

Topic	SPRITESHEET AND ANIMATION	
Class Description	Students will learn to use animated sprites. Students will learn to create their own animated sprites.	
Class	C27	
Class time	55 mins	
Goal	<ul style="list-style-type: none"> • Create an animated sprite from a GIF. • Create SpriteSheet for boat animation. • Create SpriteSheet for broken boat animation. 	
Resources Required	<ul style="list-style-type: none"> • Teacher Resources <ul style="list-style-type: none"> ○ Laptop with internet connectivity ○ Visual Studio Code ○ Earphones with mic ○ Notebook and pen • Student Resources <ul style="list-style-type: none"> ○ Laptop with internet connectivity ○ Visual Studio Code ○ Earphones with mic ○ Notebook and pen 	
Class structure	WARM-UP Teacher-led Activity Student-led Activity WRAP-UP	5 Mins 15 Mins 30 Mins 5 Mins
Credits	The class uses Piskel, a free online editor for animated sprites, which is Licensed under the Apache License, Version 2.0.	
WARM-UP SESSION - 5 mins		



Teacher starts slideshow from slides 1 to 10

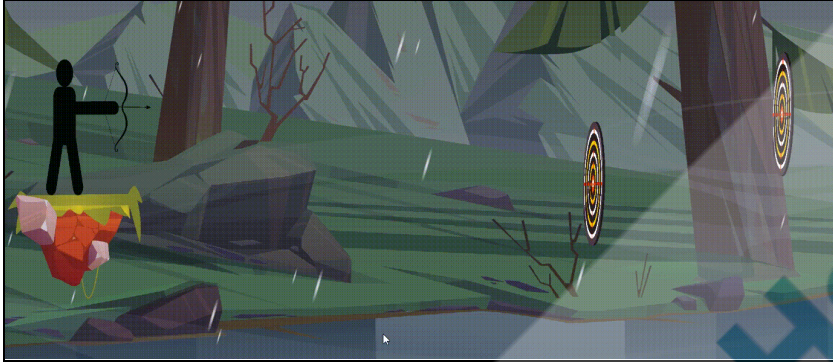
Refer to speaker notes and follow the instructions on each slide.

Activity details	Solution/Guidelines
<p><i>Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today?</i></p> <p>Run the presentation from slide 1 to slide 3.</p> <p>Following are the WARM-UP session deliverables:</p> <ul style="list-style-type: none"> • Greet the student. • Revision of previous class activity. • Quizzes 	<p>ESR: Hi, thanks. Yes, I am excited about it.</p> <p>Click on the slide show tab and present the slides.</p>
QnA Session	
Question	Answer
<p>Choose the correct option to make the board body static.</p> <p>A. <pre>var options = { isStatic: false }</pre></p> <p>B. <pre>var options = { isStatic: true }</pre></p> <p>C. <pre>var options = { Static: true }</pre></p>	<p>B</p>

```
var options = {
  isStatic= true
}
```

D.

Select the correct option to remove the **arrow**.



```
/* remove(index) {
  this.isRemoved = false;
  Matter.World.remove(world, this.body);
  delete playerArrows[index];
} */
```

A.

```
/* remove(index) {
  this.isRemoved = true;
  Matter.World.remove(world, this.body);
  delete playerArrows[index];
} */
```

B.


```
/* remove(index) {
  this.isRemoved = true;
  Matter.World.remove(this.body);
  delete playerArrows[index];
} */
```

C.

```
/* remove(index) {
  this.isRemoved = true;
  Matter.World.remove(world, this.body);
  delete playerArrows[];
} */
```

D.

