Topic	Trex and the Infinite Game World	
Class Description	**Students design a dinosaur for the Trex game with jump and run animations. Students learn to create an infinite game world in the limited screen space available to the players.**	
Class	**PRO-C9**	
Class time	**45 mins**	
Goal	• Build a dinosaur with jump and run animations. • Learn to scale the images in the game. • Learn to log messages / outputs from the program into the console for testing purposes. • Create an infinitely scrolling ground for the dinosaur to run on.	

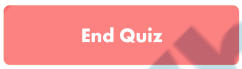
Resources Required	• Teacher Resources
	  ○ p5 login
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	• Student Resources
	  ○ p5 login
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen	

Class structure	**Warm Up** **Teacher-led Activity** **Student-led Activity** **Wrap up**	**5 mins** **10 min** **20 mins** **5 min**

CONTEXT

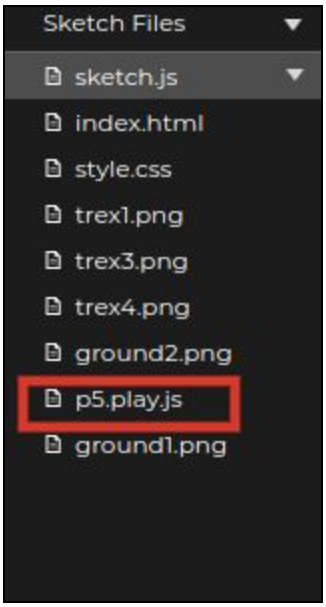
- **The problem of limited screen space available but the need for an infinite game world.**

Class Steps	Teacher Action	Student Action
Step 1: Warm Up (5 mins)	"Hi! How have you been? How has been your journey into programming and game design so far?"	**ESR:** varied response

	I have an exciting quiz question for you! Are you ready to answer this question? Teacher click on the  button on the bottom right corner of your screen to start the In-Class Quiz. A quiz will be visible to both you and the student. Encourage the student to answer the quiz question. The student may choose the wrong option, help the student to think correctly about the question and then answer again. After the student selects the correct option, the  button will start appearing on your screen. Click the End quiz to close the quiz pop-up and continue the class.	**ESR:** Yes
	Have you ever played the dinosaur game in your chrome browser when your computer is unable to connect to the internet? The game is called TRex Runner - because it is a Tyrannosaurus (often called as TRex) dinosaur running through the desert. We will be making this game but maybe we will make it in our own style - maybe with colors and better graphics.	**ESR:** Yes/No

	Let's play the game again and observe the game for its features and functions. *Teacher enters the screen share and plays the game here - Teacher Activity 1* *Teacher shares the link of the game through chat. Why don't you try playing the game as well.*	*Student opens Student Activity 1 and the student plays the TRex runner game as well.*
	Let's talk about the features in the game. What did we see?	**ESR:** - There is a jumping and running dinosaur. - There are clouds which are moving. - There is a moving ground. - There are cactus which are randomly appearing on the screen. Cacti are of different sizes and widths. - There are some crows which appear on the screen at different heights. - Player needs to press "space" to make the dinosaur jump over the obstacles. - There is a score which keeps a record of the points scored by the player. - Game is over when the dinosaur hits one of the obstacles in the game.
	Awesome, we will be starting by building the jumping and running dinosaur. Excited?	**ESR:** Yes!

	"We also have an important problem to solve in today's class. We have a limited screen space when we are designing the game but remember the Trex runner game we saw? The dinosaur keeps running infinitely.... How do we do that? Any ideas?"	**ESR:** varied response
	"This is a common problem for most advanced games. We want an infinite game world in a limited screen space. You will be solving this today. Also, we will be doing all our code in p5 and not code.org! Let's gear up for today's class."	-
Teacher Initiates Screen Share		
CHALLENGE • **Use of logging into the console for testing purpose** • **Trick the player to perceive an infinitely scrolling ground.**		
Step 2: Teacher-led Activity (10 min)	Great! Let's open and look at the activity for the class(**Teacher Activity 2)** Check under the files and folders tab, I have uploaded different pictures of the Trex dinosaur. We will use them to create the animation. You will also observe that there is a new file called p5.play.js. This is a game library which code.org was internally using to allow you to create games! We have added the library in p5 now and we can create games similar to how we were creating in code.org.	-



index.html • Saved: 1 minute ago

```
<!DOCTYPE html>
<html>
  <head>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.8.0/p5.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.8.0/addons/p5.dom.min.js">
</script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.8.0/addons/p5.sound.min.js">
</script>
<script src="./p5.play.js"></script>
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta charset="utf-8" />

  </head>
  <body>
      <script src="sketch.js"></script>
  </body>
</html>
```

Creating animation is similar to adding images.

