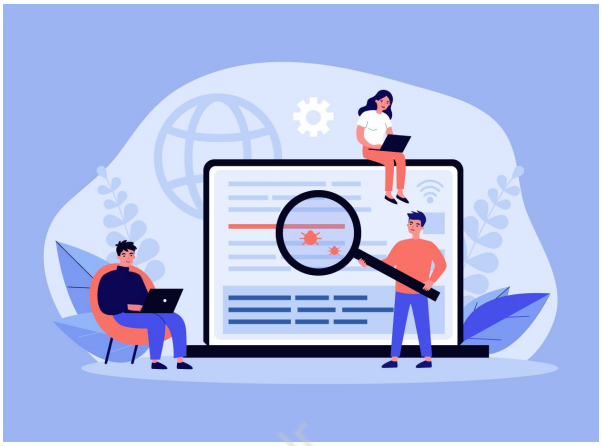
Debugging Tips and Tricks



What is our GOAL for this MODULE?

The goal for this module is to learn about debugging.

What did we ACHIEVE in the class TODAY?

We learned tips and tricks to minimize errors and bugs in the code and also how to debug the errors through the different debugging techniques. We debugged the Angry Bird trajectory when shooting multiple times

Which CONCEPTS/ CODING BLOCKS did we cover today?

- Different types of common errors.

How did we DO the activities?

Debugging is a series of techniques that programmers use to solve a bug in their code. When debugging, check the console log for errors. If there are any error messages in the console, they often tell you what happened and why this error occurred or point you in that direction. You can open the console in your browser by pressing Ctrl + Shift + J. Often you can fix the errors in your code by reading these error messages.

The most common errors are:
1) **Typos**: You accidentally misspelled a variable or a function name which the computer doesn't understand.
2) **Incorrect Use of function**: You used a function in a way it wasn't intended to be used.
3) **Using variables outside their scope**: If you are using variables outside their scope, the computer wouldn't know the value of these variables.

The techniques which will help you identify what is causing the bug or which section of your code is leading to this bug:
1) **Commenting sections of your code**: You can comment on certain sections of your code to simplify your code and check if your code still throws errors. In this way, you can narrow down to the part of the code which is causing this error.
2) **Printing values of variables in the console:** You can print the values of critical variables in the console to visually see how they are changing in the code. If they are not changing as you intend them to, there is something unexpected happening. You can manipulate the variable values to identify what is happening.
3) **Print messages in your code:** You can print messages in your code to visually understand how the code is running. For example; you can print messages in the console to understand if while executing the code, an if block is getting executed or the else block.

You can also combine all the 3 steps. Before running the code, one should always have what they expect in their mind and see the difference between what's actually happening.

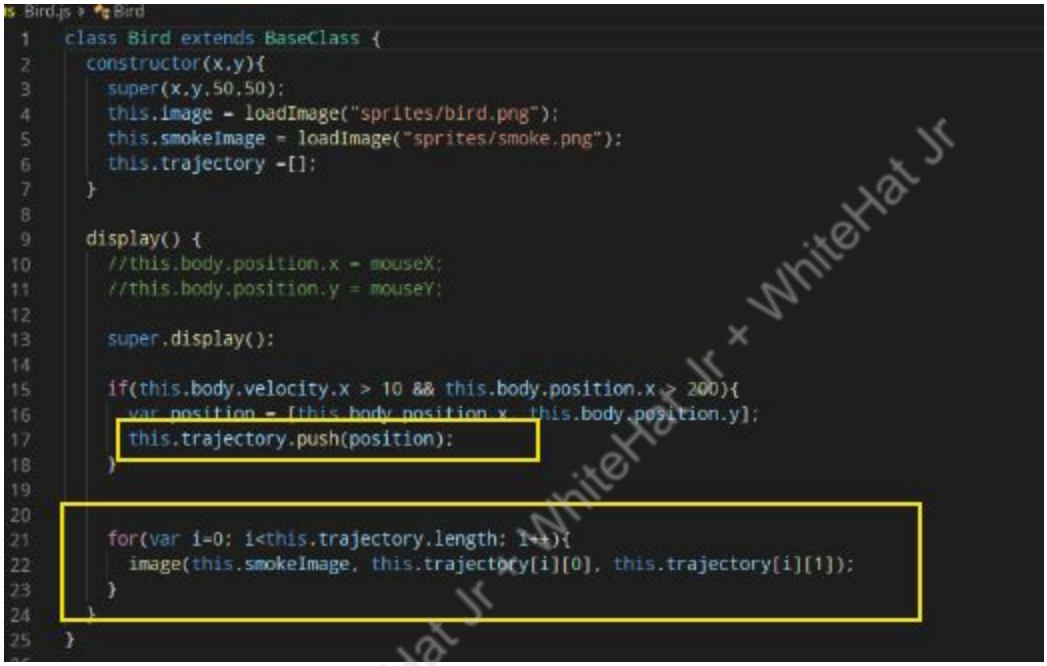
In the angry bird game, we didn't allow the player to take multiple shots at the pigs. We changed the code to allow the player to take as many shots at the pigs as they want.

```
home ▸ rajeev ▸ Lessons ▸ AngryBirdsStage8 ▸ JS sketch.js ▸ keyPressed
72
73      box5.display();
74      log4.display();
75      log5.display();
76
77      bird.display();
78      platform.display();
79      //log6.display();
80      slingshot.display();
81  }
82
83  function mouseDragged(){
84      //if (gameState!=="launched"){
85          Matter.Body.setPosition(bird.body, {x: mouseX , y: mouseY});
86      //}
87  }
88
89
90  function mouseReleased(){
91      slingshot.fly();
92      gameState = "launched";
93  }
94
95  function keyPressed(){
96      if(keyCode === 32){
97          slingshot.attach(bird.body);
98      }
99  }
100
101 async function getBackgroundImg(){
102     var response = await fetch("http://worldtimeapi.org/api/timezone/Asia/Kolkata");
103     var responseJSON = await response.json();
104
105     var datetime = responseJSON.datetime;
106     var hour = datetime.slice(11,13);
```

