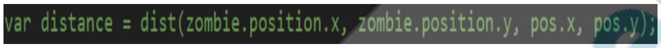
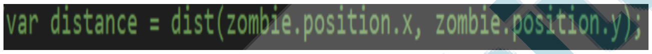
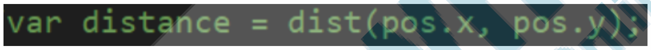
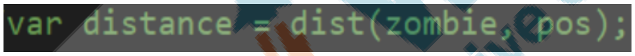
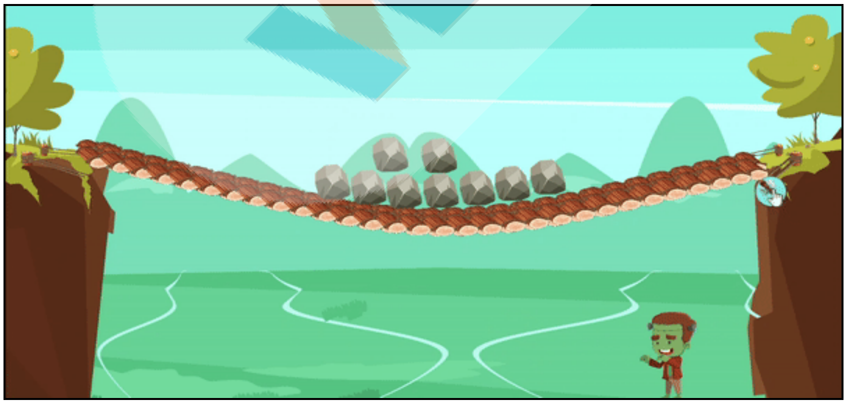
WhiteHat Jr
Live online Coding for Kids

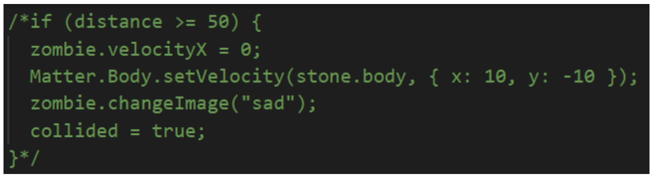
Topic	**ADDING SOUNDS AND CHALLENGES**	
Class Description	**The student learns to add sound and an air blower which will push the hanging fruit towards the direction of the bunny.**	
Class	**C32**	
Class time	**45 mins**	
Goal	● Add background music ● Add sound effects for cutting, eating, and blowing air. ● Create the air blower effect.	
Resources Required	● Teacher Resources ○ VS Code Editor ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ● Student Resources ○ VS Code Editor ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen	
Class structure	**Warm-Up - Slide show option** **Teacher-Led Activity** **Student-Led Activity** **Wrap-Up - Slide show option**	**10 Mins** **10 Mins** **20 Mins** **5 Mins**

WARM-UP SESSION - 10mins

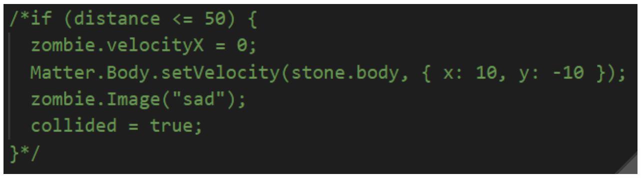
Teacher starts slideshow from slides 1 to 16
Refer to speaker notes and follow the instructions on each slide.

Teacher Action	**Student Action**

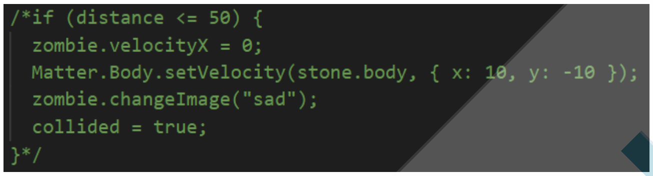
Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today? ***Run the presentation from slide 1 to slide 4.*** **The following are the warm-up session deliverables:**	**ESR**: Hi, thanks, Yes I am excited about it! Click on the slide show tab and present the slides
QnA Session	
Question	**Answer**
Select the correct option to detect collision between the **zombie** and the **stone** by checking the distance between them. A.  `var distance = dist(zombie.position.x, zombie.position.y, pos.x, pos.y);` B.  `var distance = dist(zombie.position.x, zombie.position.y);` C.  `var distance = dist(pos.x, pos.y);` D. E.  `var distance = dist(zombie, pos);`	A
Select the correct option to set the zombie's velocity to '0' and change the zombie's image to a sad image when the distance is less than 50. 	C

A. 

