Topic	Adding More Challenges	
Class Description	**The student learns to attach the fruit with multiple ropes and make the game mobile compatible. The student will also create an APK of the game using a web wrapper.**	
Class	**C33**	
Class time	**45 mins**	
Goal	• Add multiple ropes • Dynamic screen size. • Host the game on GitHub.	

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	

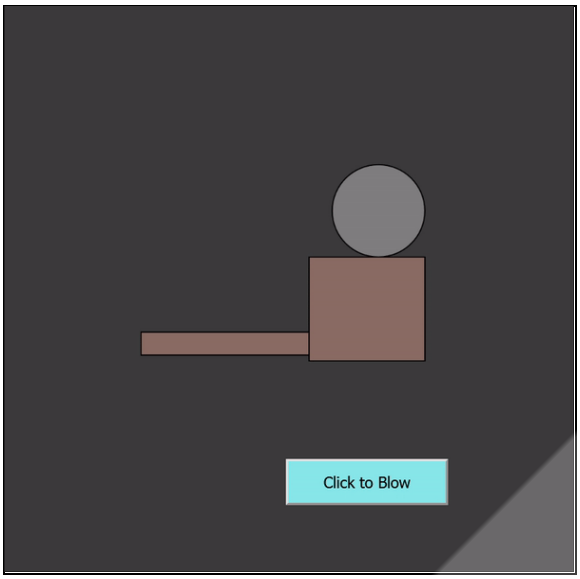
Class structure	**Warm-Up - Slide show option** **Teacher-Led Activity** **Student-Led Activity** **Wrap-Up - Slide show option**	**10 Mins** **10 Mins** **20 Mins** **5 Mins**

WARM-UP SESSION - 10mins

Teacher starts slideshow  **from slides 1 to 10**
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 3.*** **The following are the warm-up session deliverables:** • **Greet the student.** • **Revision of previous class activity.** • **Quizzes**	**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 use **Matter.Body.ApplyForce()** to apply vertical force to the **ball**. A. `Matter.Body.applyForce(ball.body, {x:0, y:0}, {x:0, y:-0.05});` B. `Matter.Body.applyForce(ball.body, {x:0, y:0}, {x:0, y:0.05});` C. `Matter.Body.applyForce(ball.body, {x:0, y:0}, {x:0.05, y:0.05});` D. `Matter.Body.applyForce(ball.body, {x:0, y:0}, {x:-0.05, y:0});`	**B**
Select the correct option to call **blow()** function on the **mousePressed()** property of the button.	**A**



A.
```
//button.mousePressed(blow);
```

B.
```
//buttonPressed(blow);
```

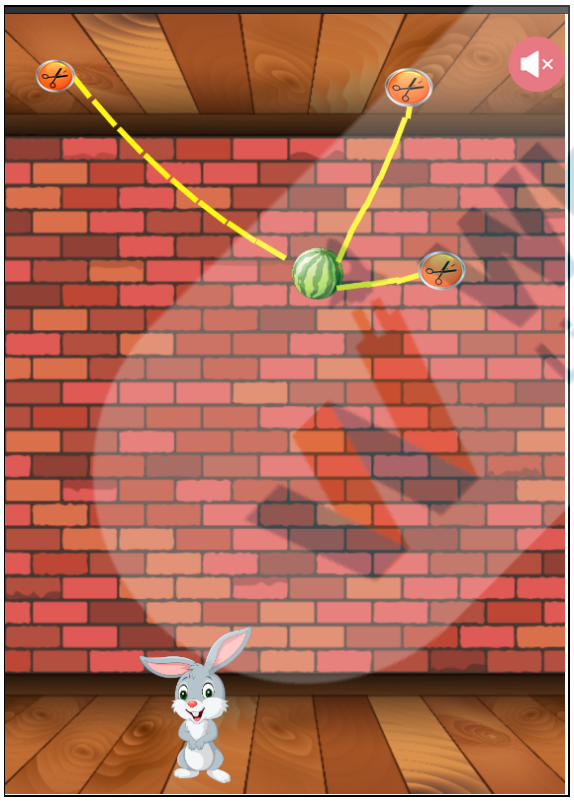
C.
```
//button = mousePressed(blow);
```

D.
```
//button.mousePressed();
```

Continue the warm-up session

Teacher Action	Student Action
Run the presentation from slide 4 to slide 10 to set the problem statement. **The following are the warm-up session deliverables:** ● Appreciate the student on his performance in the quizzes. ● Add two more ropes.	Narrate the slides by using hand gestures and voice modulation methods to bring in more interest in students.

• Create the constraints. • Detach functions.	
Teacher ends slideshow 	
TEACHER-LED ACTIVITY - 10 mins	
Teacher Initiates Screen Share	
CHALLENGE • **Add two more buttons and ropes on the screen.** • **Create the function to drop the fruit from the additional ropes.**	
Teacher Action	**Student Action**
Teacher Activity 1 In the last few classes, we learned and implemented a lot of concepts related to the physics engine. Now we are in the last session of making our game “Feed the bunny”. We will make this game a bit more challenging for the user to play, and we are also going to make this game mobile compatible so that we can host this game on GitHub and wrap it in an APK, so you can play it on your mobile phone and share it with your friends. We move onto the first challenge, which is to add multiple ropes. But, here we will not be using the balloon to push the fruit. Rather, we are going to create an arrangement that the user has to think about which rope should be cut so that fruit lands on the bunny. The user can do that in many ways by applying your creativity. We are going to create three ropes, after which we need to create buttons for each rope and call the function to drop	*The teacher downloads the Teacher Activity 1 code and opens it in the VS Code editor.*

the fruit. We already have a button and rope on the canvas, but we need to change its position. We have changed the position of the bunny in such a way that the user has to think in what order the ropes have to be cut so that fruit lands on the bunny. Let's place all the buttons on the canvas, that will make it easy for us to place the ropes.	
We will create three buttons, one is going to be at the far left-hand side of the canvas, the next two buttons will be on the right-hand side, but one button is close to the top and the other is at the center of the canvas. **Note**: Use the below image for reference.	
	
This will be clear once we start placing the buttons. All the buttons will have the same background image. But	

we need to attach them with different functions.

One function **drop** is already defined, **drop2** and **drop3** functions we will define later.

```
//btn 1
  button = createImg('cut_btn.png');
  button.position(20,30);
  button.size(50,50);
  button.mouseClicked(drop);

   //btn 2
   button2 = createImg('cut_btn.png');
   button2.position(330,35);
   button2.size(60,60);
   button2.mouseClicked(drop2);

   //btn3
   button3 = createImg('cut_btn.png');
   button3.position(360,200);
   button3.size(60,60);
   button3.mouseClicked(drop3);
```

We got the buttons on the screen, now we need to add two more ropes.

Creating the additional ropes is very easy, first define two more variables such as **var rope2** and **var rope3;**

Next, in the **setup()** function we can assign the **rope** object to these variables.

While creating the **rope** object we need to tell how many sections(elements) each rope will have, this will control the length of the rope.

The second thing is we need an x and y position where the rope will be connected on the canvas.

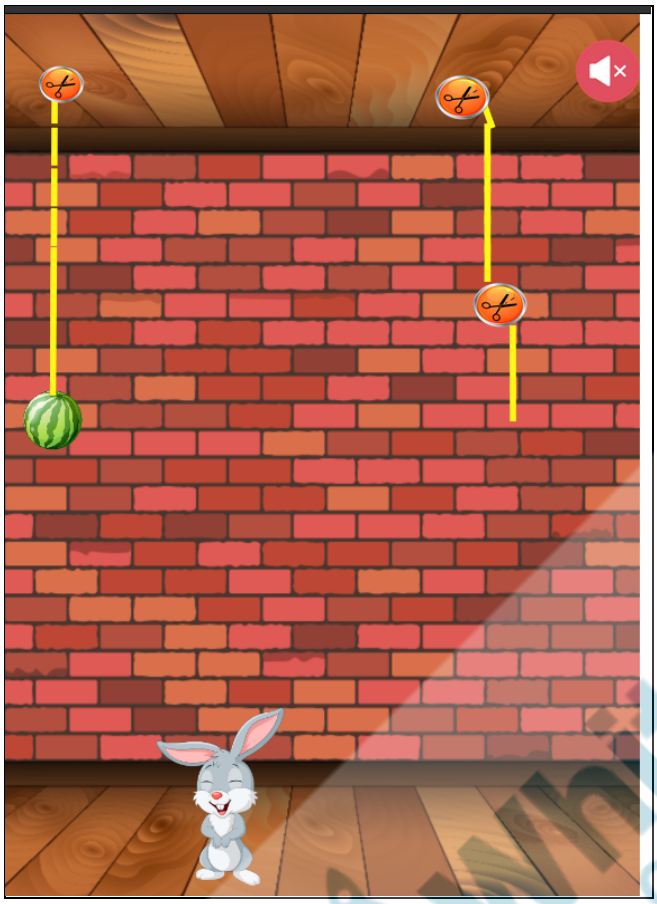
```
var rope2, rope3;
```

```
rope = new Rope(8,{x:40,y:30});
rope2 = new Rope(7,{x:370,y:40});
rope3 = new Rope(4,{x:400,y:225});
```

Now we need to display these ropes on the canvas by calling the **rope.show()** function for each object.	

```
rope.show();
rope2.show();
rope3.show();
```

Calling these functions will display the ropes, but when we run the program we can see that only one rope is attached with fruit. *The teacher runs the code and shows the output.*	*The student observe and learn.*



To attach the fruit with the other two ropes, we need to create the constraint between them.