B

Continue the WARM-UP session	
Activity details	Solution/Guidelines
<p>Run the presentation from slide 4 to slide 10 to set the problem statement.</p> <p>Following are the WARM-UP session deliverables:</p> <ul style="list-style-type: none"> Appreciate the student on his performance in the quizzes. Explain the relevance of animation. 	<p>Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.</p>
<p>Teacher ends slideshow </p>	
TEACHER-LED ACTIVITY - 15 mins	
Teacher Initiates Screen Share	
<p>CHALLENGE</p> <ul style="list-style-type: none"> Create a SpriteSheet for a moving boat. Add moving boat animation to the boat. 	
Teacher Action	Student Action
<p><i>Teacher clones the code from the Teacher Activity 1</i></p>	<p><i>Student observes.</i></p>
<p>So today we'll be adding the animations to the boats to give it a feel as if the boat is moving on the water. We already have some of the images loaded in the assets folder.</p> <p><i>Teacher opens the assets folder and shows the boat assets.</i></p> <p><i>Teacher opens the boat.json file.</i></p>	

If you see here we have the main object and inside it, we have a **frames** key and its value is an array that contains multiple objects with different positions of the images.

Looking at one example of the following we can see that an object has multiple keys such as **position**, **rotated**, **trimmed**, **spriteSourceSize** and **sourceSize**.

The most important one is the position as it contains the position of the images.

```
assets > boat > {} boat.json > ...
1  {
2    "frames": [
3      {
4        "position": { "x": 0, "y": 0, "w": 500, "h": 500 },
5        "rotated": false,
6        "trimmed": false,
7        "spriteSourceSize": { "x": 0, "y": 0, "w": 500, "h": 500 },
8        "sourceSize": { "w": 500, "h": 500 }
9      },
10     {
11       "position": { "x": 500, "y": 0, "w": 500, "h": 500 },
12       "rotated": false,
13       "trimmed": false,
14       "spriteSourceSize": { "x": 0, "y": 0, "w": 500, "h": 500 },
15       "sourceSize": { "w": 500, "h": 500 }
16     },
17     {
18       "position": { "x": 0, "y": 500, "w": 500, "h": 500 },
19       "rotated": false,
20       "trimmed": false,
21       "spriteSourceSize": { "x": 0, "y": 0, "w": 500, "h": 500 },
22       "sourceSize": { "w": 500, "h": 500 }
23     },
24     {
25       "position": { "x": 500, "y": 500, "w": 500, "h": 500 },
26       "rotated": false,
27       "trimmed": false,
28       "spriteSourceSize": { "x": 0, "y": 0, "w": 500, "h": 500 },
29       "sourceSize": { "w": 500, "h": 500 }
30     }
31   ]
32 }
```

Teacher opens the boat.png file.

In this file you can see that we have 4 different frames of the boat, each has a different action.



Alright, now we have the required data. Let's set this animation of the boat.

We'll first create a **boatAnimation** array. We'll also declare two variables: **boatSpritedata** and **boatSpriteSheet**.

boatSpritedata will contain data from JSON.

boatSpritesheet will contain the images.

Student observes and learns.

Declare variables in sketch.js file

```
var boatAnimation = [];
var boatSpritedata, boatSpritesheet;
```

Load Images and JSON in preload() function

```
function preload() {
  backgroundImg = loadImage("./assets/background.gif");
  towerImage = loadImage("./assets/tower.png");
  boatSpritedata = loadJSON("assets/boat/boat.json");
  boatSpritesheet = loadImage("assets/boat/boat.png");
}
```

Now let's get the data from the JSON and the image files.

Now we'll declare the **boatFrames** variable. This variable will contain the frame data in it.

create a boatFrames variable which contains the frames data in setup() of Sketch.js

```
var boatFrames = boatSpritedata.frames;
```

Using the for loop, we'll loop on the length of the boatFrames.

Inside this loop, we'll get the position of each frame.

From the SpriteSheet we'll get the image with respective position and push this image to the boatAnimation array.

Write code to iterate through an array of **boatFrames** in **setup()** function in sketch.js file.

variable **pos** is used to get the position of each frame from **boatFrames**.

variable **img** is used to get the image from the **boatSpritesheet** which matches the position that we got in the **pos** variable and then pushing this image in the **boatAnimation** array.

```
var boatFrames = boatSpritedata.frames;
for (var i = 0; i < boatFrames.length; i++) {
  var pos = boatFrames[i].position;
  var img = boatSpritesheet.get(pos.x, pos.y, pos.w, pos.h);
  boatAnimation.push(img);
}
```

Now we just need to add the animation to the boats that we are creating. We'll also need to modify the **Boat.js** file to add the animation.