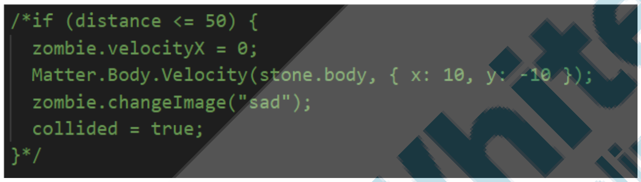
```
/*if (distance >= 50) {
  zombie.velocityX = 0;
  Matter.Body.setVelocity(stone.body, { x: 10, y: -10 });
  zombie.changeImage("sad");
  collided = true;
}*/
```

B. 

```
/*if (distance <= 50) {
  zombie.velocityX = 0;
  Matter.Body.setVelocity(stone.body, { x: 10, y: -10 });
  zombie.Image("sad");
  collided = true;
}*/
```

C. 

```
/*if (distance <= 50) {
  zombie.velocityX = 0;
  Matter.Body.setVelocity(stone.body, { x: 10, y: -10 });
  zombie.changeImage("sad");
  collided = true;
}*/
```

D. 

```
/*if (distance <= 50) {
  zombie.velocityX = 0;
  Matter.Body.Velocity(stone.body, { x: 10, y: -10 });
  zombie.changeImage("sad");
  collided = true;
}*/
```

Continue the warm-up session

Activity details	Solution/Guidelines
Run the presentation from slide 5 to slide 16 to set the problem statement. **The following are the warm-up session deliverables:** ● Explain about the background sound and mute button. ● Explain the working of applied force.	Narrate the slides by using hand gestures and voice modulation methods to bring in more interest in students.

Teacher ends slideshow

TEACHER-LED ACTIVITY - 10 mins

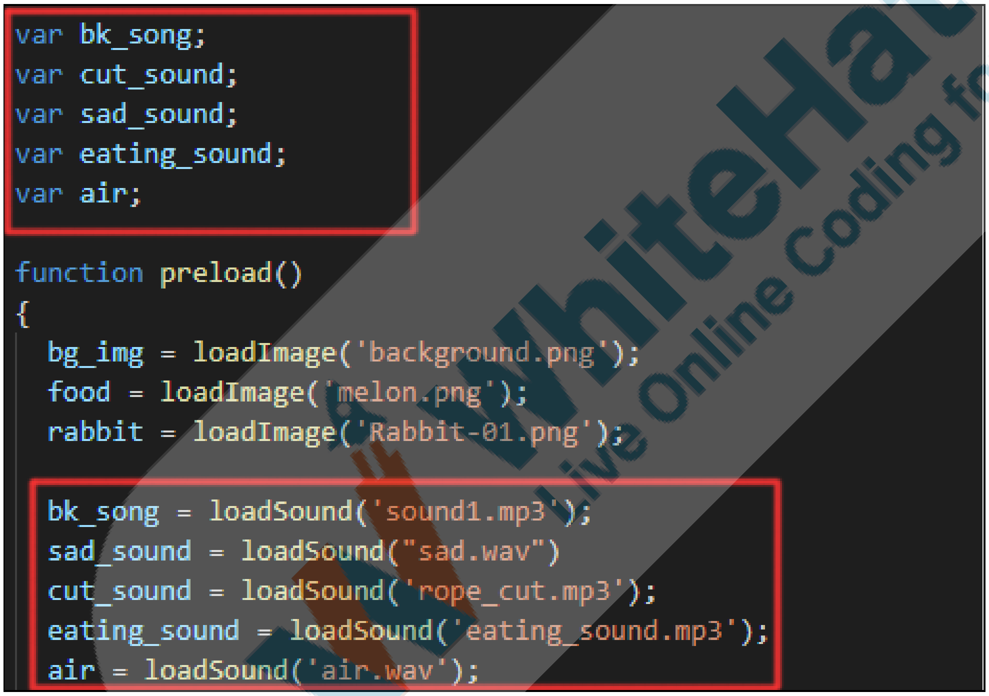
Teacher Initiates Screen Share	
CHALLENGE • **Add a background soundtrack to the game.** • **Add a sound effect when the rope is detached.**	
Teacher Action	**Student Action**
Teacher-led Activity-1: *The teacher downloads the Teacher Activity 1 code and opens it in the VS Code editor.* What do you think are the elements that make an amazing game? Games are fun when we have music and sound effects in them. For example, in the game Fortnight, if you don't have sounds such as shooting and background music you can imagine how boring it will become. In this class, we are going to learn how to add sound files to our code and trigger them based on certain events. The process is very similar to loading images in our code. Are you ready? First, we need to load all the sound files in the code. Can you tell me which function we used to do that? Great! In the **sketch.js** file create the **preload()** function we are going to load the sound files. In our case, we will need one background music that will play continuously.	**ESR:** Sounds and Music. **ESR:** Yes! **ESR:** The **preload()** function

Then we need sound effects when the user:

- Cuts the rope.
- When the bunny eats the fruit or when the bunny could not eat the fruit.
- We will also add the sound of the air blowing when the user presses the blower button.

First, we will create variables for all these sounds. Then we will load the sounds in the **preload()** function and assign them to the respective variables.

The teacher writes code as shown below.



```
var bk_song;
var cut_sound;
var sad_sound;
var eating_sound;
var air;

function preload()
{
  bg_img = loadImage('background.png');
  food = loadImage('melon.png');
  rabbit = loadImage('Rabbit-01.png');

  bk_song = loadSound('sound1.mp3');
  sad_sound = loadSound("sad.wav")
  cut_sound = loadSound('rope_cut.mp3');
  eating_sound = loadSound('eating_sound.mp3');
  air = loadSound('air.wav');
```

We have our sounds loaded now, we need to apply these sounds to their correct places now.