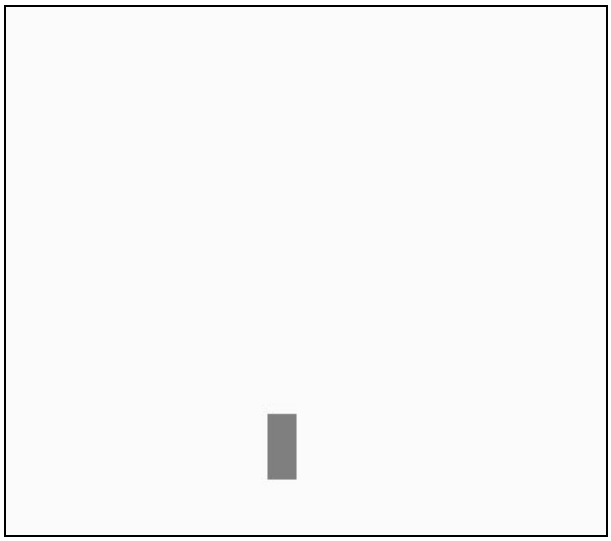
p5 has something called loadAnimation and addAnimation to add animations to the sprite.

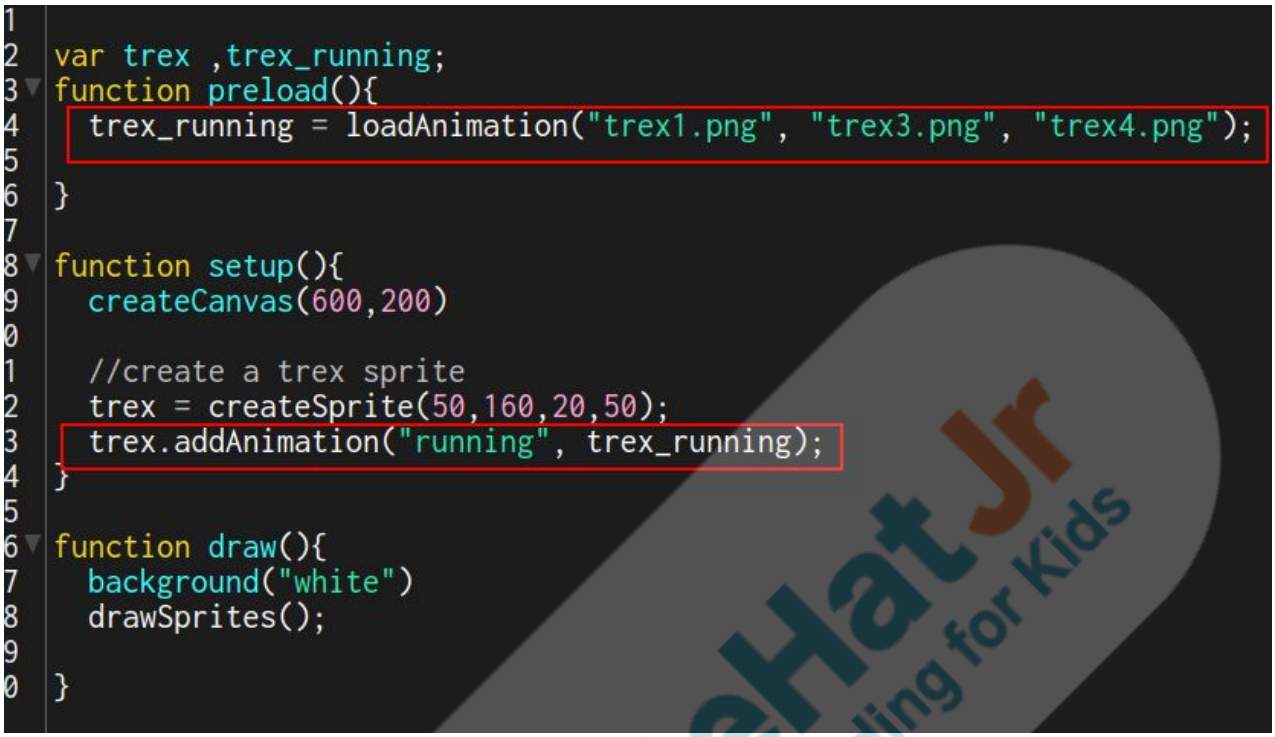
We use **loadAnimation** to load all the images for the animation and then use **addAnimation** to add the animation to the sprite.

	Let's code and create a sprite somewhere near the ground. Let's give a name to our sprite and store it in a variable. *Ask the student to give inputs while you are writing the code.* *Teacher runs the code to check the output.*	**ESR:** *Student asks the teacher to use createSprite() to create a sprite and give it specific positions, width and height.* *The student uses drawSprites() to draw the sprite on the screen.*

```
var trex

function setup(){
  createCanvas(600,200)

  //create a trex sprite
  trex = createSprite(50,160,20,50);

}

function draw(){
  drawSprites();

}
```

Output:

		
	Let's load our animation for our Trex sprite in function preload and then set animation. **Note:** using **loadAnimation()** load the animations in the preload function. you can load multiple images in a single variable to make it animated. **addAnimation** takes two arguments - a label which could be any string and the name of the animation which was loaded.	*Student observes how to load and set animation for the Trex.*



```
var trex ,trex_running;
function preload(){
  trex_running = loadAnimation("trex1.png", "trex3.png", "trex4.png");

}

function setup(){
  createCanvas(600,200)

  //create a trex sprite
  trex = createSprite(50,160,20,50);
  trex.addAnimation("running", trex_running);
}

function draw(){
  background("white")
  drawSprites();

}
```

