

Memory leak and Switch Statement



What is our GOAL for this MODULE?

In this lesson, we learned to solve the memory leak problem and the usage of string concatenation. Also introduced to switch statements.

What did we ACHIEVE in the class TODAY?

- Created a Vowel checker using Switch Statements
- Corrected the memory leak problem in T rex.
- Used switch statements to spawn different obstacles randomly.
- Designed a simple scoring system.
- Used string concatenation to display score.

Which CONCEPTS/ CODING BLOCKS did we cover today?

- String concatenation
- Scoring system
- Switch statements
- Correcting memory leak

How did we DO the activities?

1. Practice the switch statements, It is used to control the flow of a program, it is also known as a type of conditional statements. Switch statements allow the execution of a specific code block based on the value passed to it. You developed a code to check whether an alphabet is a vowel or not.

```
var input, heading;

function setup() {

  createCanvas(300, 200);
  background(178,255,102);

  input = createInput();
  input.position(5, 60);

  heading = createElement('h4', 'Enter any alphabet:');
  heading.position(5, 20);

  textAlign(CENTER);
  textSize(50);
}
```

```
function draw() {

  const value = input.value();
  switch (value) {

    case 'a':
      console.log("Vowel");
      break;

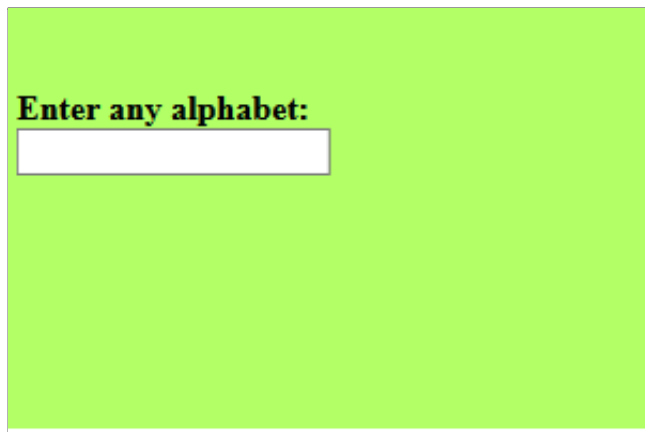
    case 'e':
      console.log("Vowel");
      break;

    case 'i':
      console.log("Vowel");
      break;

    case 'o':
      console.log("Vowel");
      break;

    case 'u':
      console.log("Vowel");
      break;

    default:
      console.log("Please enter any character");
  }
}
```



2. Use switch statements in T rex to give different obstacle images to the same sprite.
3. Start with fixing the memory leak issue: Even after sprites are gone out of canvas, it still occupies space in the memory, which eventually makes the game slow or crash. This is known as the Memory Leak issue. To fix this issue:
 - We need to remove each sprite manually
 - Or make them destroy by giving them a lifetime.
 - We assigned lifetime to each cloud variable which is getting created.
 - Formula for lifetime: $\text{Time} = \text{Distance} / \text{Speed}$; $400 / 3 = 134$

```
function spawnClouds() {  
  //write code here to spawn the clouds  
  if (frameCount % 60 === 0) {  
    cloud = createSprite(600,300,40,10);  
    cloud.drawImage(cloudImage);  
    cloud.y = Math.round(random(280,320));  
    cloud.scale = 0.4;  
    cloud.velocityX = -3;  
  
    //assign lifetime to the variable  
    cloud.lifetime = 134;  
  
    //adjust the depth  
    cloud.depth = trex.depth;  
    trex.depth = trex.depth + 1;  
  }  
}
```

4. Practice string concatenation, When any text information is stored in a computer, it is written inside quotes "_" and called a String.
5. Use string concatenation, two strings / numbers & string can be joined together using + sign. Like this: "Hello" and "World":

```
invisibleGround = createSprite(200,190,400,10);
invisibleGround.visible = false;

console.log("Hello"+"World");

}

function draw() {
  background(180);

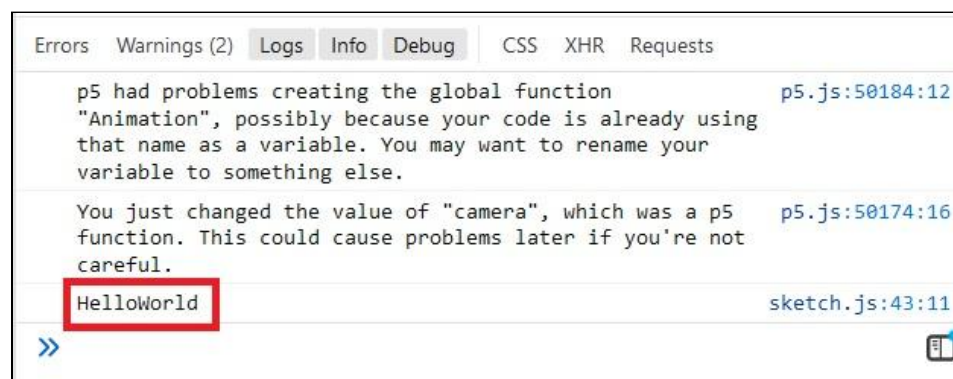
  if(keyDown("space") && trex.y>=100) {
    trex.velocityY = -10;
  }

  trex.velocityY = trex.velocityY + 0.8

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

  trex.collide(invisibleGround);
```

OUTPUT:



- Example of a string & number concatenation:

```
invisibleGround = createSprite(200,190,400,10);
invisibleGround.visible = false;

console.log("Hello"+5);

}

function draw() {
  background(180);

  if(keyDown("space") && trex.y>=100) {
    trex.velocityY = -10;
  }

  trex.velocityY = trex.velocityY + 0.8

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

  trex.collide(invisibleGround);
```

OUTPUT:

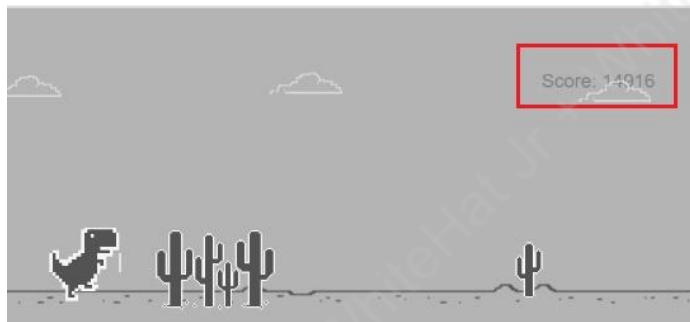
Errors	Warnings (2)	Logs	Info	Debug	CSS	XHR	Requests
p5 had problems creating the global function "Animation", possibly because your code is already using that name as a variable. You may want to rename your variable to something else.					p5.js:50184:12		
You just changed the value of "camera", which was a p5 function. This could cause problems later if you're not careful.					p5.js:50174:16		
Hello5					sketch.js:43:11		

>>

6. Build a simple scoring system. We used the **frameCount** as the score., as frame count keeps increasing as the game progresses. String concatenation concept is used to display the score:

```
console.log("Hello" + 5);  
  
score = 0;  
}  
  
function draw() {  
  background(180);  
  text("Score: " + score, 500, 50);  
  score = score + Math.round(frameCount/60);  
}
```

OUTPUT:



7. Create an empty function called **spawnObstacles** and call it inside the **draw()** function:

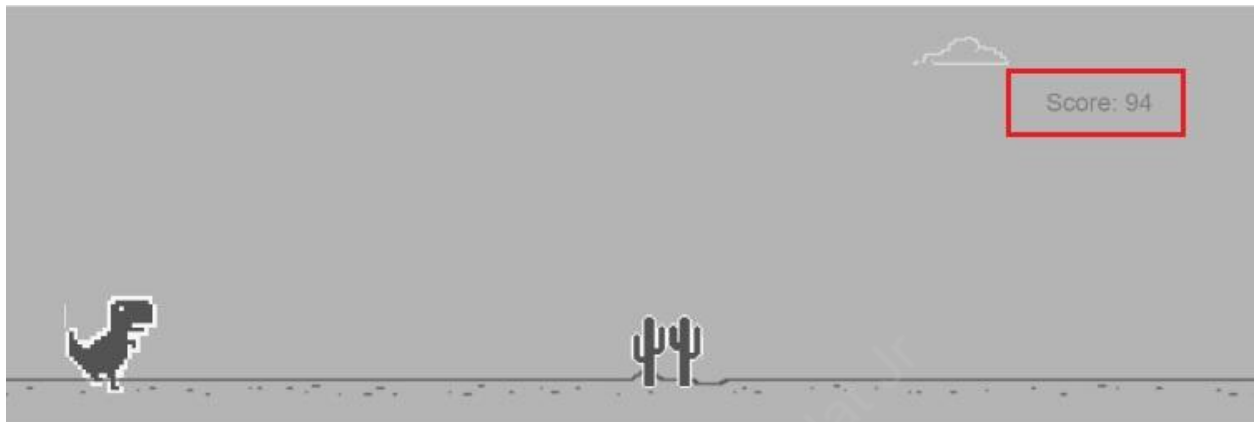
```
trex.collide(invisibleGround);  
  
//spawn the clouds  
spawnClouds();  
  
//spawn obstacles on the ground  
spawnObstacles();  
  
drawSprites();  
}  
  
function spawnObstacles(){  
}
```


8. Generate Obstacles for Trex Game:

- Created an obstacle sprite every 60 frames or so.
- Velocity of the obstacle should be the same velocity as the ground. The obstacles need to move with the ground.
- Generate and store a random number between **1** to **6**.
- Use switch statement to assign different obstacle animations for the obstacle sprites using this random number.
- Scale the obstacle images.
- Assign lifetime to obstacle to avoid Memory Leak.

```
if (frameCount % 60 === 0){  
  var obstacle = createSprite(400,365,10,40);  
  obstacle.velocityX = -6;  
  
  // //generate random obstacles  
  var rand = Math.round(random(1,6));  
  switch(rand) {  
    case 1: obstacle.addImage(obstacle1);  
            break;  
    case 2: obstacle.addImage(obstacle2);  
            break;  
    case 3: obstacle.addImage(obstacle3);  
            break;  
    case 4: obstacle.addImage(obstacle4);  
            break;  
    case 5: obstacle.addImage(obstacle5);  
            break;  
    case 6: obstacle.addImage(obstacle6);  
            break;  
    default: break;  
  }  
  
  //assign scale and lifetime to the obstacle  
  obstacle.scale = 0.5;  
  obstacle.lifetime = 300;  
}
```


OUTPUT



What's next?

In the next class we will write code to build collisions with the obstacles and use game states which we learned in the last class.

Extend Your Knowledge:

You can read more about Memory Management by Mozilla Contributors is licensed under CC-BY-SA 2.5

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Memory_Management