Now is your turn.

Please share your screen with me.

Teacher starts slideshow : Slide 17 to 27	
Run the presentation slide 17 to slide 27 to set the student activity context.	

<table>
<tr><th colspan="2">Teacher ends slideshow</th></tr>
<tr><td colspan="2"><b>Teacher Stops Screen Share</b></td></tr>
<tr><td colspan="2"><b>STUDENT-LED ACTIVITY - 20 mins</b></td></tr>
<tr><td colspan="2"><b>● Ask Student to press ESC key to come back to panel
● Guide Student to start Screen Share
● Teacher gets into Fullscreen</b></td></tr>
<tr><td colspan="2"><b><u>ACTIVITY</u></b>
<b>● Create a button and function for the air blow button.</b>
<b>● Add sounds.</b>
<b>● Add the mute button.</b></td></tr>
<tr><td><b>Teacher Action</b></td><td><b>Student Action</b></td></tr>
<tr><td><b>Student Activity 1</b>
We now have our sounds.
But before we start adding sounds to the functions, we need to do one more important thing.
We want the game to be challenging for the user.
So we need to place the bunny a little away from the fruit so that if the user cuts the rope, the fruit will not fall directly on the bunny.
So to push the fruit in the direction of the bunny we will</td><td><i>The student downloads the</i> Student Activity 1 <i>code and runs it in the VS Code editor.</i></td></tr>
</table>

create an air balloon button, when the user clicks on this button it will push the fruit towards the bunny.

For this task we need two things:

1. Function to push the fruit.
2. Button to execute the function.

Do you have any idea how to do it?

Let’s create the first function.

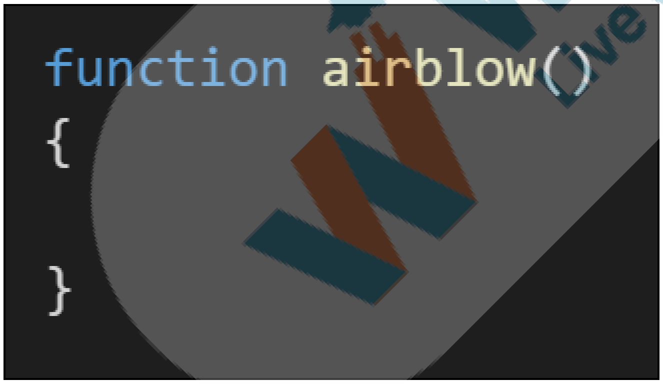
In the **matter.js** library, we have a function to apply force to a body. In our case, we will use this function to apply the force on the fruit body.

But we need an event to apply force and that event is the button press.

We will create a function named **airblow()** in the **sketch.js** file.

The student writes the code.

ESR:
Varied



```
function airblow()
{


}
```

In this function, we will add code to apply force on the fruit body.

To apply force we need the point of the body where we want to apply the force followed by the direction and amount of the force.

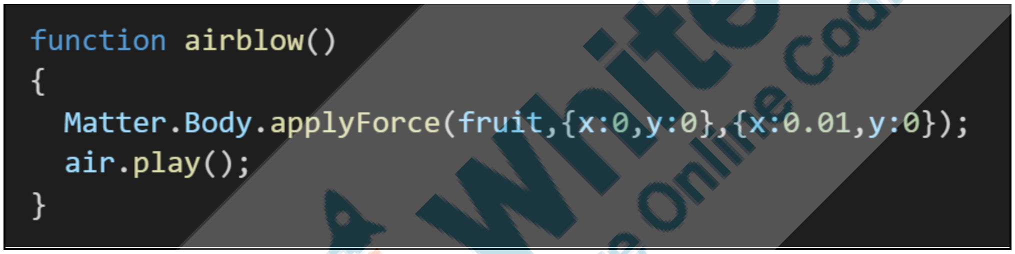
Points and direction to apply.
applyForce(fruit,{x:0,y:0},{x:0.01,y:0});

We will apply force on the center point of the fruit and the direction set as towards the right side. That is why the x value is +ve **0.01**(*this also sets the amount of force*), to reverse the direction of force change the sign to negative, if you want to apply force in the vertical direction then you need to do the same with the y value.

Along with this, we also want the air sound to be played when this function is called. To play this sound, we will simply call the **play()** function on that sound.

Air is the sound variable, so we can write **air.play();**

Now add this to our function.



```
function airblow()
{
  Matter.Body.applyForce(fruit,{x:0,y:0},{x:0.01,y:0});
  air.play();
}
```

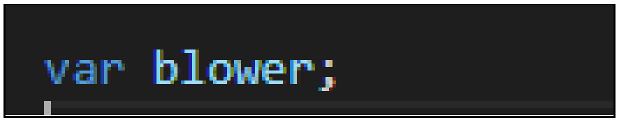
We have our function, but this function won't do anything until we call this function.

To call the function we will attach this with a button and the button will be the image of a balloon.