Can you tell me how can we do that?

Very good!

We have made a **Link** class, we just need to create the object of that class and while creating the object we need to pass the two bodies.

Here bodies will be **rope** and **fruit**, we already have a constraint between **rope** and **fruit**, now we just need to create this constraint between **rope2** and **rope3**.

Declare variables as **var fruit_con_2,** and **fruit_con_3;**

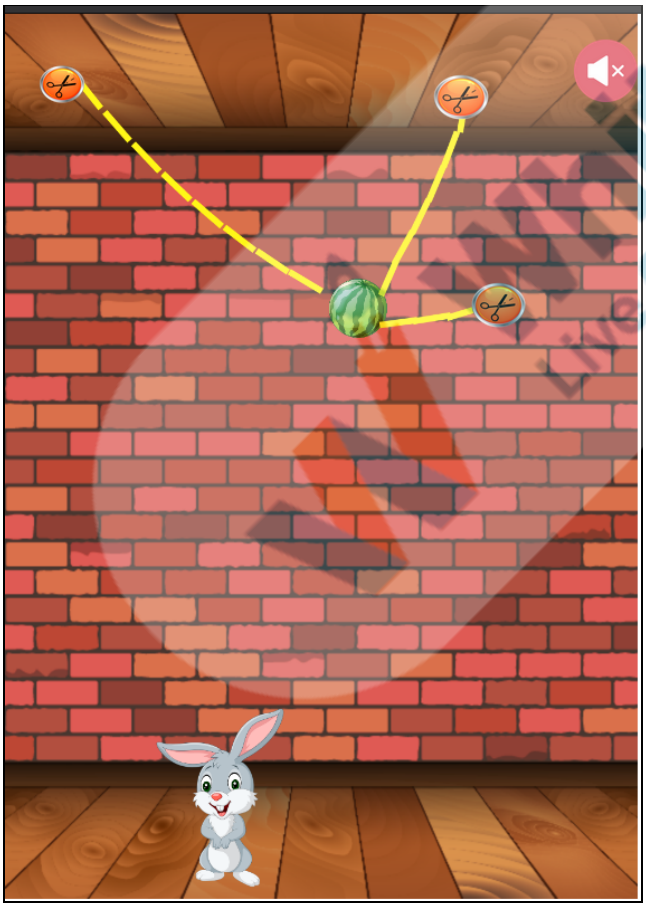
ESR:
By using the **Link** class.

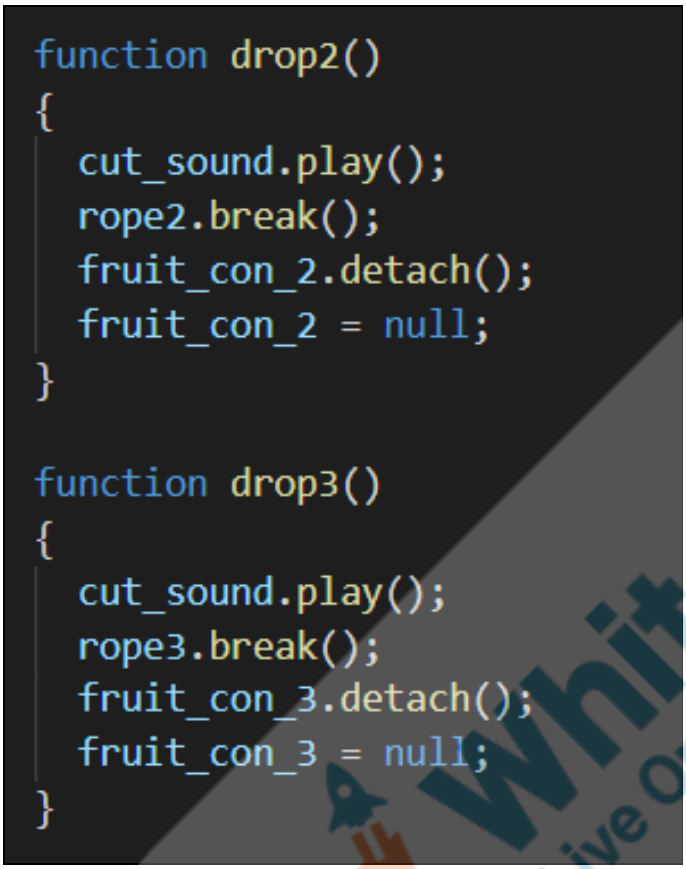
In the **setup()** function, assign the **Link** object to these variables.

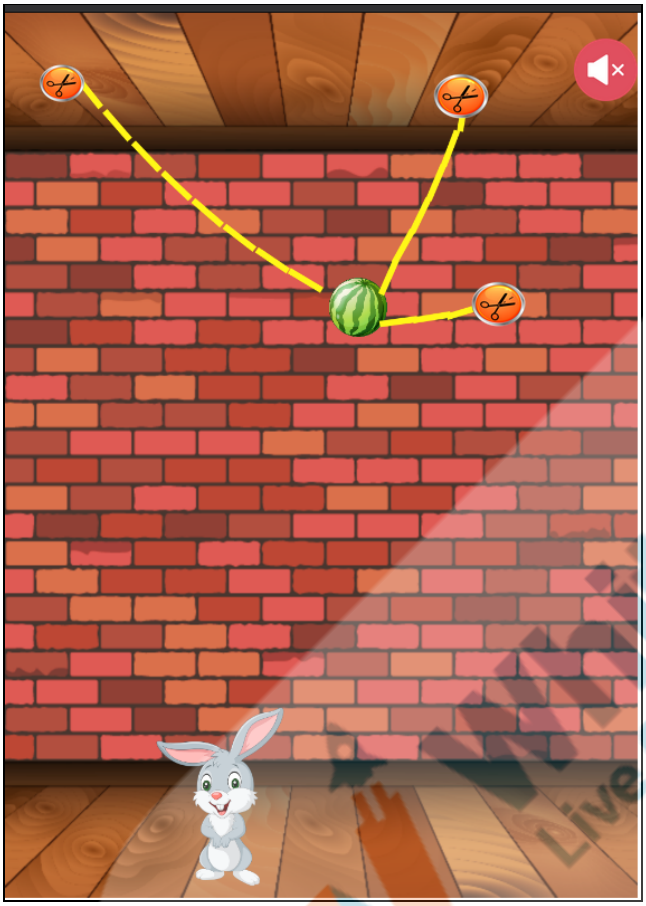
```
fruit_con = new Link(rope,fruit);
fruit_con_2 = new Link(rope2,fruit);
fruit_con_3 = new Link(rope3,fruit);
```

Now when we run the code, we can see that fruit is attached with all the ropes.

Now, we need to write the code to drop the fruit from the other two ropes.



We have already attached the function with the buttons, now we will define two more functions to drop the fruit.	
	
In the functions, we are calling the **break()** function from the **rope** class and the **detach()** function from the **Link** class. After that, we are setting the constraint as **null** so that it does not interfere with our fruit body. Now if we press on any button, the rope will detach from there and fruit falls down. We have to cut the rope carefully so that fruit lands on the bunny.	
At this point, we have our game is ready.	

Now the task is to make this game compatible with mobile phones and to host it online so that we can make an installable app out of it.	
	
Now it's your turn, please share your screen with me.	

Teacher starts slideshow :Slide 11 to Slide 16	
Run the presentation for slides to set the student activity context.	
• Make the game compatible with all screen size. • Upload files on GitHub. • Host the game on GitHub page.	

• Create an APK using Thunkable.	