	We have our running TRex. Now, let's make it jump when we press "space". The dinosaur should jump when we press space and then drop back to the ground. Any ideas on how to do it? Let's recollect some physics and gravity here. What happens to a ball when it bounces off the ground? How does its velocity change? *Teacher demonstrates the physics behind a falling object with a physical example.*	**ESR:** varied **ESR:** When the ball jumps off the ground, it has some velocity but due to gravity the velocity keeps reducing. It becomes 0 at the top but due to gravity it starts gaining velocity again when it starts falling towards the ground.
	Exactly! Let's give the ball some velocity when space is pressed. Do you know how to do that?	**ESR:** Using if condition **ESR:**

	Which direction should we give the velocity? In our game is upwards velocity positive(+) or negative(-)? Let's give a velocity of -10 to the TRex in vertical direction when space is pressed. Remember to add background() to the game. *Teacher writes code to give the dinosaur a velocity of -10 when the "space" key is pressed.* *Teacher runs the code to see the output.*	Upwards? Negative

```
1  var trex, trex_running, edges;
2
3  function preload(){
4    trex_running = loadAnimation("trex1.png","trex3.png","trex4.png");
5  }
6
7  function setup(){
8    createCanvas(600,200);
9    trex = createSprite(50,160,20,50);
10   trex.addAnimation("running", trex_running);
11
12 }
13
14
15 function draw(){
16   background("white");
17
18   if(keyDown("space")){
19     trex.velocityY = -10;
20   }
21
22   drawSprites();
23 }
```

	What happened?	The T-Rex jumped off the screen.

	This is because there was no gravity! Let's give some gravity. What does gravity do? What will be the effect of gravity on the velocity of the TRex which is going up? Let's add a line in our code which will do that. **trex.velocityY = trex.velocityY + 0.5;** Since velocity is -ve, the +0.5 will reduce the velocity everytime in the upward direction and bring it to 0. Then, it will make the trex move in the other direction. *Teacher adds the line of code which will give the effect of gravity for the dinosaur.* *Teacher runs the code to see the output.*	**ESR:** It pulls the object towards itself. Gravity will slow down the TRex and reduce its velocity to 0. It will then pull the trex down.

```
10    trex.addAnimation("running", trex_running);
11    edges = createEdgeSprites();
12  }
13
14
15  function draw(){
16    background("white");
17
18    if(keyDown("space")){
19      trex.velocityY = -10;
20    }
21
22    trex.velocityY = trex.velocityY + 0.5;
23    drawSprites();
24  }
```

	Do you see the gravity? What is the problem now?	**ESR:** There is gravity but the Trex falls off the ground.

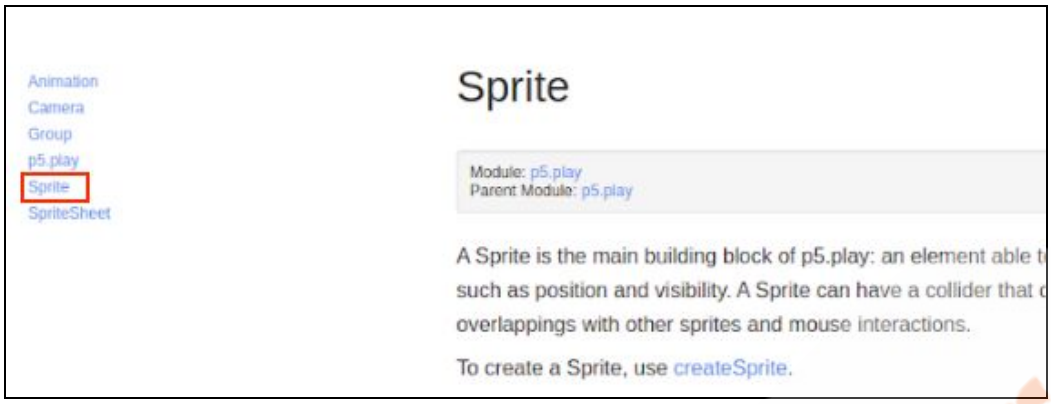
	Any ideas on how to solve this. Let's create the edges and make the trex collide with the bottom edge so that it does not fall off the ground.	**ESR:** We can write code to create the edges and make the trex collide with the bottom edge.