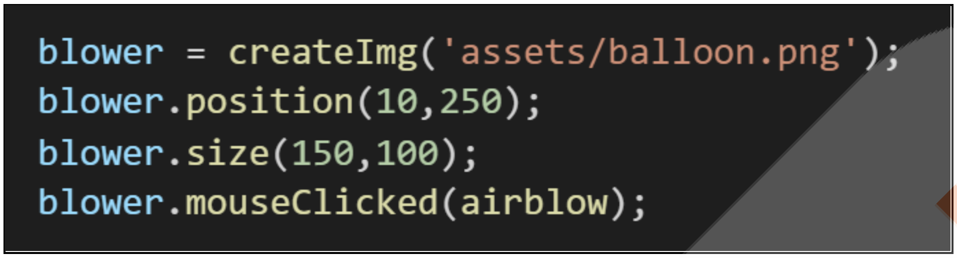
We will declare a variable for that, let's call it **var blower.**

In the **setup()** function, we will add code to create the image button, and we will also add the **airblow()** function with this button.

To create the button, we need to specify the position and the size of the button.



```
var blower;
```

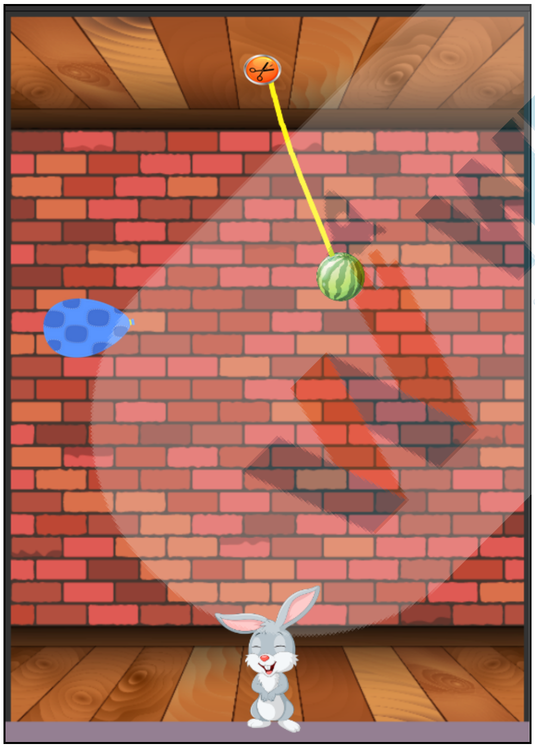


```
blower = createImg('assets/balloon.png');
blower.position(10,250);
blower.size(150,100);
blower.mouseClicked(airblow);
```

Now if you run this code.

You can see when you click on the button it pushes the fruit towards the right-hand side and the sound of air blowing is played as well.

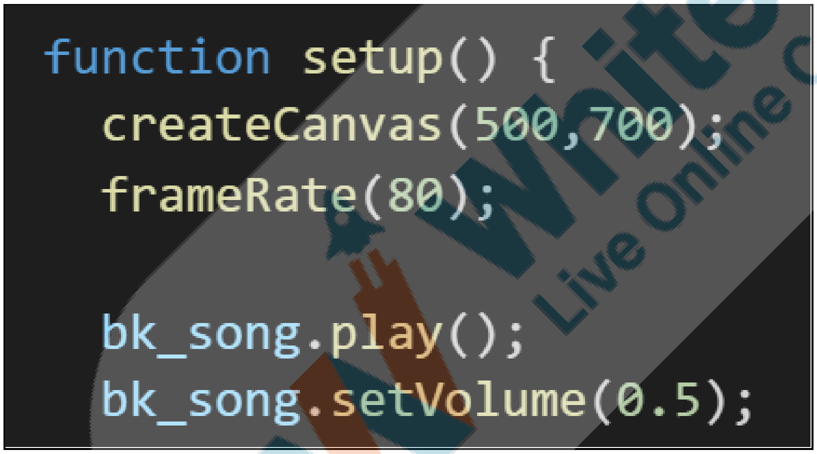
The student runs the code.



Appreciate the student.

Now, let's move onto making our game more interesting!

Games have background music. Now, let's add this background music to our game. This step is very simple, we have loaded sounds in the **preload()** function. We will, now, start playing this background song in a **setup()** function, and this will play continuously. To do that we can say **bk_song.play()**, we also want to set the volume of the song. For that, we will use the **setVolume()** function, in the brackets of the function you will pass the volume percentage, where **0.5** means **50 %** and **1** means **100%**. You can choose any value according to your needs. Here we will set it as **0.5**.	



```
function setup() {
  createCanvas(500,700);
  frameRate(80);

  bk_song.play();
  bk_song.setVolume(0.5);
```

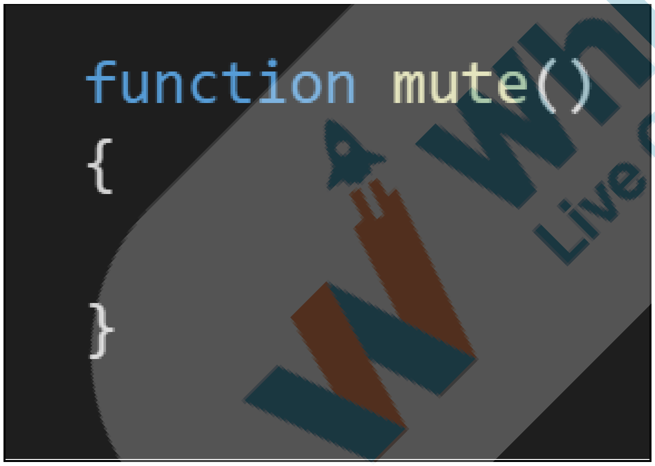
Ask the student to run the code. Here you can hear the background song playing. But we have no control over it. It is going to keep on playing. Can this make it a bit annoying for the user? Do you think we should give the user an option to switch the music on or off?	*The student runs the code to see the output.* **ESR:** Yes

<table>
<tr><td>You must have noticed in games we can turn the music on or off.