Teacher ends slideshow 	
Teacher Stops Screen Share	
STUDENT-LED ACTIVITY - 20mins	
• **Ask Student to press ESC key to come back to panel** • **Guide Student to start Screen Share** • **Teacher gets into Fullscreen**	
ACTIVITY • **Make the game mobile phone compatible.** • **Host on GitHub.** • **Create an installable game file like APK.**	
Teacher Action	**Student Action**
Student Activity 1: Our game is ready, we have added sounds, and multiple ropes to make it complex as well. But it only works on our computer. What if you want to share your game with your friends? How can you do it? Great! We have done this with the T-rex game. So we are going to follow the same process here, which is: 1. Make the game compatible with different screen sizes. 2. Host on GitHub.	*The student downloads the student Activity 1 code and opens it in the VS Code editor.* **ESR:** By hosting it on GitHub.

3. Make an installable file with Thunkable. To make the game adapt to any screen size, we already have inbuilt variables in p5js. When we create the canvas we specify the screen size such as **400** width and **400** height. In our case, we had **500** and **700**. But if the screen size is changing, this will create problems. To overcome that, we will use the **windowWidth** and **windowHeight** variables instead of having a fixed screen size. This will automatically change the screen size based on the size of the screen of the user. But here is the problem, this will only work for a computer screen. This won't work with the mobile screen. For mobile screens, we need to use **displayWidth** and **displayHeight,** and we are going to use both of these. We will create a variable var **isMobile**, which will store the type of mobile such as iphone, ipad or android. Then we will write a condition that will check the device we are currently on, if we are on a mobile device it will use **diplayWidth** and **displayHeight**, if we are on a computer screen it will use **windowWidth** and **windowHeight**. Now, create two variables as **var canW** and **canH**, these will store the width and height of the screen. **Note:** We are adding **80** to the **displayWidth** in order to properly fit the screen width. As in some devices, game width is lesser than the screen width. If you feel the game is stretched too much in the mobile, you can remove the **80** or reduce the value to an appropriate number.	

```
function setup()
{
  var isMobile = /iPhone|iPad|iPod|Android/i.test(navigator.userAgent);
  if(isMobile){
    canW = displayWidth;
    canH = displayHeight;
    createCanvas(displayWidth+80, displayHeight);
  }
  else {
    canW = windowWidth;
    canH = windowHeight;
    createCanvas(windowWidth, windowHeight);
  }
```

This will create a dynamic screen size. But it will not work. Because we need to add a very important line of code in our HTML file that will make the webpage screen responsive.

```
<!DOCTYPE html><html><head>
  <script src="p5.min.js"></script>
  <script src="p5.dom.min.js"></script>
  <script src="p5.sound.min.js"></script>
  <link rel="stylesheet" type="text/css" href="style.css">

  <meta name="viewport" content="user-scalable=no,initial-scale=1,
maximum-scale=0.825,minimum-scale=0.8,width=device-width">

  <meta charset="utf-8">

</head>

<body>
  <script src="matter.min.js"></script>
  <script src="p5.play.js"></script>
  <script src="rope.js"></script>
  <script src="ground.js"></script>
  <script src="link.js"></script>
  <script src="sketch.js"></script>

</body></html>
```

This line will make sure that our game will adapt to any screen size. This will set the game width and height according to the screen size of the device	
Now, we need to change a few more things, such as the position of the bunny and the ground. The current position we set for the bunny and ground is by assuming a screen size of **500** x **700**, but since we made it screen adaptive, it will place the bunny and the ground at different locations which may not be desirable to us.	

```
ground = new Ground(200,canH,600,20);
  blink.frameDelay = 20;
  eat.frameDelay = 20;

  bunny = createSprite(170,canH-80,100,100);
  bunny.scale = 0.2;
```

Here we are placing the ground and bunny with respect to the new screen size.

So the ground y-position will be **canH** which is the height of the screen.
And the y-position of the bunny will be **canH-80**. So the bunny will be placed on the ground.

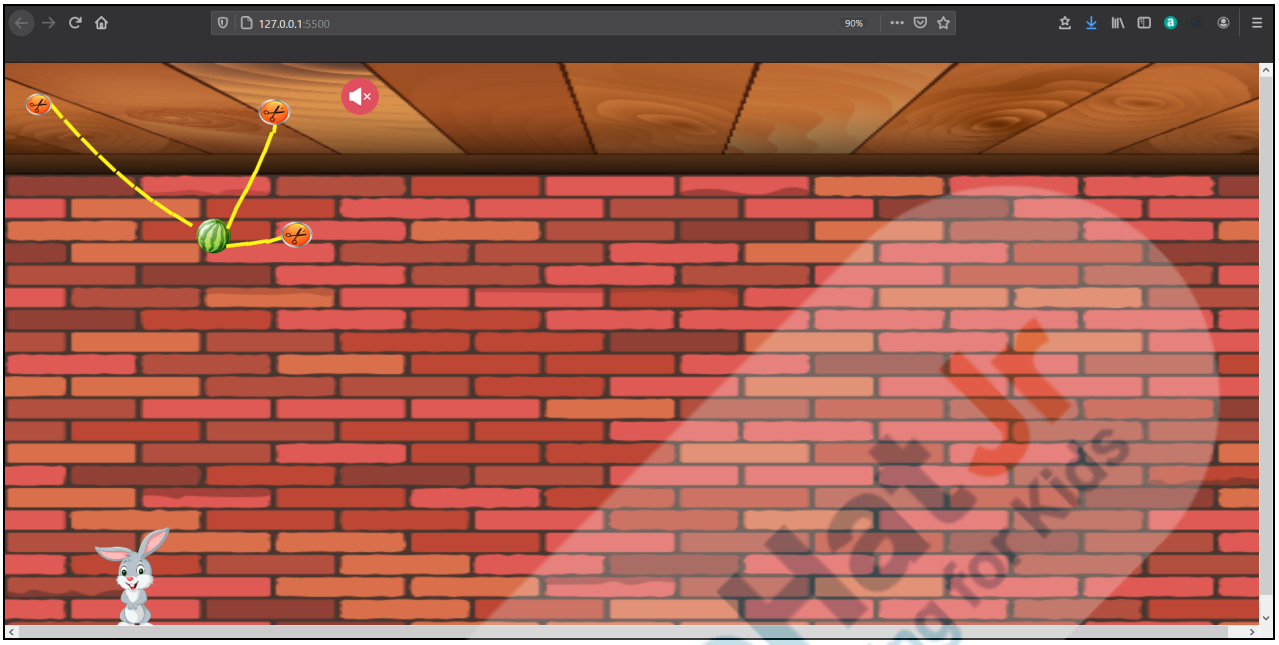
One more important change we need to make in the background image, the background image is placed at **0**,**0** which is the top left corner of the canvas. But the image size will be fixed as **500** x **700px**.

This will not cover our entire screen if the screen size is bigger. But we can specify the image size as well, and here we will do that using **displayWidth** and **displayHeight**.

We will also add some value in **displayWidth** to make sure the image covers the entire screen size.

```
function draw()
{
  background(51);
  image(bg_img,0,0,displayWidth+80,displayHeight);
```

If you run the code we can see the image is covering the whole screen.

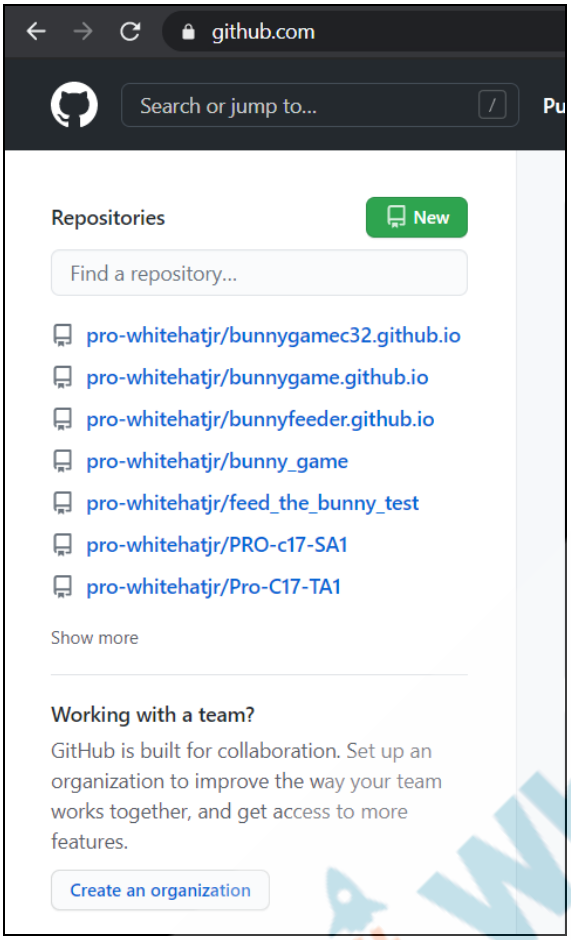


Our coding part is complete. Now let's host the game on GitHub.