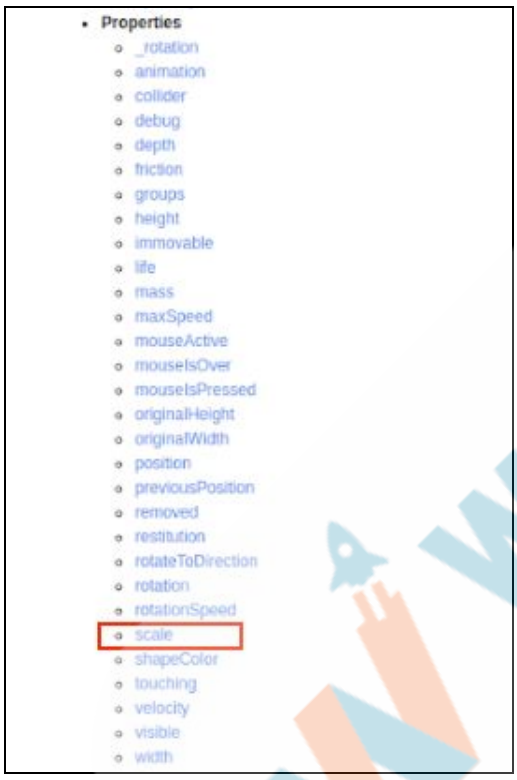
```
function setup(){
  createCanvas(600,200);
  trex = createSprite(50,160,20,50);
  trex.addAnimation("running", trex_running);
  edges = createEdgeSprites();
}


function draw(){
  background("white");

  if(keyDown("space")){
    trex.velocityY = -10;
  }

  trex.velocityY = trex.velocityY + 0.5;
  trex.collide(edges[3])
  drawSprites();
}
```

	Looks like we have the jumping and running dinosaur now! And that was quick!	
	“Our code doesn't have any comments. Can you quickly help me in writing comments for the code?"	The student helps the teacher in writing comments for different blocks of lines in the code.

```
function preload(){
  trex_running = loadAnimation("trex1.png","trex3.png","trex4.png");
}

function setup(){
  createCanvas(600,200);

  // creating trex
  trex = createSprite(50,160,20,50);
  trex.addAnimation("running", trex_running);
  edges = createEdgeSprites();
}


function draw(){
  //set background color
  background("white");

  //jump when space key is pressed
  if(keyDown("space")){
    trex.velocityY = -10;
  }

  trex.velocityY = trex.velocityY + 0.5;

  //stop trex from falling down
  trex.collide(edges[3])
  drawSprites();
}
```

	"Good work. Now anyone can read our code and understand easily. Don't you think our dinosaur is too huge compared to the original game?"	ESR: Yes
	"There should be something in our Sprite module to help us scale the dinosaur. Help me look for it. Let us look for the sprite object in p5.play documentation.	*The student helps the teacher in looking for the sprite.scale in the sprite documentation.*



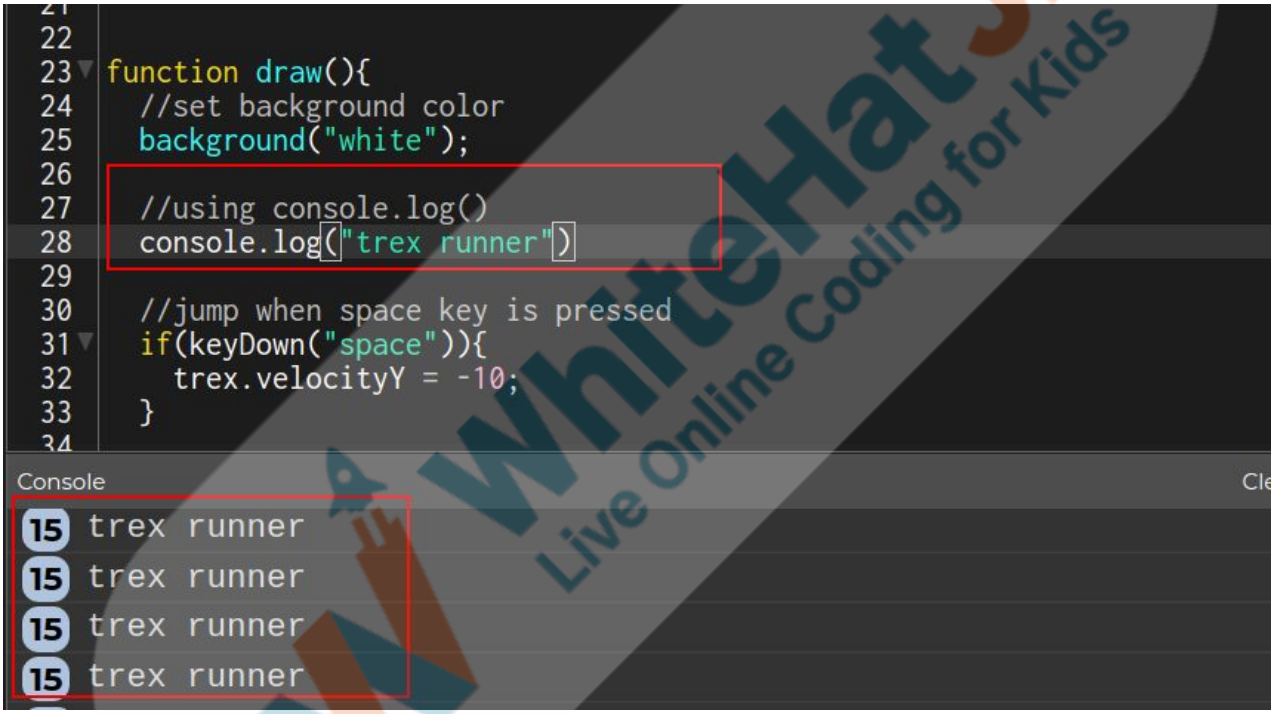


	"Let us use the sprite.scale instruction to scale the dinosaur to half its size. *Teacher writes and runs the code.* Does the dinosaur look the same size as in the original game?"	**ESR:** Yes
	"How can we position the dinosaur to the extreme left?"	**ESR:** By setting the x position of the Trex sprite.