Why not do that for our game.
Sounds exciting right!

Can you tell me how can we do that?</td><td><i>The student thinks about it.</i></td></tr>
<tr><td>We will do a similar procedure as we did for the air blower sound.

We will create a function to stop the music, then we will add that function with a button on the canvas.</td><td></td></tr>
<tr><td>First, we will create a function to mute the sound.

Let’s define a function named <b>mute()</b>.</td><td></td></tr>
<tr><td colspan="2">

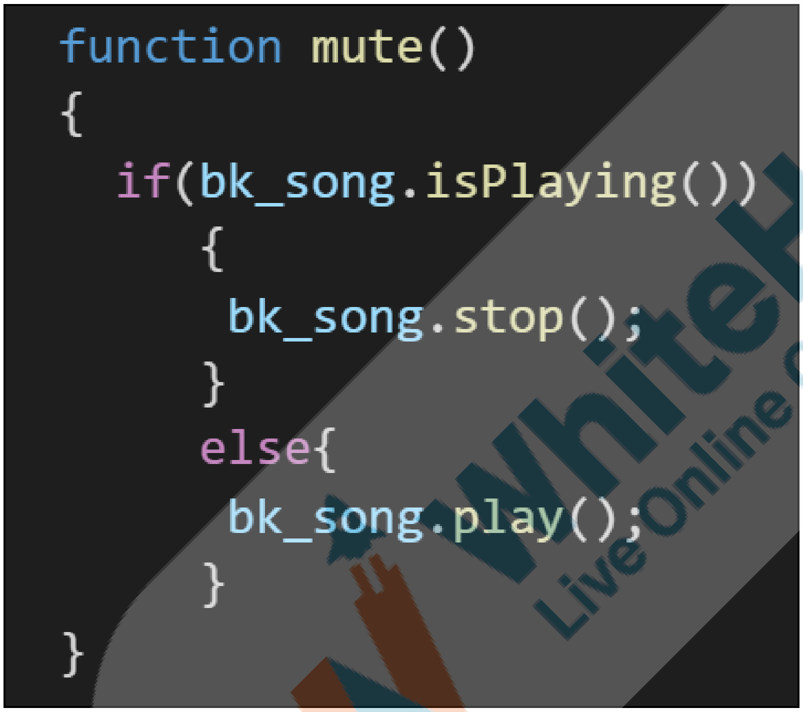
```
function mute()
{

}
```
</td></tr>
<tr><td>To stop any sound, we can use the <b>sound.stop()</b> function, But this would completely stop the sound. We won't be able to play the sound until we restart the game.

We need to make sure that we have the ability to stop and play sound with the press of a button.</td><td></td></tr>
<tr><td>To implement this, we need to test whether the sound is already playing or not.</td><td></td></tr>
</table>

If the sound is playing then we will stop the sound else we will play the sound.
To check whether the sound is playing or not, we have an inbuilt function in p5js as **isPlaying()**.

This will give the output as **true** if the sound is already playing, else it will give **false**.
Let's write this in the **mute()** function.



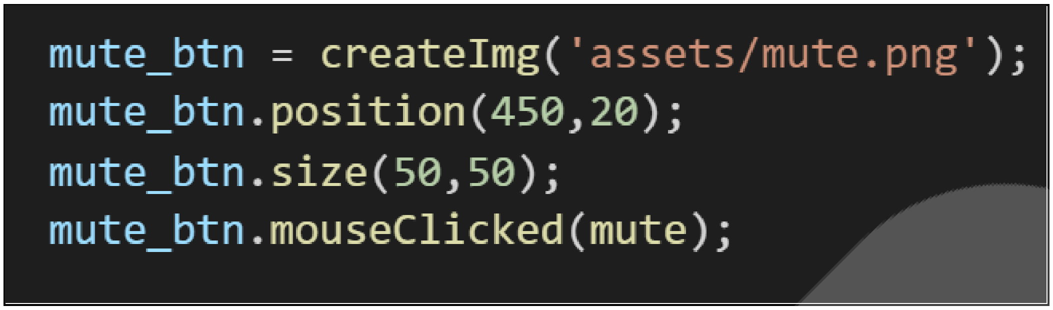
```
function mute()
{
   if(bk_song.isPlaying())
      {
       bk_song.stop();
      }
      else{
       bk_song.play();
      }
}
```

Now, the function is complete.

Let's attach this with a button image, just like we did for the air blower.