So in the **constructor()** of the **Boat** class we'll pass one more variable called **boatAnimation**.

Writing code in the Boat.js file in constructor (); Adding the boatAnimation as a parameter to the constructor function and then using this.animation adding it to the boat.


```
class Boat {  
  constructor(x, y, width, height, boatPos, boatAnimation) {  
    var options = {  
      restitution: 0.8,  
      friction: 1.0,  
      density: 1.0,  
      label: "boat"  
    };  
    this.animation = boatAnimation;  
    this.speed = 0.05;  
    this.body = Bodies.rectangle(x, y, width, height, options);  
    this.width = width;  
    this.height = height;  
  
    this.boatPosition = boatPos;  
    this.image = loadImage("assets/boat.png");  
    World.add(world, this.body);  
  }  
}
```

We need the animation to move at a certain speed.
So we'll write an **animate()** function which will help us to set the speed of the animation. This speed will determine how fast every frame in our animation will move.

The speed will keep on increasing as the game progresses and will be divided by the length of the animation so that every frame gets a certain amount of screen time.
We will increment the speed by 0.05.

setting the speed of the animation in Boat.js file

```
animate() {  
  this.speed += 0.05;  
}
```

In the **display()** function we'll create a variable **index**. We can use this variable to traverse through a set of animations.

The index is calculated using the **floor()** method on **this.speed** & **this.animation.length**. Inside **image()** we'll pass the animation.

*Creating a variable called an **index** which we'll be using to traverse through the set of animations.*

*We are calculating the index by dividing the speed of animation by animation length to get the smallest number inside the **image()** function using this animation.*

Writing code in the Boat.js file in the display() function.

```
display() {
  var pos = this.body.position;
  var index = floor(this.speed % this.animation.length);

  push();
  translate(pos.x, pos.y);
  imageMode(CENTER);
  image(this.animation[index], 0, this.boatPosition, this.width, this.height);
  pop();
}
```

Similarly, we need to pass the **boatAnimation** while we are creating the boat objects and also call the **animate()** function.

Writing code in sketch.js file in showBoat function.

```
function showBoats() {
  if (boats.length > 0) {
    if (
      boats[boats.length - 1] === undefined ||
      boats[boats.length - 1].body.position.x < width - 300
    ) {
      var positions = [-40, -60, -70, -20];
      var position = random(positions);
      var boat = new Boat(
        width,
        height - 100,
        170,
        170,
        position,
        boatAnimation
      );

      boats.push(boat);
    }


    for (var i = 0; i < boats.length; i++) {
      if (boats[i]) {
        Matter.Body.setVelocity(boats[i].body, {
          x: -0.9,
          y: 0
        });

        boats[i].display();
        boats[i].animate();
      }
    }
  } else {
    var boat = new Boat(width, height - 60, 170, 170, -60, boatAnimation);
    boats.push(boat);
  }
}
```

We are passing the boat animation to the new boats that we are creating and also calling the **animate()** function for the boats to animate.

Teacher runs the code to show the output.

Now we can see the boats moving.