	"Let us set the x position of the Trex sprite so that it is to the left. What could be the instruction?" *Teacher writes and runs the code.*	**ESR:** trex.x = 50;

```
function preload(){
  trex_running = loadAnimation("trex1.png","trex3.png","trex4.png");
  groundImage = loadImage("ground2.png")
}

function setup(){
  createCanvas(600,200);

  // creating trex
  trex = createSprite(50,160,20,50);
  trex.addAnimation("running", trex_running);
  edges = createEdgeSprites();

  //adding scale and position to trex
  trex.scale = 0.5;
  trex.x = 50
}

function draw(){
  //set background color
  background("white");

  //jump when space key is pressed
  if(keyDown("space")){
    trex.velocityY = -10;
  }
```

	"Awesome. Do you see our dinosaur doesn't jump correctly right now? Can you identify the problem?"	**ESR:** The dinosaur keeps moving up and doesn't fall if we keep the space key pressed.
	"Right. This is a problem that we will solve later. For now, I will tell you an important tool using which you can find out what's happening inside the program while the program is running. This will	

	help us find out and correct errors in our program."	
	"Our editor has a console window where we can log any message while the program is running. We do this using console.log() instruction. Let us log the name of the game in the console window" Teacher writes and runs the code.	*Student observes how to use console.log() instruction.*
		
	Do you know just like movies, games are made up of many frames. Anything we write inside the function draw() runs for every frame. Anything we write outside the function draw() runs only once. What would happen if we run console.log() inside the draw() function?"	**ESR:** We would see the message from console.log() multiple times in the console

	"Let us write the console.log() instruction inside the draw() function. Also instead of logging the name of the game, let us log the y position of the trex sprite." *Teacher writes the code and runs.*	window for every frame. *Student observes.*
		
	Teacher runs the code and presses the space key to make the dinosaur jump. "What do you see in the console window? Why do the numbers change?"	**ESR:** There are numbers as output in the console window. These numbers correspond to the y position of the trex. When the trex jumps, the y position of the trex changes and the

		numbers in the console change.
	"Isn't it amazing to see how the y position of the trex is changing? console.log() is a powerful tool to help us understand what our program is doing at each stage. We will use it repeatedly in our projects. For now, I want you to get started on moving the dinosaur along an infinite ground space. I will guide you to do that."	**ESR:** Yes.
Teacher Stops Screen Share		
	Now it's your turn. Please share your screen with me.	
• **Ask Student to press ESC key to come back to panel** • **Guide Student to start Screen Share** • **Teacher gets into Fullscreen**		
ACTIVITY • **Create a ground sprite and make it move backwards.** • **Reset the ground position when the dinosaur reaches the end of the ground.** • **Use a symmetrical ground image for the ground animation.**		
Step 3: Student-Led Activity (20 mins)	Let us create a rectangular sprite called ground. This is where the Trex dinosaur will run. The ground sprite should ideally cover the entire screen. What will be the height and width of such a sprite? What will be its x and y position?	*Student opens the **Student Activity 2** and duplicates the code.* **ESR:** Height : 20 Width: 400 x: 200 (the sprite should be centred on the screen) y: 180 *Student writes the code*