Declare a variable first as **var mute_btn;**

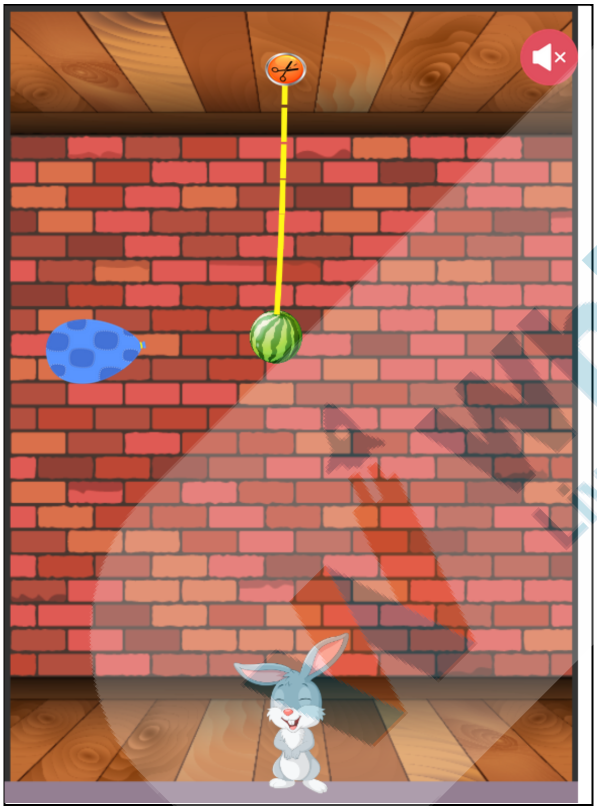
Then in the **setup()** function we create the image button and attach it with the **mute()** function.



```
mute_btn = createImg('assets/mute.png');
mute_btn.position(450,20);
mute_btn.size(50,50);
mute_btn.mouseClicked(mute);
```

Run the code and click on the sound icon to see if we can play and stop the sound.

The student runs the code and tries out the new button.



Appreciate the student.

Very well done!

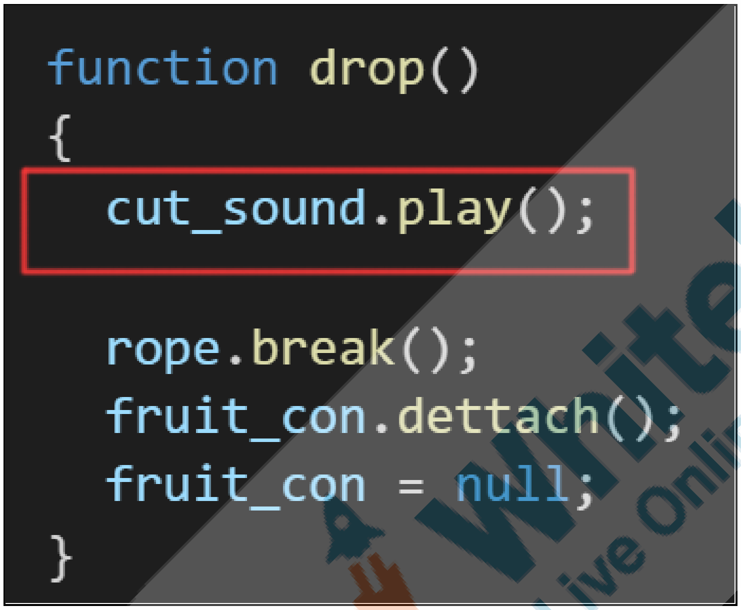
Now we need to finally add three more sound effects:

1. When we cut the rope.

2. When the bunny eats fruit.
3. When the fruit drops on the ground and the bunny is sad.

First, we add the sound when the user cuts the rope.

We already have the function **drop()** for it, we will just add **sound.play()** in this function, and it will be called when the user presses the cut button.



```
function drop()
{
   cut_sound.play();

   rope.break();
   fruit_con.dettach();
   fruit_con = null;
}
```

We have our function ready. Now, run the code.

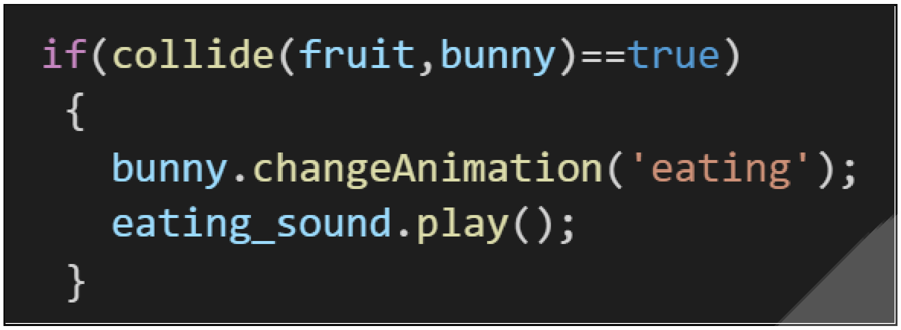
Each time you click on the cut button, it will play the sound.

The student runs the code and observes the output.

Now let's add the sound when the bunny eats the fruit.

This sound needs to be added where we are detecting the collision and changing the animation.