Don't you think we should also have animation when the boat is hit by the cannonball? Would you like to try it? Let's get you started.		ESR: Yes.
Teacher Stops Screen Share		
Teacher starts slideshow  :Slide 11 to Slide 12		
Run the presentation slide to set the student activity context. <ul style="list-style-type: none"> Broken Boat Animation. 		
STUDENT-LED ACTIVITY - 30 mins		
	Now it's your turn. Please share your screen with me.	
<ul style="list-style-type: none"> Ask the student to press the ESC key to come back to the panel. Guide the student to start Screen Share. The teacher gets into Fullscreen. 		
<ul style="list-style-type: none"> Add the broken boat animation to the boat when it collides with the cannonball. 		
Teacher Action		Student Action
<i>Teacher guides the student to download the code from student Activity 1</i>		<i>The student downloads the code from Student Activity 1.</i>

```
boat > {} brokenBoat.json > ...
{
  "frames": [
    {
      "position": { "x": 0, "y": 0, "w": 500, "h": 500 },
      "rotated": false,
      "trimmed": false,
      "spriteSourceSize": { "x": 0, "y": 0, "w": 500, "h": 500 },
      "sourceSize": { "w": 500, "h": 500 }
    },
    {
      "position": { "x": 500, "y": 0, "w": 500, "h": 500 },
      "rotated": false,
      "trimmed": false,
      "spriteSourceSize": { "x": 0, "y": 0, "w": 500, "h": 500 },
      "sourceSize": { "w": 500, "h": 500 }
    },
    {
      "position": { "x": 1000, "y": 0, "w": 500, "h": 500 },
      "rotated": false,
      "trimmed": false,
      "spriteSourceSize": { "x": 0, "y": 0, "w": 500, "h": 500 },
      "sourceSize": { "w": 500, "h": 500 }
    },
    {
      "position": { "x": 0, "y": 500, "w": 500, "h": 500 },
      "rotated": false,
      "trimmed": false,
      "spriteSourceSize": { "x": 0, "y": 0, "w": 500, "h": 500 },
      "sourceSize": { "w": 500, "h": 500 }
    },
    {
      "position": { "x": 500, "y": 500, "w": 500, "h": 500 },
      "rotated": false,
      "trimmed": false,
      "spriteSourceSize": { "x": 0, "y": 0, "w": 500, "h": 500 },
      "sourceSize": { "w": 500, "h": 500 }
    }
  ]
}
```

Now we just need to add the sprites data. We just need to set the animation to the boat.

First we'll create an empty array called **brokenBoatAnimation** and declare 2 variables **brokenBoatSpritedata**, **brokenBoatSpriteSheet**.

And load the data to it.

Creating an empty **brokenBoatAnimation** array and declaring the variables **brokenBoatSpritedata** and **brokenBoatSpritesheet** to store the frames and images of

the broken boat.

Declare a variable to load the JSON and images of an animation.

```
var brokenBoatAnimation = [];
var brokenBoatSpritedata, brokenBoatSpritesheet;
```

in the preload() function

```
brokenBoatSpritedata = loadJSON("assets/boat/broken_boat.json");
brokenBoatSpritesheet = loadImage("assets/boat/broken_boat.png");
```

Using the for loop we'll loop on the length of the **boatFrames**.

Inside this loop we'll get the position of each frame.
 From the SpriteSheet we'll get the image with its respective position and push this image to the boatAnimation array.

<The teacher helps the student with the code.>

<The student codes to use the for loop.>

Student codes to get the positions of the frames.

Student codes to get the images of the given position and push in the brokenBoatAnimation array.>

```
var brokenBoatFrames = brokenBoatSpritedata.frames;
for (var i = 0; i < brokenBoatFrames.length; i++) {
  var pos = brokenBoatFrames[i].position;
  var img = brokenBoatSpritesheet.get(pos.x, pos.y, pos.w, pos.h);
  brokenBoatAnimation.push(img);
}
```

Using for loop on the length of the boatFrames .

In variable pos we are getting the position of each frame from boatFrames.

In variable img we are getting the image from the boatSpritesheet which matches the position that we got in the pos variable and then pushing this, image in the boatAnimation array.

We had already written a function to remove the boats from the game.

We want our boat to fade away slowly and we want to play our animation before the boat disappears.

So for this we can use the function called **setTimeout()**. This function executes the code inside it after a certain interval of time.

*<The student codes to use the **setTimeout** function.>*

<The teacher helps the student with the code.>

```
remove(index) {  
  this.animation = brokenBoatAnimation;  
  this.speed = 0.05;  
  this.width = 300;  
  this.height = 300;  
  this.isBroken = true;  
  setTimeout(() => {  
    Matter.World.remove(world, boats[index].body);  
    delete boats[index];  
  }, 2000);  
}
```

Using the **remove()** function we are removing the boat of the given index from the world. As we want to remove the boat slowly. We have used the **setTimeout** function which calls the code inside it after the set intervals of time. We have set the time limit to 2 seconds.

Inside the **setTimeout()** we have used the **Matter.World.remove()** function to remove the boat from the world and **delete boats[index]** to remove it from the array.

Teacher asks student to run the code to check the output

Student runs the code to check the output.