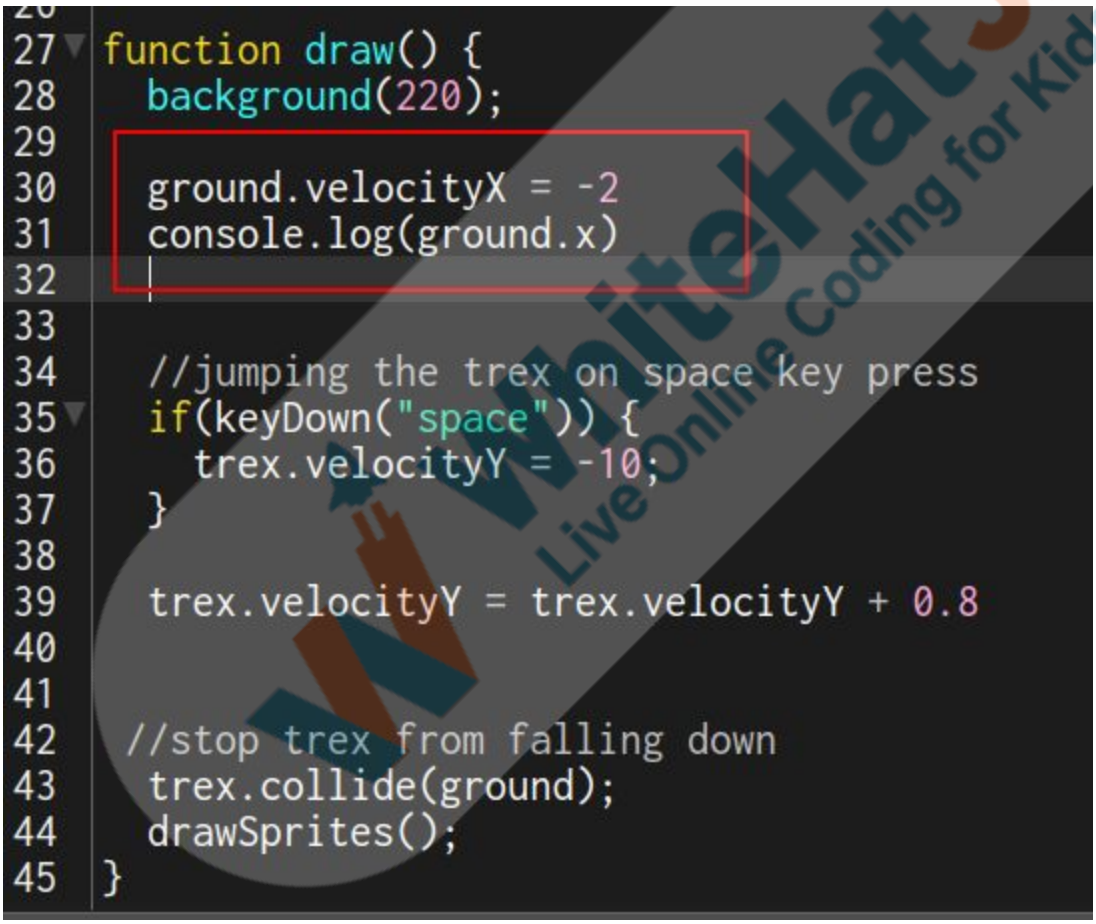
	Guide the student to create a ground sprite.	*to create a ground sprite with the guidance of the teacher.*

```
9
10 function setup() {
11   createCanvas(600, 200);
12
13   //create a trex sprite
14   trex = createSprite(50,160,20,50);
15   trex.addAnimation("running", trex_running);
16
17   //adding scale and position to trex
18   trex.scale = 0.5;
19   trex.x = 50
20
21   //create ground sprite
22   ground = createSprite(200,180,400,20);
23
24 }
```

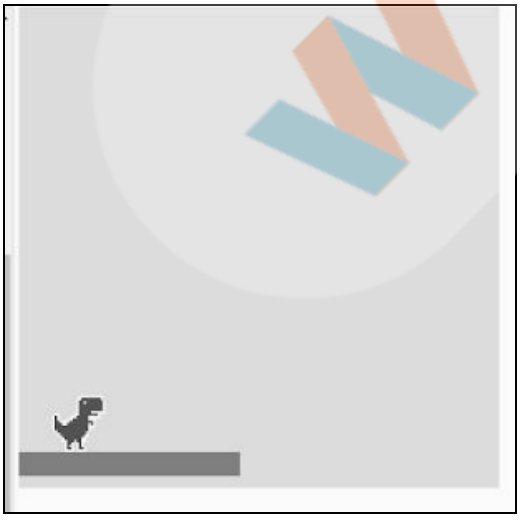
	Right now the ground is on the dinosaur, don't you think the dinosaur should collide with the ground instead of the edges? Let's fix this. Why don't we remove the edges and make the Trex collide with the ground sprite.	**ESR:** Yes *Student writes the code.*

```
20
21    //create ground sprite
22    ground = createSprite(200,180,400,20);
23
24  }
25
26  function draw() {
27    background(220);
28
29
30    //jumping the trex on space key press
31    if(keyDown("space")) {
32      trex.velocityY = -10;
33    }
34
35    trex.velocityY = trex.velocityY + 0.8
36
37
38   //stop trex from falling down
39    trex.collide(ground);
40    drawSprites();
41  }
```