This is very easy in the **draw()** function to go to the collision condition and add the code **eating_sound.play()**.

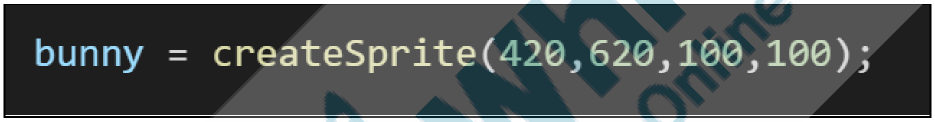


```
if(collide(fruit,bunny)==true)
 {
   bunny.changeAnimation('eating');
   eating_sound.play();
 }
```

Now we are left with the sound of the fruit landing on the ground and missing the bunny.

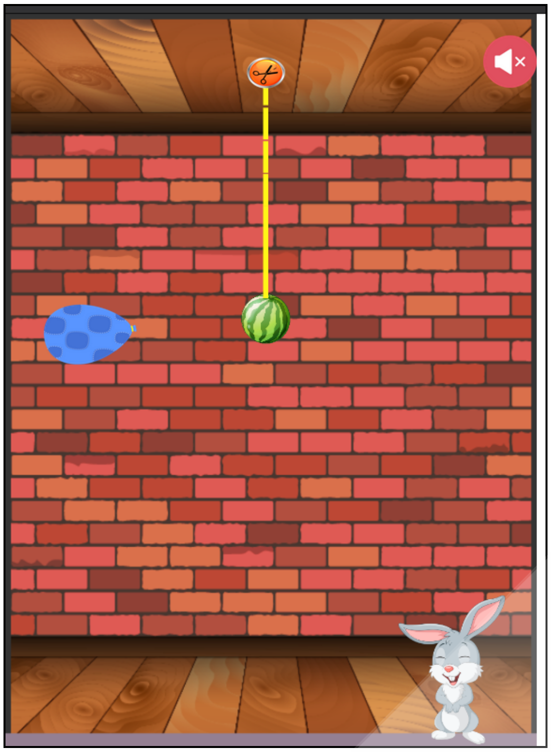
But before we add the sound, let’s move the bunny on the right side as a challenge for the user which we mentioned earlier, because if the bunny is placed directly below the fruit it won’t be a fun game.

For that, we just need to update the x position, where we created the bunny sprite.



```
bunny = createSprite(420,620,100,100);
```

Run the code to see the output.

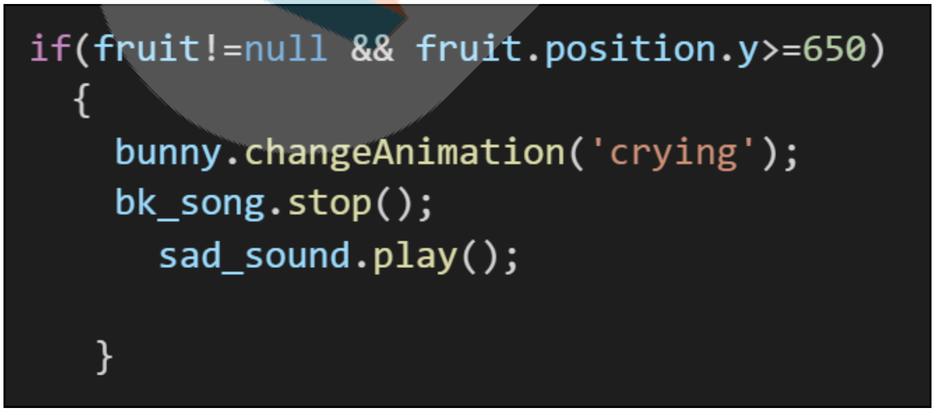


It looks very challenging now to feed the bunny, right?

Now, finally, we will add the sound of a sad bunny then you can play the game.

We are already checking the collision of the bunny and ground.

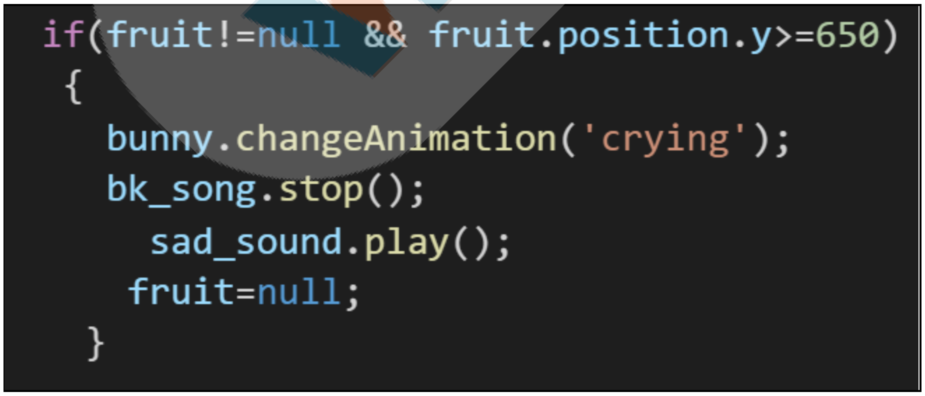
We will add the **sad_sound** in this condition itself.



```
if(fruit!=null && fruit.position.y>=650)
  {
    bunny.changeAnimation('crying');
    bk_song.stop();
      sad_sound.play();

     }
```