For that, we first need to create a repository, and then we will create a GitHub page for our repo.

Open https://github.com/ and log in to your account.

Click on the **New** button to create a repository.



Now we need to give a name to our repository.

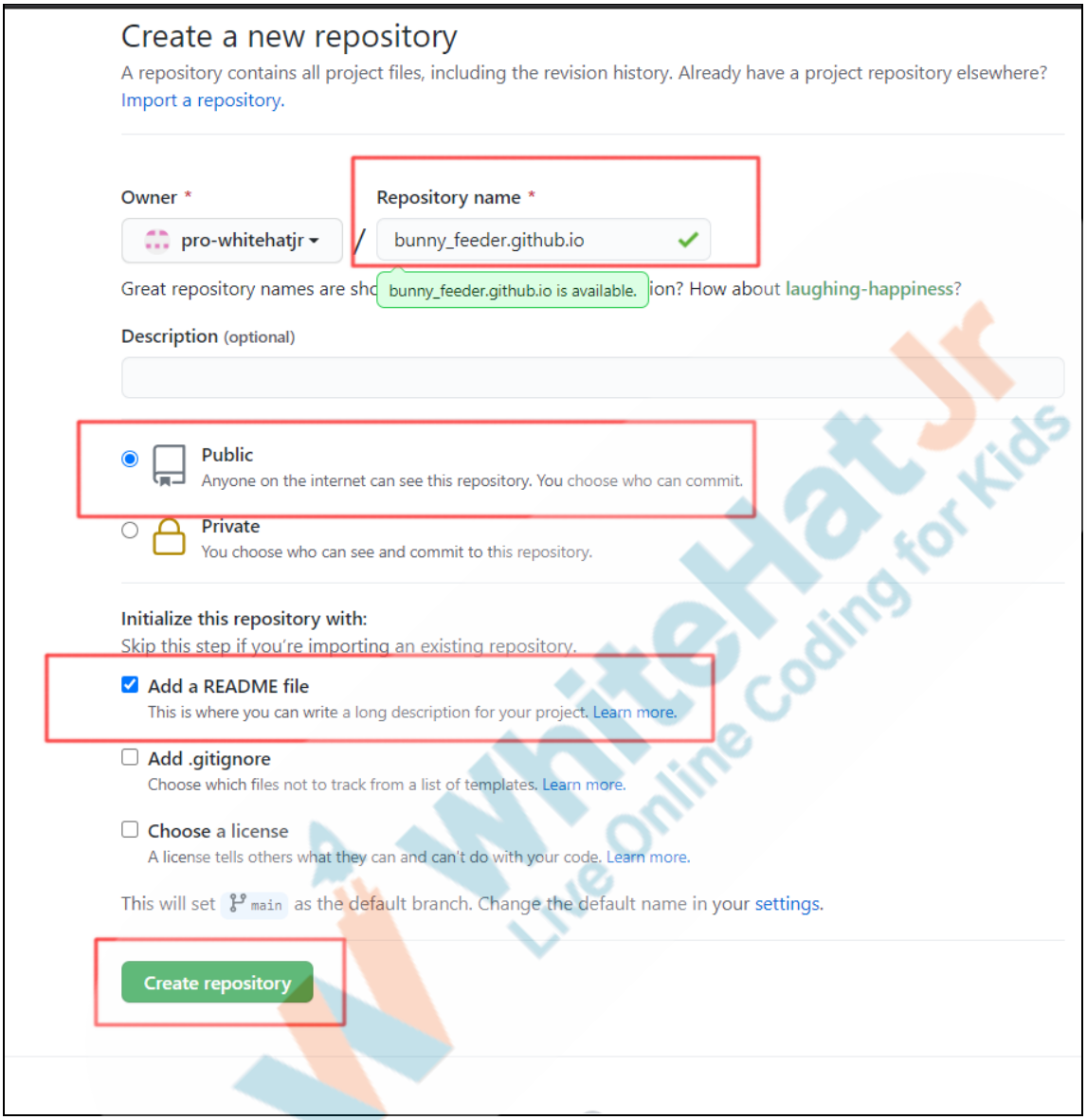
Here we have to be very careful because we want to host this game on GitHub pages. When we do that, the repository name should end with **.github.io**.

Note: You can set any name for the repo but at the end of it, it should be **.github.io**.

We can now set it as **bunny_feeder.github.io**.

Our repository should be set to the public and check the **Add a README file** box.

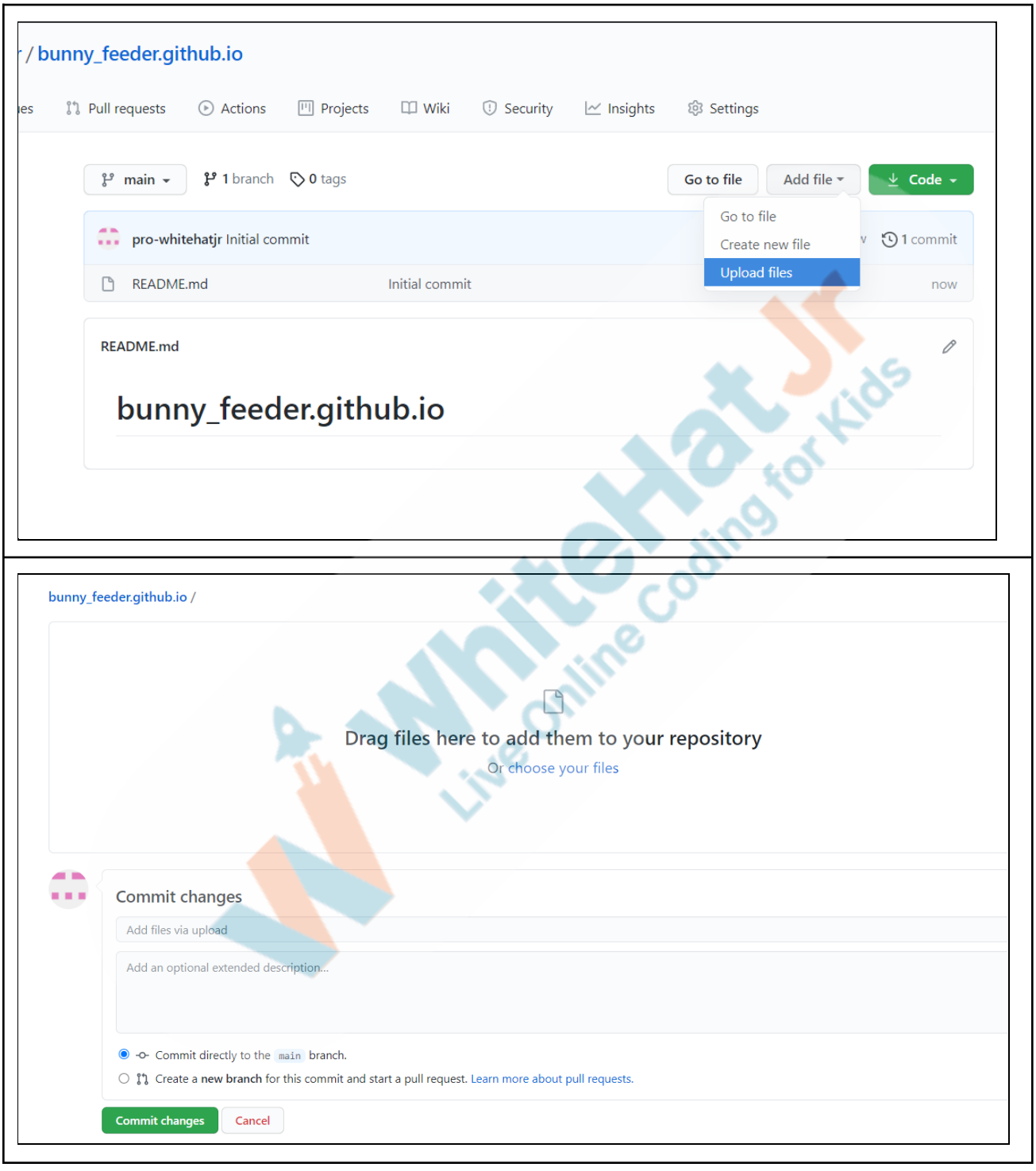
After that, click on the button **Create repository**.