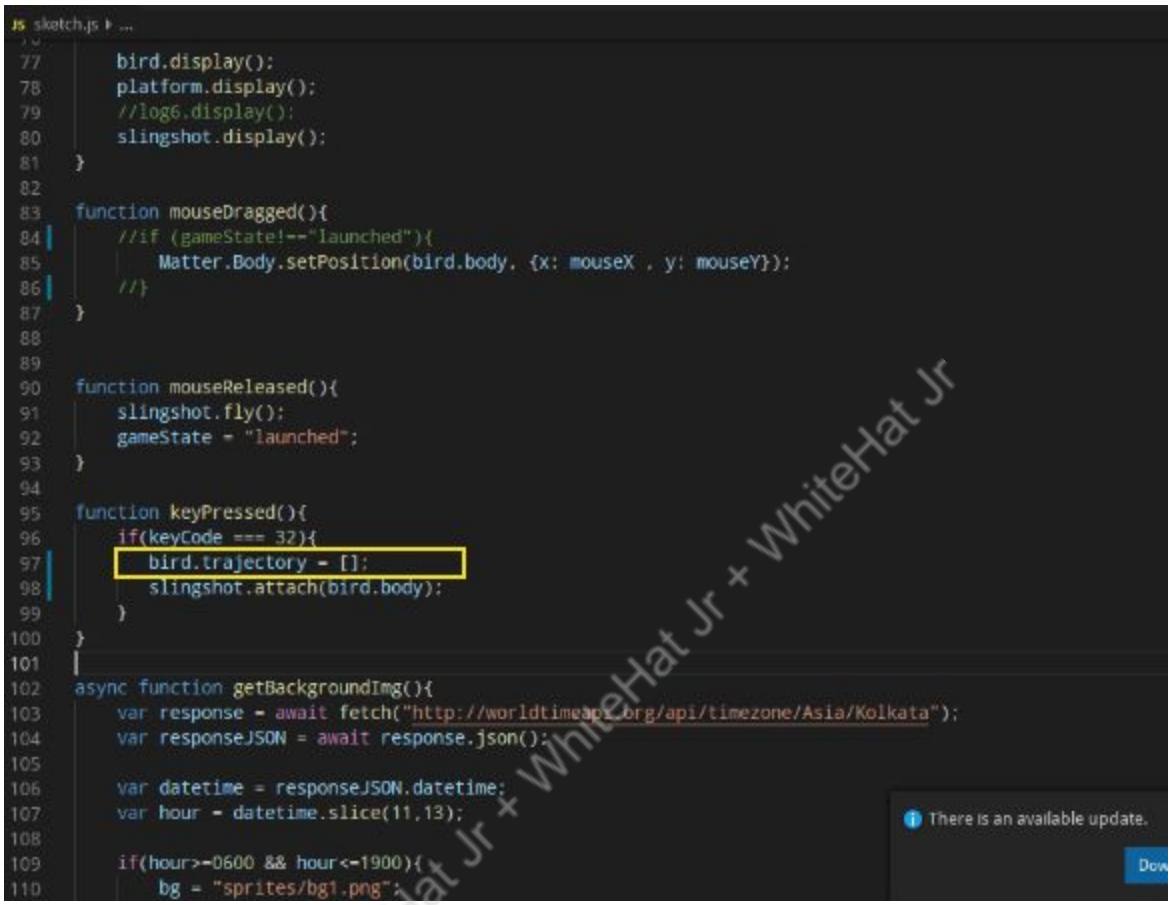
After you run the code, you realise that there were multiple trajectories of the bird on the screen:

1. When we reset the bird back to the slingshot, we wanted the previous trajectory to be deleted.
2. The bird swings rapidly when it is reset. This was not desired. Also, it caused an additional trajectory mark.

Thus, you debugged the code.