Run the code and click on the cut button so that it misses the bunny. Do you hear anything wrong?	**ESR:** The sad bunny sound is now playing continuously.
Why is this happening? When we are detecting the collision of fruit and ground we have one condition as **fruit!=null**, which is **true**, and the other condition of the fruit position being greater than **650** is also **true**, that is why the sound is playing endlessly. What can we do here to overcome this bug?	**ESR:** Varied. **ESR:** Varied
We can make one condition **false**, then it will stop acting like this. We will set, **fruit=null;** due to this, the above condition **if(fruit!=null && fruit.position.y>=650)** will be **false**. Since, we are using an **AND** operator, in this, both the **if** conditions should be **true**. If we set **fruit=null,** the above condition will be **false** and the following code will not execute, thus stopping the continuous music being played. So our sound will only play once which we desire.	

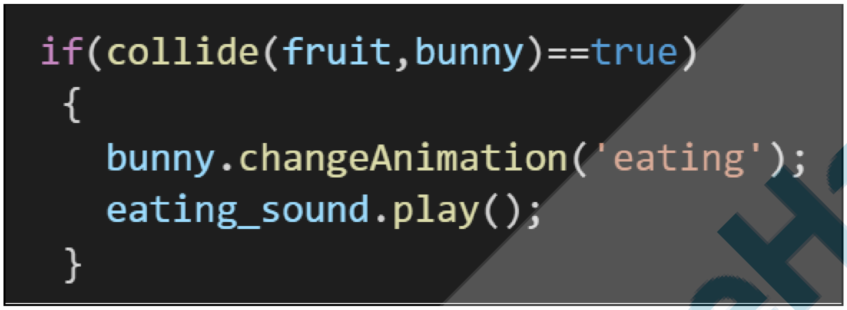


```
if(fruit!=null && fruit.position.y>=650)
 {
   bunny.changeAnimation('crying');
   bk_song.stop();
     sad_sound.play();
    fruit=null;
  }
```

Run the code and observe the output. We now have to use the balloon to push the fruit towards the bunny. If you feel it is very difficult, we can increase the amount of force applied to the fruit. Now you play with this.	
We came to the end of the session. You did great work! Please stop sharing the screen.	

Teacher Guides Student to Stop Screen Share	
WRAP-UP SESSION - 5 Mins	
Teacher starts slideshow from slide 28 to slide 37	
Activity details	**Solution/Guidelines**
Run the presentation from slide 28 to slide 37 **Following are the WRAP-UP session deliverables:** ● **Explain the facts and trivias** ● **Next class challenge** ● **Project for the day** ● **Additional Activity**	Guide the student to develop the project and share it with us.
Teacher ends slideshow	
Quiz time - Click on in-class quiz	
Question	**Answer**

To stop any sound, we use____________ A. sound.end() B. sound.play() C. sound.stop() D. stopSound()		**C**

What does the following snippet of code do?

`if(collide(fruit,bunny)==true)`
`{`
`bunny.changeAnimation('eating');`
`eating_sound.play();`
`}`
A. When the bunny eats, the fruit bunny animation changes, and sound is played.
B. When the fruit falls, the bunny animation changes, and the sound is played.
C. When the rope is cut, the bunny animation changes, and the sound is played.
D. None of the above.
		A

To set the volume of the music in our game to 50%, which of the following commands is used? A. setVolume(50%) B. setVolume(50) C. setVolume(0.5) D. setVolume(1)		**C**
End the quiz panel		
	You get hats off.	Make sure you have given at least 2 Hats Off during the class for:

	See you in the next class then.	
*** This Project will take only 30 mins to complete. Motivate students to try and finish it immediately after the class.** **Project Overview** **BLOWER PIPE** **Goal of the Project:** In class 32, we learned how to apply force on the body by pressing a button. In this project, we will create a blower pipe game using the same concepts. **Story:** Jenny went to a carnival fest There she saw a toy which was quite interesting as to win she had to continuously keep blowing into the pipe to keep the ball in the air. When she got home, she thought of creating that toy virtually. Can you help her to make that toy? I am excited to see your project! Bye Bye!		*The student engages with the teacher over the project.*

Links:

Activity	Description	Link
Teacher Activity 1	Template code	https://github.com/pro-whitehatjr/C32_TA_1
Student Activity 1	Template code	https://github.com/pro-whitehatjr/C32_SA_1
Reference code	Reference code	https://github.com/pro-whitehatjr/c32-game_code
Project	Project Solution	https://github.com/pro-whitehatjr/Project_Solution_C32
Teacher Reference visual aid link	Visual aid link	https://curriculum.whitehatjr.com/Visual+Project+Asset/PRO_VD/BJFC-PRO-V3-C32-withcues.html
Teacher Reference In-class quiz	In-class quiz	https://s3-whjr-curriculum-uploads.whjr.online/525fbf2e-fca9-47b4-b5b6-8e4fbd47dce1.pdf
Project Solution	Blower Pipe	https://github.com/pro-whitehatjr/Project_Solution_C32