GAME STATES

What is our GOAL for this MODULE?

Define the behavior of different objects of the game based on gameState “**PLAY**” & “**END**”.

What did we ACHIEVE in the class TODAY?

- Used JavaScript objects to save different types of data in key-value format
- Created two new game states - PLAY and END
- Assigned different game behavior for the different states
- Designed a simple scoring system using string concatenation

Which CONCEPTS/ CODING BLOCKS did we cover today?

- Scoring system
- Game states and assigning different behaviors

How did we DO the activities?

1. Print a string on the console. When any text information is stored in a computer, it is written inside quotes "_" and called a **string**.

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

  console.log("Hello");

}

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);
```

2. Two strings can be joined together using the **+** sign, this concept is called **string concatenation**. For example: **"Hello"** + **"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);
```

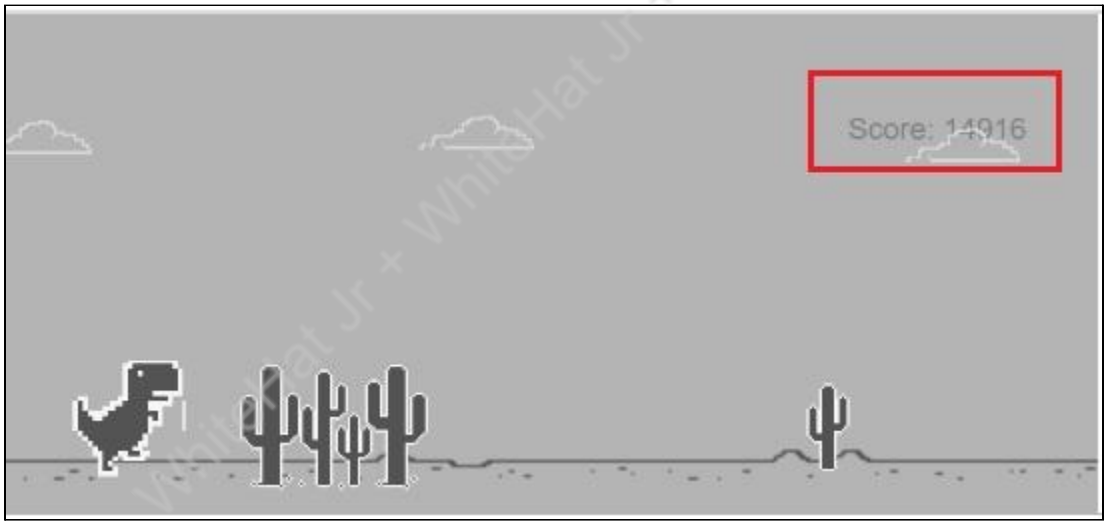
3. Build a simple scoring system. Use the **frameCount** variable as the score.

```
  console.log("Hello" + 5);

  score = 0;
}

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

Output:



4. Take **frameCount** as the score since we want to increment the score after 1 second, and our frame is 60 (60 frames per second), so if the **frameCount** is divided by **60** we will get 1 as shown in the following screenshot:
   Note: Use **Math.round()** function since we do not want any decimal, we can round off the values

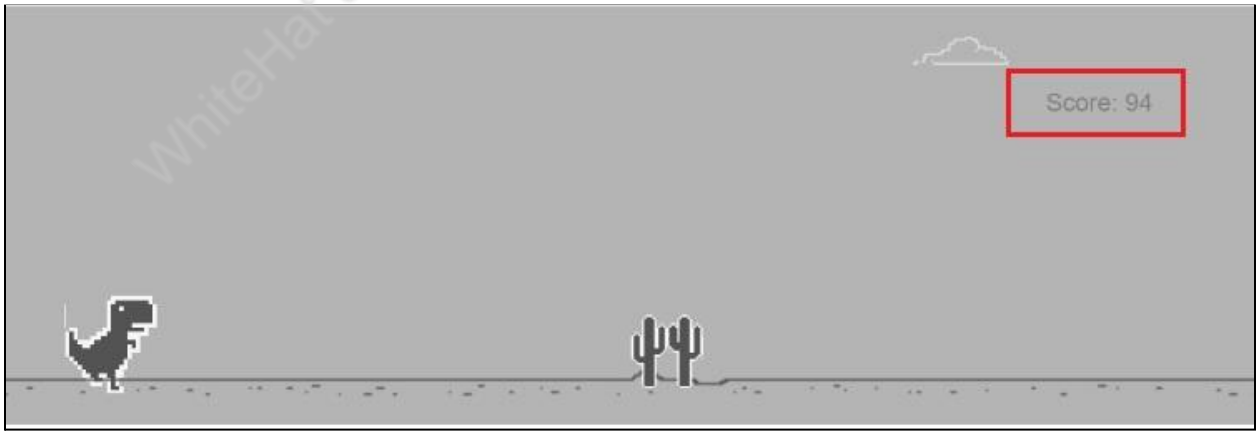
```
  console.log("Hello" + 5);