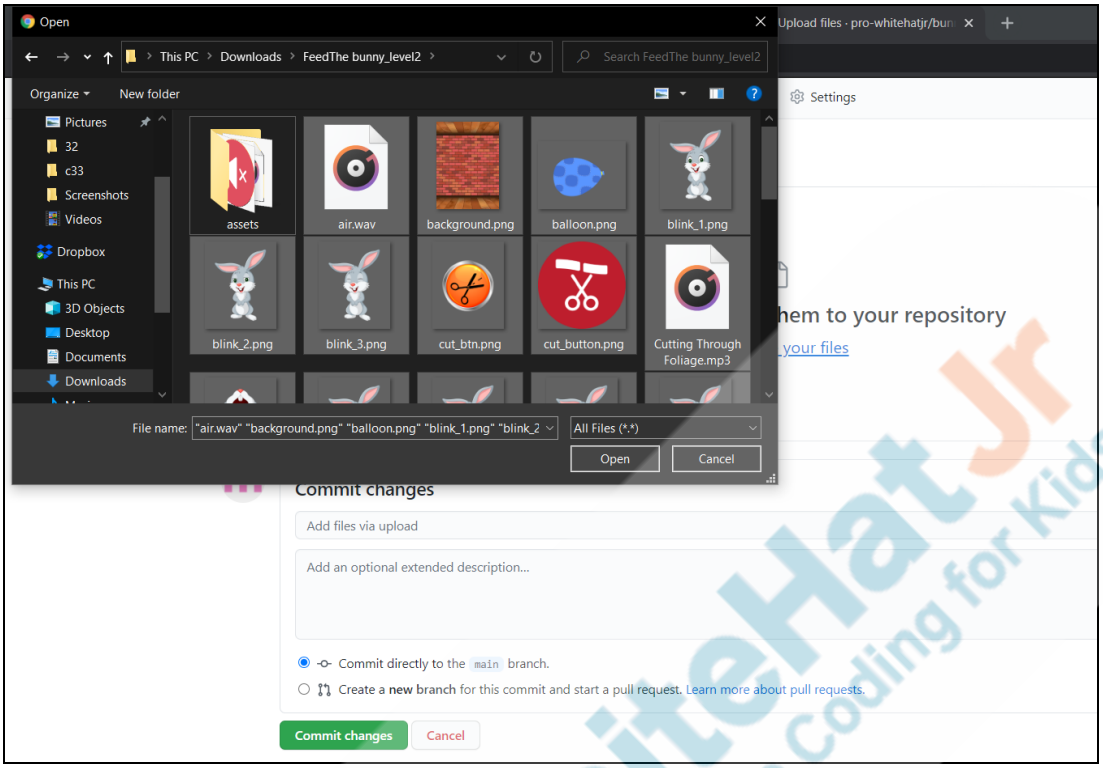


Now we are ready to add files to this repository.

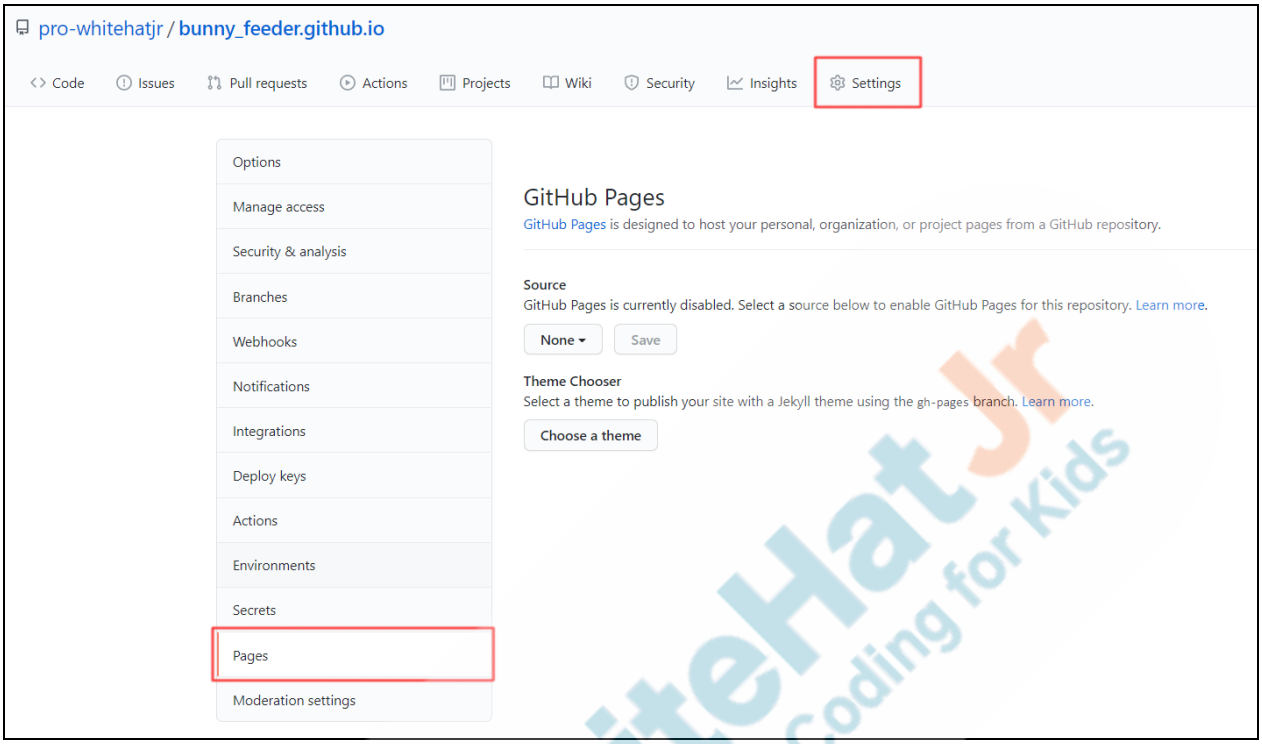
Click on **Add file** and from the dropdown menu click on **Upload files**. This will give the option to choose files from the computer.

Select the folder from our computer and select all the necessary game files.





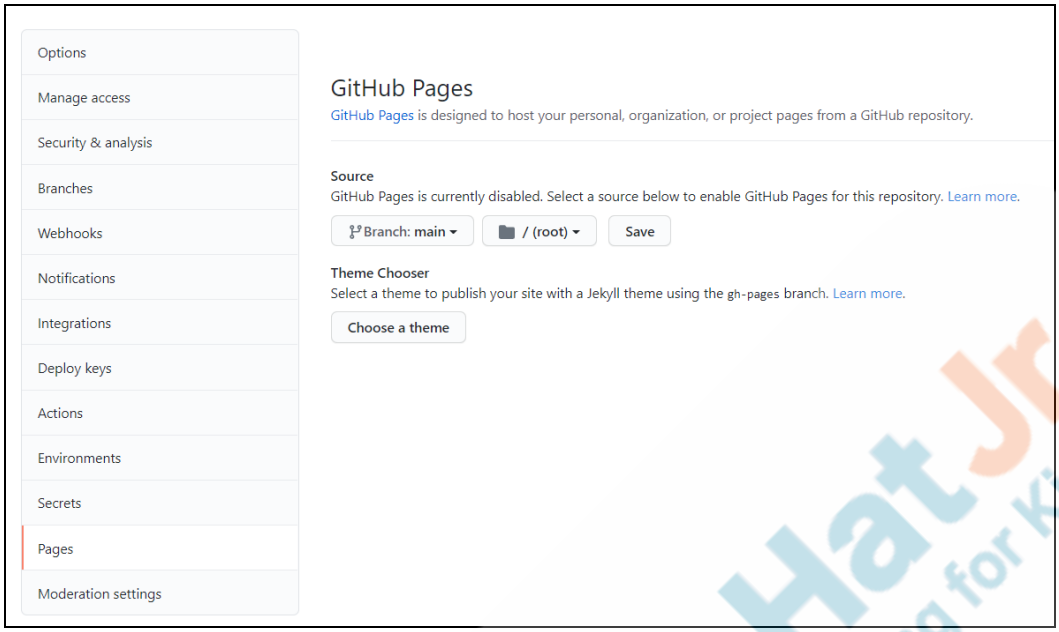
Once all the files are uploaded, check the option **commit directly** to the main branch and then click on **Commit changes**. Click on the **Settings** button and select the **Pages** tab.	

