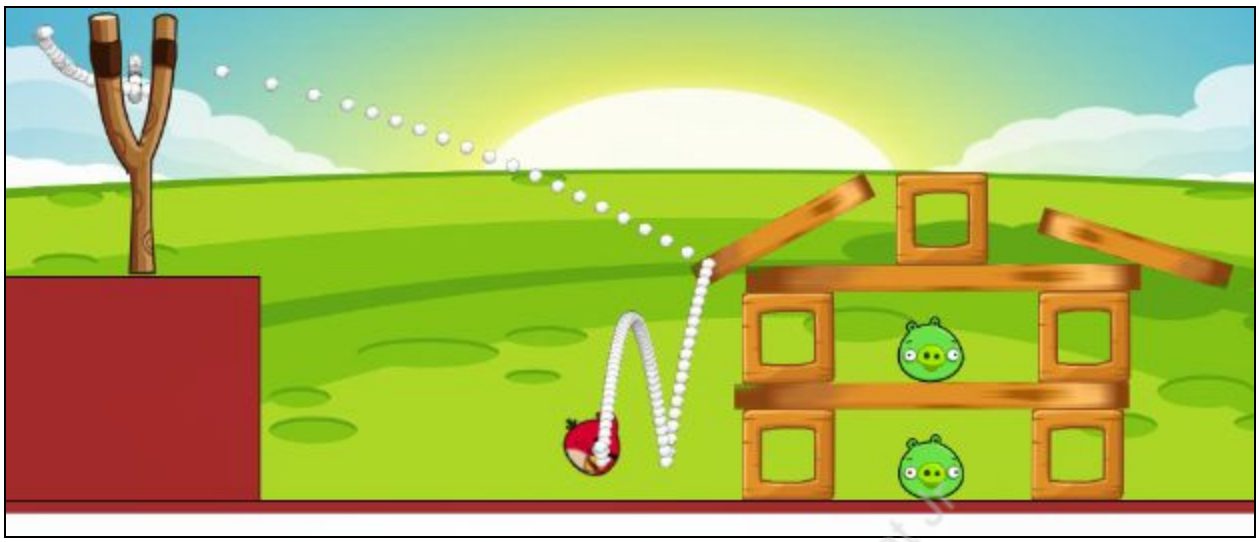
To load the smoke image in the trajectory of the bird:

To create an empty array to store the positions where the bird had moved:

Note: Store the x and y position for each point the bird moves in its trajectory.
The display() function gets called in each frame.

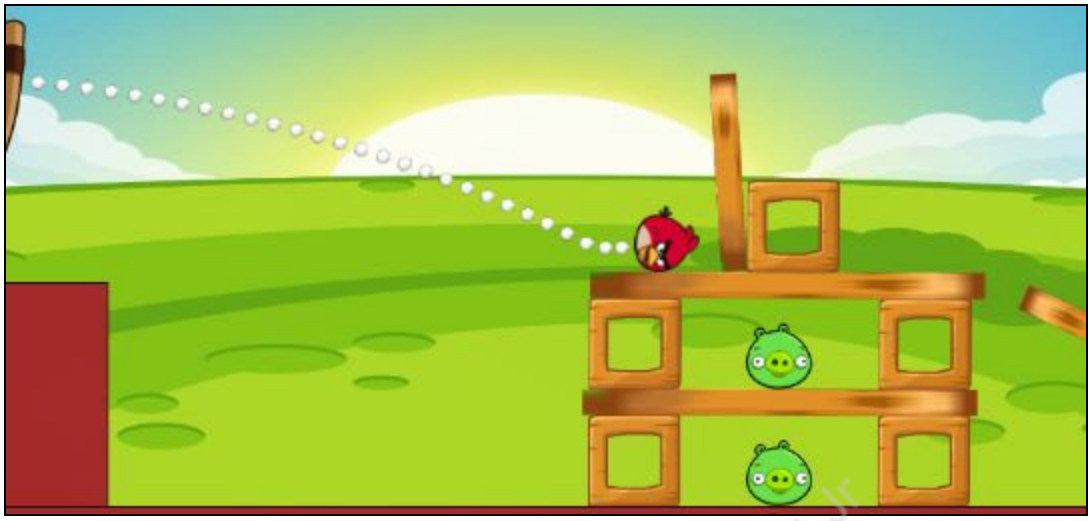
For each element in the array we wanted to draw the smoke image at those positions: We used For loop to move over the arrays and image() function to draw the image at given points.





There was smoke even when the bird had stopped. In order to stop this action, we used if conditions.

```
Bird.js > ...
class Bird extends BaseClass {
  constructor(x,y){
    super(x,y,50,50);
    this.image = loadImage("sprites/bird.png");
    this.smokeImage = loadImage("sprites/smoke.png");
    this.trajectory =[];
  }

  display() {
    //this.body.position.x = mouseX;
    //this.body.position.y = mouseY;

    super.display();

    if(this.body.velocity.x > 10 && this.body.position.x > 200){
      var position = [this.body.position.x, this.body.position.y];
      this.trajectory.push(position);
    }


    for(var i=0; i<this.trajectory.length; i++){
      image(this.smokeImage, this.trajectory[i][0], this.trajectory[i][1]);
    }
  }
}

```



We identified two problems in the game:

1. We could still press space and get the angry bird back to the slingshot. This created multiple trajectories. We did not want this feature in a fair game.
2. Our bird followed the mouse even after the collision! This allowed the player to destroy the pigs even after they were missed.

We declared gameState and used it in mouseDragged and mouseReleased function to disable the dragging of the bird after launch.

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

var engine, world;
var box1, pig1,pig3;
var backgroundImg,platform;
var bird, slingshot;

var gameState = "onSling";

function preload() {
    backgroundImg = loadImage("sprites/bg.png");
}

function setup(){
    var canvas = createCanvas(1200,400);
    engine = Engine.create();
    world = engine.world;


    ground = new Ground(600,height,1200,20);
    platform = new Ground(150, 305, 300, 170);

    box1 = new Box(700,320,70,70);
    box2 = new Box(920,320,70,70);
    pig1 = new Pig(810, 350);
```

```
    box3.display();
    box4.display();
    pig3.display();
    log3.display();

    box5.display();
    log4.display();
    log5.display();

    bird.display();
    platform.display();
    //log6.display();
    slingshot.display();
}

function mouseDragged(){
    if (gameState!=="launched"){
        Matter.Body.setPosition(bird.body, {x: mouseX , y: mouseY});
    }
}


function mouseReleased(){
    slingshot.fly();
    gameState = "launched";
}

function keyPressed(){
    if(keyCode === 32){
        // slingshot.attach(bird.body);
    }
}
```


What's NEXT?

In the next class, you will be learning about JSON and making API calls.

EXTEND YOUR KNOWLEDGE:

You can learn about different javascript methods through following link
https://www.w3schools.com/js/js_array_methods.asp