  score = 0;
}

function draw() {
  background(180);
  text("Score: "+ score, 500,50);
  score = score + Math.round(frameCount/60);

  if(keyDown("space")&& trex.y >= 362) {
    trex.velocityY = -13;
  }
```

Output:



5. Introduce a variable that will hold the value of the **gameState**. The **gameState** could be **PLAY** or **END**:

```
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;
```

6. Add an **if** and **else if** conditional statement inside the **draw()** function:

```
function draw() {
  background(180);
  //displaying score
  text("Score: "+ score, 500,50);
  score = score + Math.round(frameCount/60);

  if(gameState === PLAY){


  }
  else if (gameState === END) {

  }
```

7. Add **background** color to the game:

```
function draw() {
  background(180);
  //displaying score
  text("Score: "+ score, 500,50);
  score = score + Math.round(frameCount/60);

  if(gameState === PLAY){


  }
  else if (gameState === END) {

  }
```

8. Move the ground, in **PLAY** state, stop the movement in **END** state:

```
//displaying score
text("Score: "+ score, 500,50);
score = score + Math.round(frameCount/60);

if(gameState === PLAY){
  //move the ground
  ground.velocityX = -4;

}
else if (gameState === END) {
  //move the ground
  ground.velocityX = 0;
}
```

9. Display **score** at all times:

```
function draw() {
  background(180);

  //displaying score
  text("Score: "+ score, 500,50);

  if(gameState === PLAY){
    //move the ground
    ground.velocityX = -4;
    //scoring
    score = score + Math.round(frameCount/60);

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

10. **Reset** the ground during **PLAY** state:

```
  if(gameState === PLAY){
    //move the ground
    ground.velocityX = -4;
    //scoring
    score = score + Math.round(frameCount/60);

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

    //jump when the space key is pressed
    if(keyDown("space")&& trex.y >= 100) {
        trex.velocityY = -13;
    }
```

11. Make **Trex** jump only during the **PLAY** state:

```
  //move the ground
  ground.velocityX = -4;
  //scoring
  score = score + Math.round(frameCount/60);

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

  //jump when the space key is pressed
  if(keyDown("space")&& trex.y >= 100) {
      trex.velocityY = -13;
  }

  //add gravity
  trex.velocityY = trex.velocityY + 0.8

}

  else if (gameState === END) {
    ground.velocityX = 0;
```

12. Make the **invisible ground** support the **Trex** at all times.

```
  else if (gameState === END) {
    ground.velocityX = 0;

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


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


  drawSprites();
}
```

13. **Spawn** the cloud and the obstacles In **PLAY** state:

```
//jump when the space key is pressed
if(keyDown("space")&& trex.y >= 362) {
    trex.velocityY = -12;
}

//add gravity
trex.velocityY = trex.velocityY + 0.8

//spawn the clouds
spawnClouds();

//spawn obstacles on the ground
spawnObstacles();

if(obstaclesGroup.isTouching(trex)){
    gameState = END;
}
}
else if (gameState === END) {
  ground.velocityX = 0;

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


What's next:
We will add the code for the **END** state and create a **restart** button.

Extend Your Knowledge:
Learn & Experiment with Groups:
https://studio.code.org/docs/gamelab/createGroup/#:~:text=Creates%20a%20new%20group%20and,all%20the%20%22enemy%22%20sprites