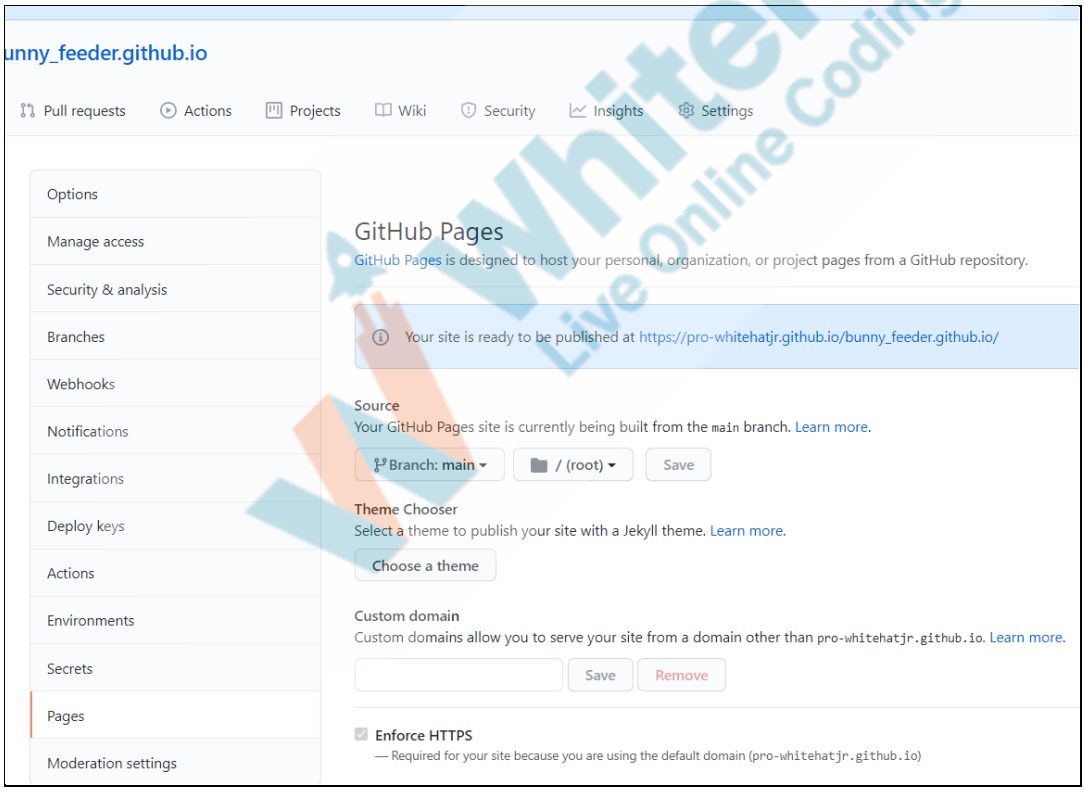


Here we will host our game on the GitHub pages.

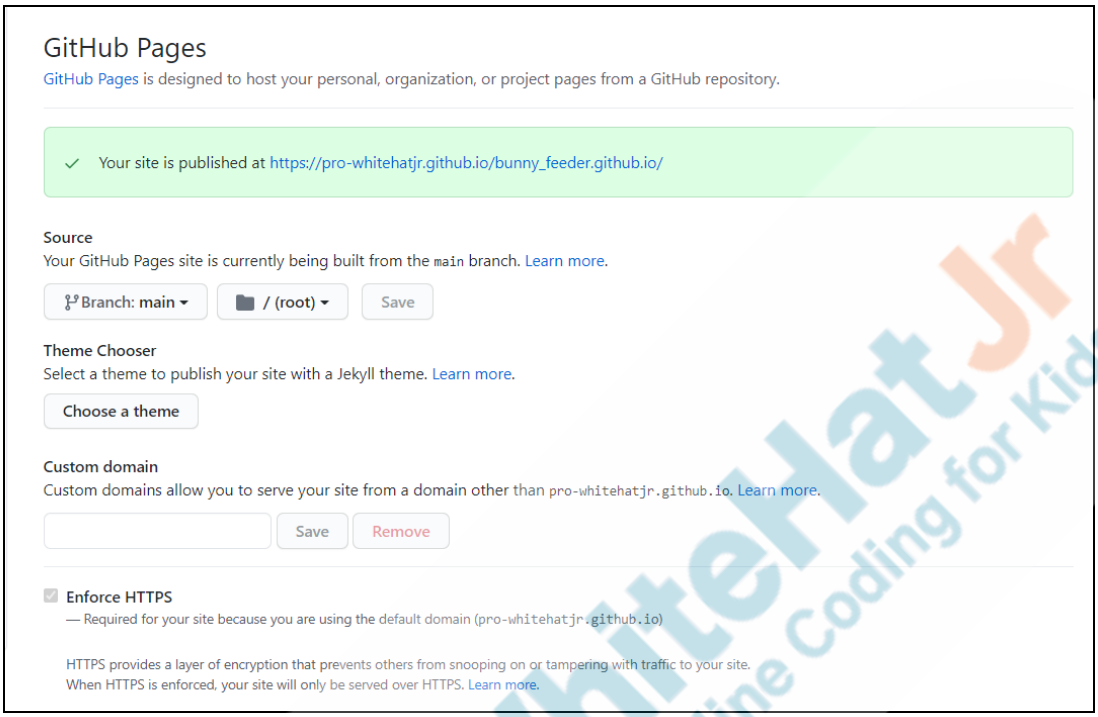
Select the source as the **main branch**, and it will choose the root folder by default, then click on **Save**.

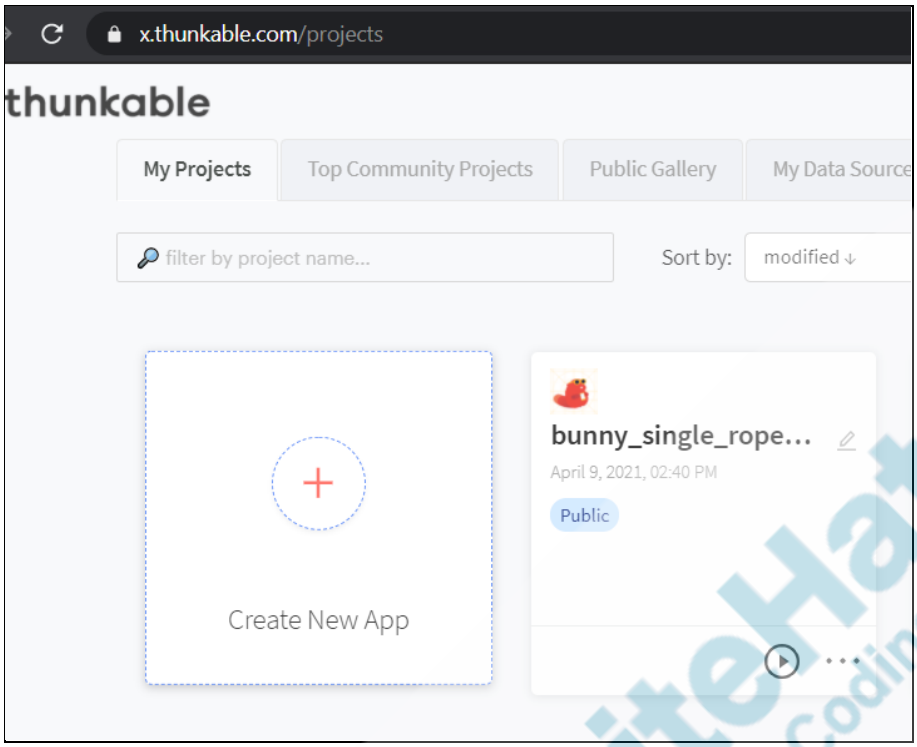
This will reload the page and a link should appear. Our game will be available at this link. It may take a few minutes to publish the game, when our game is published the link will turn green from blue.