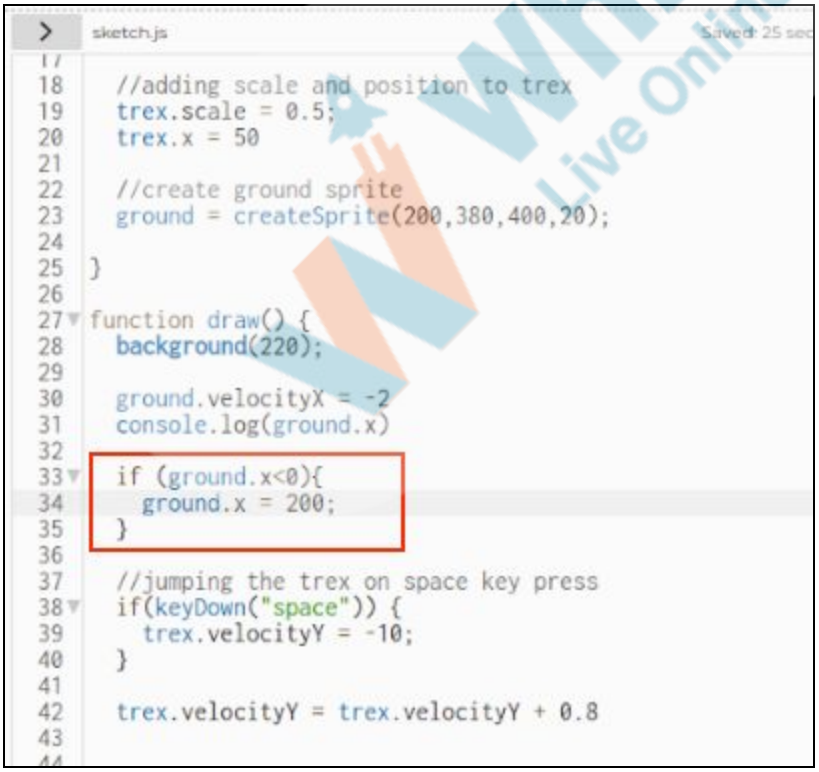
	Ok, now let's move the dinosaur. There are two ways to make the player feel that the dinosaur is moving. One would be to give some forward velocity to the trex and the other would be to give some backward velocity to the ground. Let us try each of the ways. Let us first give some forward velocity to the trex and see what happens. Let's keep logging both the trex.x and trex.y in the console window.	*Student writes code to give some x velocity to the trex.* *The student first logs the trex.x and then trex.y in the console window.* *Student runs the code and observes both the output and the console window.*



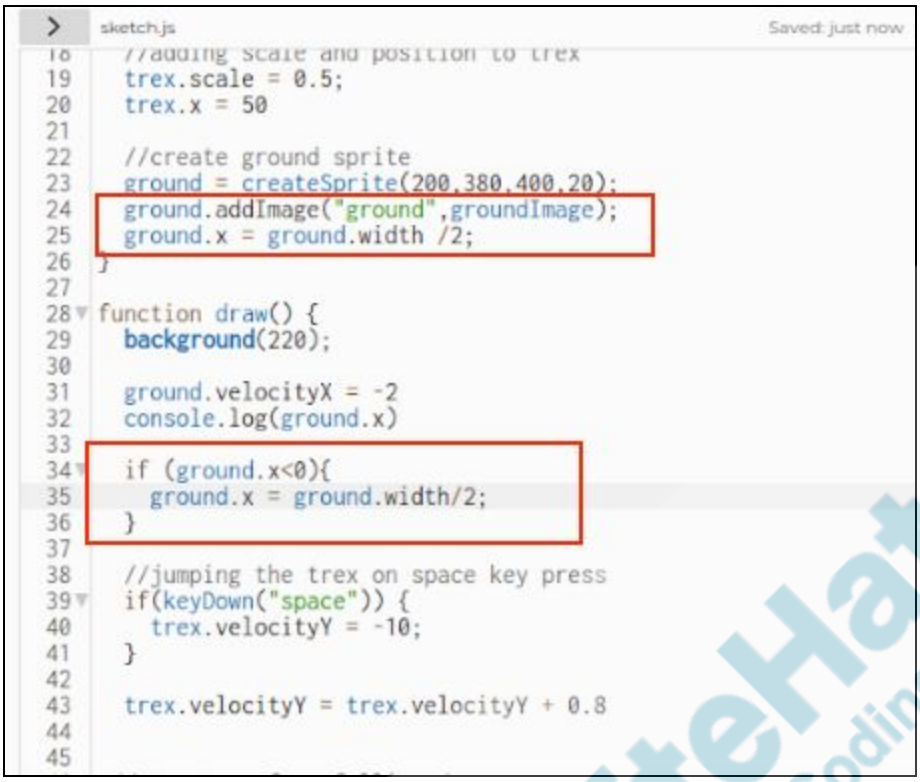
```
function draw() {
  background(220);

  ground.velocityX = -2
  console.log(ground.x)

  //jumping the trex on space key press
  if(keyDown("space")) {
    trex.velocityY = -10;
  }

  trex.velocityY = trex.velocityY + 0.8


 //stop trex from falling down
  trex.collide(ground);
  drawSprites();
}
```

	"What is happening to the trex?" Guide the student to look at the log messages and infer	**ESR:** The Trex is going outside the screen and then falling. trex.x is increasing constantly. trex.y is increasing exponentially.
		