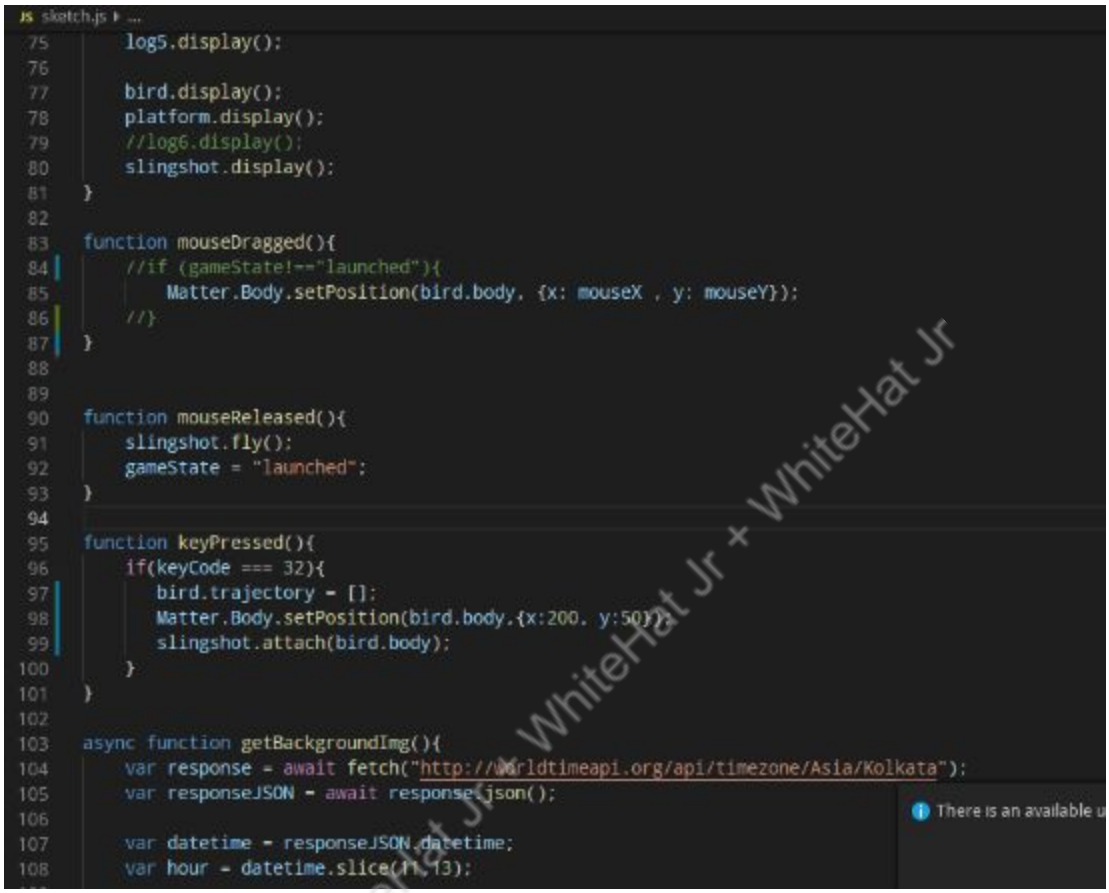
All the positions of the birds were stored in the array property of the Bird called trajectory. We did empty this array whenever the space key was pressed.



```
sketch.js ▸ ...
77      bird.display();
78      platform.display();
79      //log6.display();
80      slingshot.display();
81  }
82
83  function mouseDragged(){
84      //if (gameState!=="launched"){
85          Matter.Body.setPosition(bird.body, {x: mouseX , y: mouseY});
86      //}
87  }
88
89
90  function mouseReleased(){
91      slingshot.fly();
92      gameState = "launched";
93  }
94
95  function keyPressed(){
96      if(keyCode === 32){
97          bird.trajectory = [];
98          slingshot.attach(bird.body);
99      }
100 }
101
102 async function getBackgroundImg(){
103     var response = await fetch("http://worldtimeapi.org/api/timezone/Asia/Kolkata");
104     var responseJSON = await response.json();
105
106     var datetime = responseJSON.datetime;
107     var hour = datetime.slice(11,13);
108
109     if(hour>=0600 && hour<=1900){
110         bg = "sprites/bg1.png";
```

There is an available update.

Then you noticed that the bird swings widely when attached to the slingshot.
Thus, you reset the position of the bird when space was pressed to the place near the slingshot. We used Matter.Body.setPosition to position the bird.



```
sketch.js ▸ ...
    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){
        bird.trajectory = [];
        Matter.Body.setPosition(bird.body,{x:200, y:50});
        slingshot.attach(bird.body);
    }
}

async function getBackgroundImg(){
    var response = await fetch("http://worldtimeapi.org/api/timezone/Asia/Kolkata");
    var responseJSON = await response.json();

    var datetime = responseJSON.datetime;
    var hour = datetime.slice(11,13);
```


Thus, you debugged the code for your game.

What's NEXT?

In the next class, you will be learning about real time databases.

EXTEND YOUR KNOWLEDGE:

This document contains a detailed description of debugging trick. You can explore it to learn more about it
https://jonskeet.uk/csharp/debugging.html