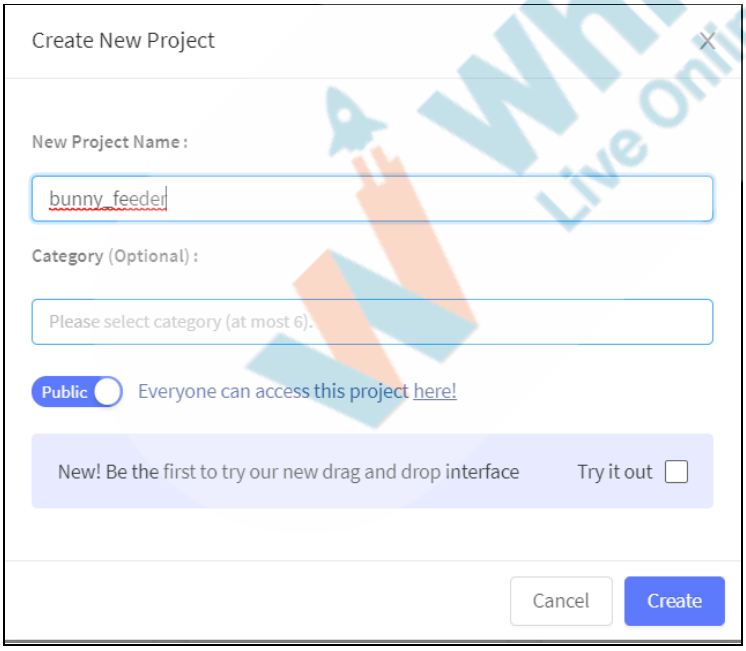




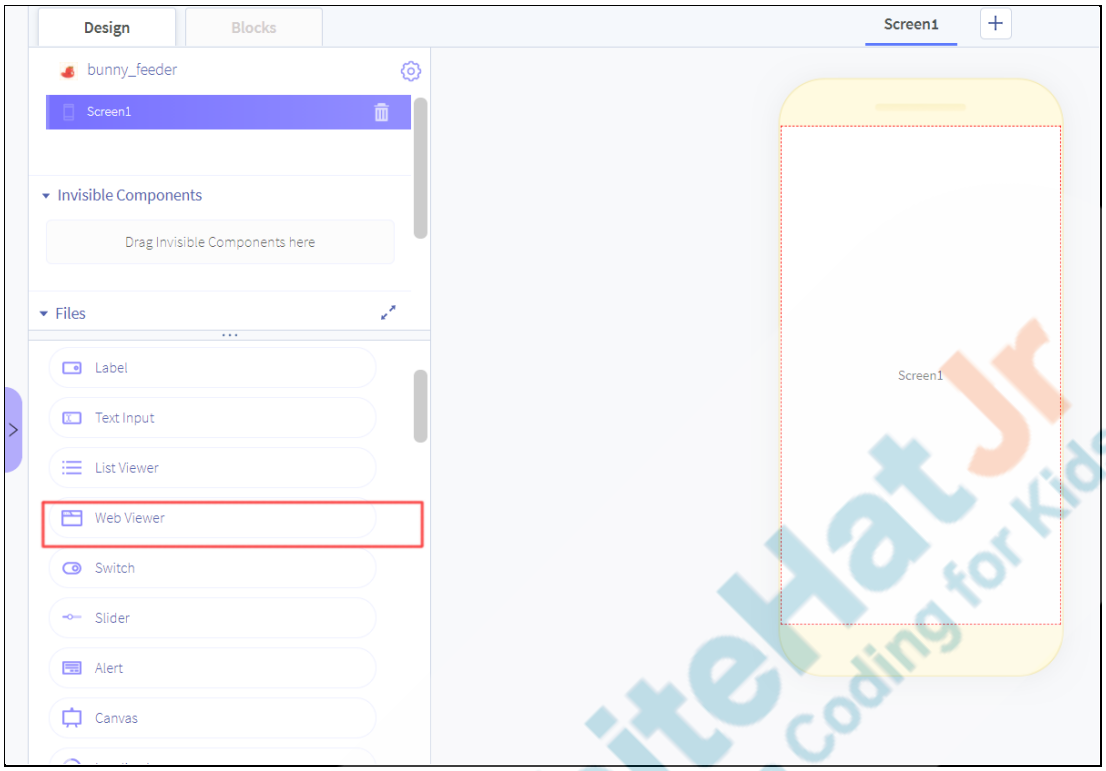
Refresh the page and now our game is published on the GitHub pages and the link is highlighted as green.

You can click this link to open and play the game, you can even share this, and it will work in any mobile browser too.	
	
Copy this link and keep it saved in any notepad file or in sticky notes because we are going to use this link to create an APK file using Thunkable. Click on the link, and it will open your game in the browser. We move onto the final step for this class now: Open www.thunkable.com and log in to your account. Click on the **Create New App** button to **Create a New Project**. Choose the name for the app and click on the **Create** button.	

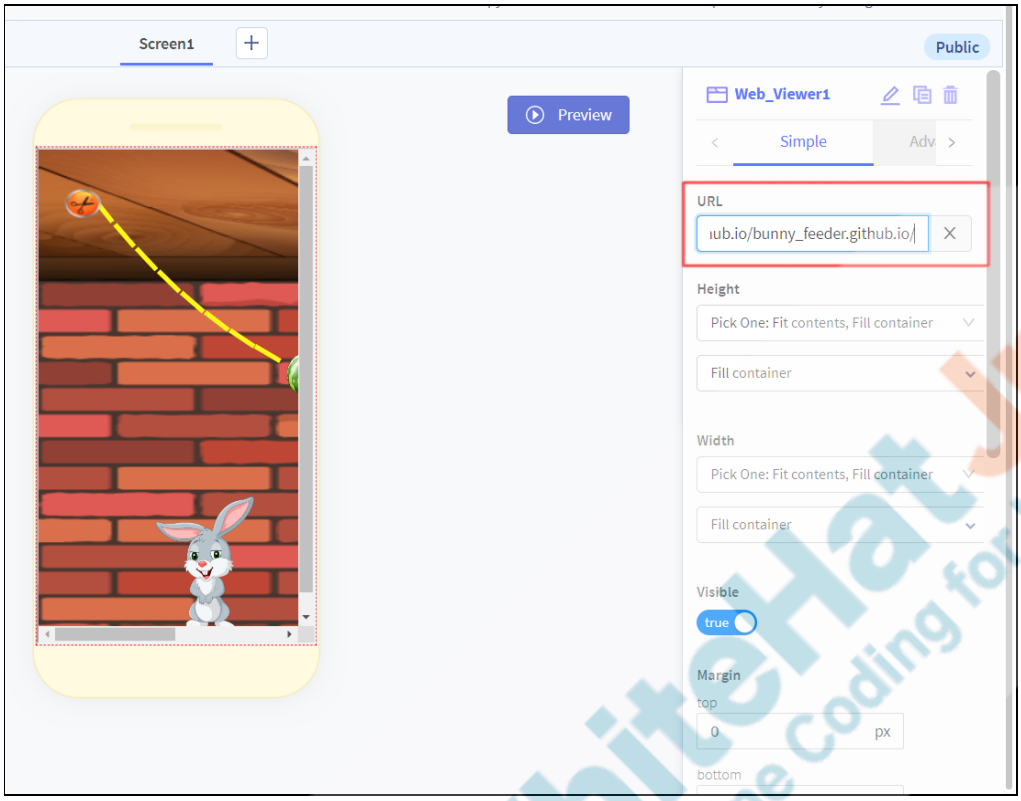




Drag and drop the **Web Viewer** component on the screen. This will open the options for the **Web_Viewer1** on the right side.



Paste the GitHub pages link in the URL section.



When the link is added, it will show the game preview as well.

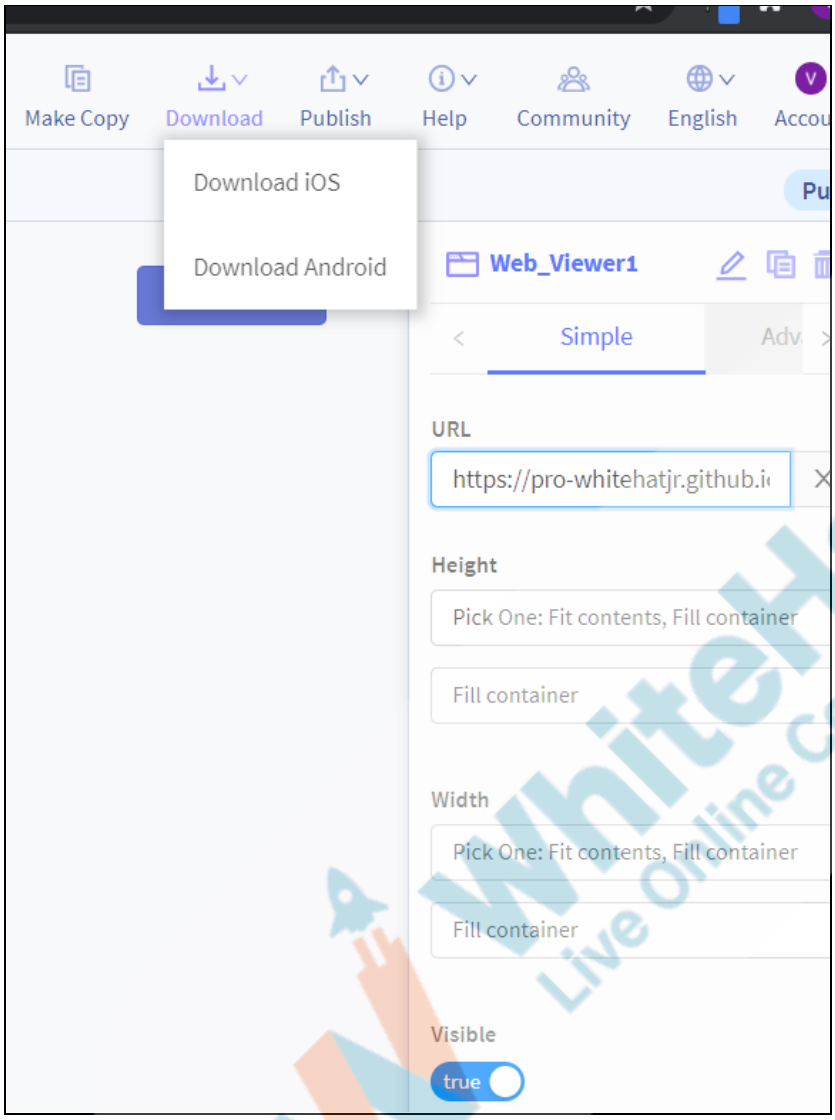
Now we just need to export the game in the required format of iOS and Android.