	"Let's try the other way now. Let us try to give some backward velocity to the ground and see what happens. Also let us log the ground.x now"	*The student writes code to give some x velocity to the ground in the negative directions.* *Student logs the ground.x on the console window.* *Student runs the code and observes both the output and the console window.*
		

	What do you observe?	**ESR:** Trex reaches the edge of the ground and then falls off.
	"Can you think of a way where the trex never falls and the ground keeps scrolling infinitely?"	**ESR:** varied
	"One simple way would be to reset the ground back to the center if ground.x < 0. Once the ground has crossed the screen to the left, we are bringing it back to its original position. This way the ground will always be there. Let's do that and see what happens. How will we do that?" Teacher guides the student to write code to reset the ground to the centre if ground.x < 0	**ESR:** Using conditional programming *Student writes and runs the code to reset the ground if ground.x < 0*



	"What do you see now?" "What is the problem right now?"	**ESR:** The ground keeps resetting itself. "We can see the ground moving and resetting itself."
	"How can we solve this problem?"	**ESR:** varied
	Let us join two identical ground images. When half of the ground image goes off the screen, we will reset the ground. This will build an illusion that there is always ground to cover. The player will never feel that the ground has reset itself and will see an infinitely scrolling ground. Let us use the image "ground1" already uploaded in the files directory. It has two ground images - one is red and the other is green - so that you can see what is happening. Later we will use a ground image as in the game. Where should the ground.x be for this image? We want the ground to be symmetrically placed on the screen. So let us keep ground.x = ground.width / 2; When ground.x < 0, we will reset the ground back to its center.	**ESR:** At half the width of the ground. *Student sets the ground's sprite animation to ground1 and makes changes in the code. Student runs the code and sees the output.*



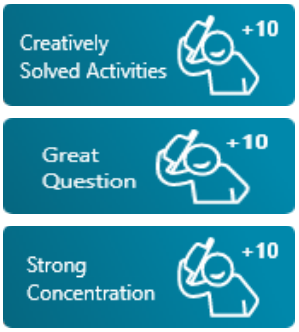
	You can see the ground resetting itself. Let us use the actual ground image. There is a ground image uploaded on the animations tab. It contains two ground images joined to each other.	*Student sets the new animation for the ground and runs the code.*

<table>
<tr><th colspan="3">Teacher Guides Student to Stop Screen Share</th></tr>
<tr><td colspan="3"><u>FEEDBACK</u>

Appreciate the student for their efforts in the class.
Review the content of the class.
Ask the student to make notes for the reflection journal along with the code they wrote in today's class.
</td></tr>
<tr><td>Step 4:
Wrap-Up
(15 min)</td><td>Can you quickly capture what we learned in today's class?</td><td>ESR:
- We learned the use of console.log() to log what is happening in the program in real time.
- We also learned how to trick the player into believing that the game world is infinite by resetting the ground every time.</td></tr>
<tr><td></td><td>You get Hats Off for your excellent work!

Awesome! and remember we have a couple of bugs (problems) in our game. In computer programs, if we don’t get the output we expect, we term it as a bug!

You might try to fix these bugs on your own. Let us meet in the next class again. Keep your game energy alive.</td><td>Make sure you have given at least 2 Hats Off during the class for:

 
</td></tr>
</table>

	Also, your next class project is going to be about creating your own version of an endless runner game! We will need you to call one of your friends in the class - it is going to be a fun and exciting class where you will brainstorm with your friend about the game and generate new ideas. You can call any of your classmates or anyone who could help you in brainstorming about your game!	
	Also, keep writing your notes in the reflection journal.	-
Project Overview	**BALLOON BUSTER - 1** **Goal of the Project:** In Class 9 you have learned how to scale the images in the game and how to create an infinitely scrolling ground. In this project, you will have to practice and apply what you have learned in the class and create a Balloon Buster game. **Story:** Meera visited her grandparents. She loved the game of bursting balloons with a bow and arrow. When she came back home, she could not play the game anymore. So she decided to build a computer game similar to the actual balloon bursting game. Can you help Meera design the game? I am very excited to see your project solution and I know you will do really well.	*Students engage with the teacher over the project.*

	Bye Bye!	
Teacher Clicks 		
Additional Activities	*Challenge the student to create the running ghost animation using* **Student Additional Activity Link** *Images are uploaded* *Teacher can show the output to the student on what the ghost animation should look like.*	*Student creates the running ghost animation similar to* ***Teacher Additional Activity Reference*** *provided*
	Encourage the student to write reflection notes in their reflection journal using markdown. Use these as guiding questions: • What happened today? • Describe what happened • Code I wrote • How did I feel after the class? • What have I learned about programming and developing games? • What aspects of the class helped me? What did I find difficult?	*Student uses the markdown editor to write her/his reflection as a reflection journal.*

Activity	Activity Name	Links
Teacher Activity 1	Trex Game	http://www.trex-game.skipser.com/
Teacher Activity 2	Trex Stage 0	https://editor.p5js.org/whitehatjr/sketches/s42U2SBau
Teacher Activity 3	Sprite documentation	https://molleindustria.github.io/p5.play/docs/classes/Sprite.html
Teacher Activity 3	Reference code (Trex Stage 1)	https://editor.p5js.org/Abhijeet/sketches/bqCypXeR0

Student Activity 1	Trex Game Link	http://www.trex-game.skipser.com/
Student Activity 2	Trex Stage 0.5	https://editor.p5js.org/whitehatjr/sketches/MJwoAMYpY
Student Additional Activity	Running Ghost (Empty Activity)	https://editor.p5js.org/whitehatjr/sketches/CGeUr7i3h
Teacher Additional Activity	Reference	https://editor.p5js.org/whitehatjr/sketches/vBXI1xC5u