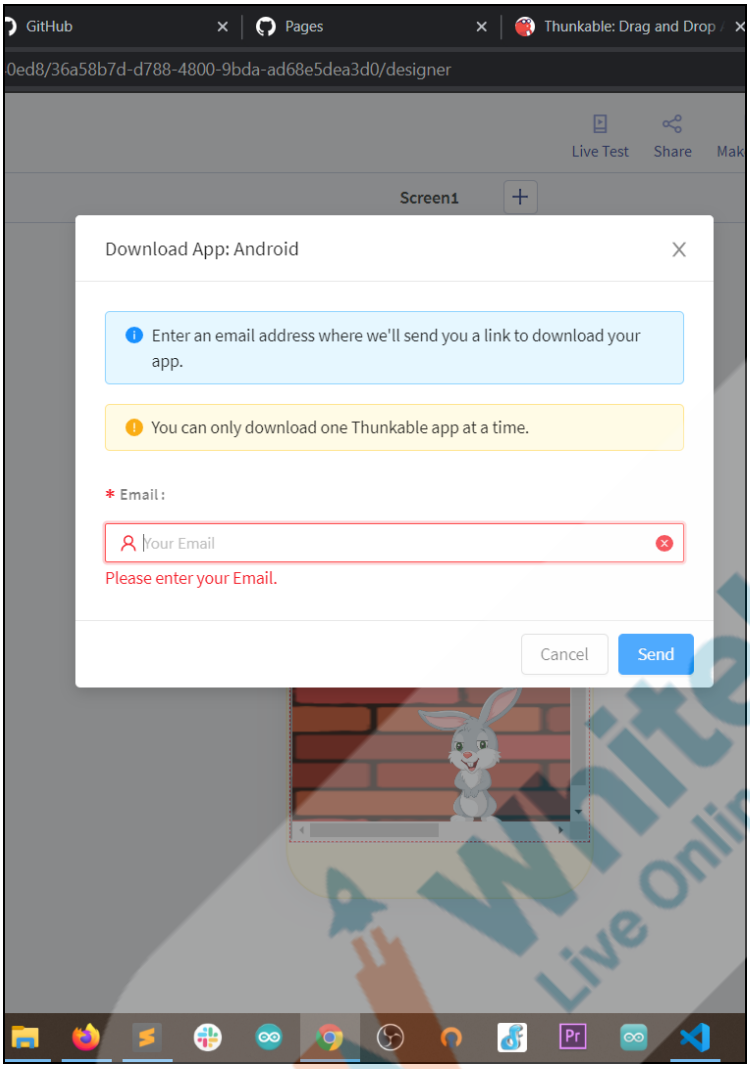
When you select the choice of device it will send the download link on your mail, from that link we can download the installable file.

Then transfer that file to the device and install it.

Now enjoy the game on the mobile.

Note: It may take some time to receive the download link from Thunkable.





I hope you enjoyed the class. Now we can discuss the project for this class.	

Teacher Guides Student to Stop Screen Share
WRAP-UP SESSION - 5 Mins
Teacher starts slideshow : from slide 17 to slide 25

Activity details	Solution/Guidelines
Run the presentation from slide 17 to slide 25. **The following are the wrap-up session deliverables:** • **Appreciate the student.** • **Revise the current class activities.** • **Discuss the quizzes.**	Discuss with the student the current class activities and the student can ask doubts related to the activities.
Quiz time - Click on in-class quiz	
Question	**Answer**

In the following piece of code, what does the first parameter of the **Rope()** represent?
 	
`rope = new Rope(8,{x:40,y:30});`
`rope2 = new Rope(7,{x:370,y:40});`
`rope3 = new Rope(4,{x:400,y:225});`
 	
A. The sections(elements) the rope will have.
B. The number of ropes the game will have.
C. The number of objects the Rope class will have.
D. The number of fruits that will be there in the game.
	A

Which of the following commands will create a constraint between the rope and the fruit? A. Fruit_con = new(rope,fruit) B. Fruit_con = new Link(rope,fruit) C. Fruit_con = Link(rope,fruit) D. Fruit = new Link(rope)	**B**
What does the following snippet of code do? `<meta name="viewport" content="user-scalable=no,initial-scale=1, maximum-scale=0.825,minimum-scale=0.8,width=device-width">` A. This line will make sure that our game will adapt to	**C**

any platform. B. This line will make sure that our game will adapt to any device. C. This line will make sure that our game will adapt to any screen size. D. This line will make sure that our game will adapt to any region.	
End the quiz panel	
Activity details	**Solution/Guidelines**
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 	
You get hats off. See you in the next class then.	Make sure you have given at least 2 Hats Off during the class for:   
*** This Project will take only 30 mins to complete. Motivate students to try and finish it immediately after the class.**	*The student engages with the teacher over the project.*

Project Overview

FEED THE BUNNY -Level Up!

Goal of the Project:

In this project, we will add another challenge to the "Feed the Bunny" game, in which you should place the bunny at the top and the fruit should be hung with two ropes. If the user clicks on the bottom left button, the fruit should swing and hit the bubble. After which the bubble along with the fruit in it should float upwards. If the fruit collides with the bunny, the bunny will eat the fruit.

Story:

One day, Jenny saw her friend Sara was playing the Feed the Bunny game. Jenny noticed the game was too easy for Sara. When she got home, she thought of creating a better challenge to the game. Can you help her with this?

Bye Bye!

Note to teachers [Only Applicable for C34]:

Next class C34 is a CHECKPOINT REVISION CLASS meant for revising concepts learned so far. Teachers should guide students to complete pending/expired projects. Please check on the dashboard if the student has pending projects less than 3, then you can continue to do the class activity.

IF STUDENT HAS > 3 PENDING PROJECT
SKIP CLASS ACTIVITY (INCLUDING VA & QUIZ) & HELP THEM COMPLETE PROJECT

ELSE
CONTINUE CLASS ACTIVITY

Activity	Description	Links
Teacher Activity 1	Template Code	https://github.com/pro-whitehatjr/C33_TA_1
Student Activity 1	Template code	https://github.com/pro-whitehatjr/C33_SA1
Complete Reference Code	Reference code of the complete Game	https://github.com/pro-whitehatjr/bunny_feeder.github.io
Hosted game link	Hosted link	https://pro-whitehatjr.github.io/bunny_feeder.github.io/
Thunkable project	Thunkable project link	https://x.thunkable.com/projects/60768805ca30dc0011540ed8/bc3296ee-6b4f-450a-8b62-971201e321aa/designer
Project Solution	Project Solution	https://github.com/pro-whitehatjr/C33_project_solution
Teacher Reference visual aid link	Visual aid link	https://curriculum.whitehatjr.com/Visual+Project+Asset/PRO_VD/BJFC-PRO-V3-C33-withcues.html
Teacher Reference In-class quiz	In-class quiz	https://s3-whjr-curriculum-uploads.whjr.online/1ee06cc1-b6e7-4321-84c6-69fa70940901.pdf
Project Solution	Feed the Bunny- Level Up	https://github.com/pro-whitehatjr/PRO-C33_Project_Solution_