Similarly, the code for water splash when the cannonball hits the ground is already provided in the code. So we can see the splash animation when the cannonball hits the water.



Awesome job. Now we can see the broken boat animation when the boat is hit by a cannonball.

What should we do next?

Great ideas.

Before we end the class, do you remember how we added a **.gif** background to our pirate invasion?

ESR: Varied.

ESR: Varied.

*Teacher can ask student to check **preload()** function and **index.html** to remind how **.gif** was added*

In today's project, you will add a similar animated background in EPIC Archery.

Teacher Guides Student to Stop Screen Share

WRAP-UP SESSION - 5 Mins

Teacher starts slideshow



from slide13 to slide 24

Activity details

Solution/Guidelines

Run the presentation from slide 13 to slide 18.

Following are the **WARM-UP** session deliverables:

- **Appreciate the student.**
- **Revise the current class activities.**
- **Discuss the quizzes.**

Discuss with the student the current class activities and Student will ask doubts related to the activities.

Quiz time - Click on in-class quiz

Question

Answer

How to give the effect of movement using static images?

- A. using SpriteSheet
- B. using GIF images
- C. using videos
- D. using frame concept



A



What does the following snippet of code do?

```
brokenBoatSpritedata = loadJSON("assets/boat/broken_boat.json")
```

- A. loading the image in a variable
- B. loading the JSON data in a variable

B

C. loading the animation in a variable D. creating the JSON data for animation.		
Which function is used to remove a particular element from the array? A. slice() B. pop() C. splice() D. push()		C
End the quiz panel		
Activity details		Solution/Guidelines
Run the presentation from slide 19 to slide 24. Following are the WARM-UP session deliverables: <ul style="list-style-type: none"> ● Explain the facts and trivia ● Next class challenge ● Project for the day ● Additional Activity 		Guide the student to develop the project and share with us
FEEDBACK <ul style="list-style-type: none"> ● Appreciate the student for their efforts in the class. ● Ask the student to make notes for the reflection journal along with the code they wrote in today's class. 		
Step 4: Wrap-Up (5 mins)	You get a hats off. In the next class, we'll add sounds to the game and write conditions for game over and score.	Make sure you have given at least 2 Hats Off during the class for: <div> Creatively Solved Activities  +10 </div> <div> Great Question  +10 </div>

		<div> <div>Strong Concentration</div> <div>  </div> </div>
<p>* This Project will take only 30 mins to complete. Motivate students to try and finish it immediately after the class.</p> <p>Project Overview</p> <p>EPIC ARCHERY STAGE 6</p> <p>Goal of the Project:</p> <p>In Class 27, you saw how to use animated sprites and create your own animated sprites to make the game more fun and attractive. In this project, you will add a trajectory path for the arrow to make it more fun and attractive.</p> <p>*This is a continuation of Project 22, 23, 24, 25 & 26. Make sure to complete that work before attempting this one.</p> <p>Story:</p> <p>Archery is one of the oldest arts which is still practiced. After reading the information about Archery in a book, your friend Georgie wants to play Archery. To give him a virtual experience, you want to use your coding expertise and physics engine concepts to create an Archery game for him.</p> <p>Add a trajectory path for the arrow to make it more fun and attractive.</p> <p>I am very excited to see your project solution and I know you will do really well.</p> <p>Bye Bye!</p>		<p>Note: You can assign the project to the student in class itself by clicking on the Assign Project button which is available under the projects tab.</p> <p><i>Students engage with the teacher over the project.</i></p>
<div> <div>Teacher ends slideshow</div> <div>  </div> </div>		

Teacher Clicks

✕ End Class

ADDITIONAL ACTIVITIES

Additional Activities to create the Spritesheet from the gif.

Let's see how to create a SpriteSheet.

<The teacher opens the link from [Teacher Activity 2](#).>

We will be using PISKEL APP. This is only supported by Chrome, Firefox and Edge browsers.

Download the GIF from the [Teacher Activity 3](#).

This is the site that we are going to use to create the SpriteSheet. **Piskel** is an online, in-browser application for easily creating static pixel art images and animated pixel art GIFs.

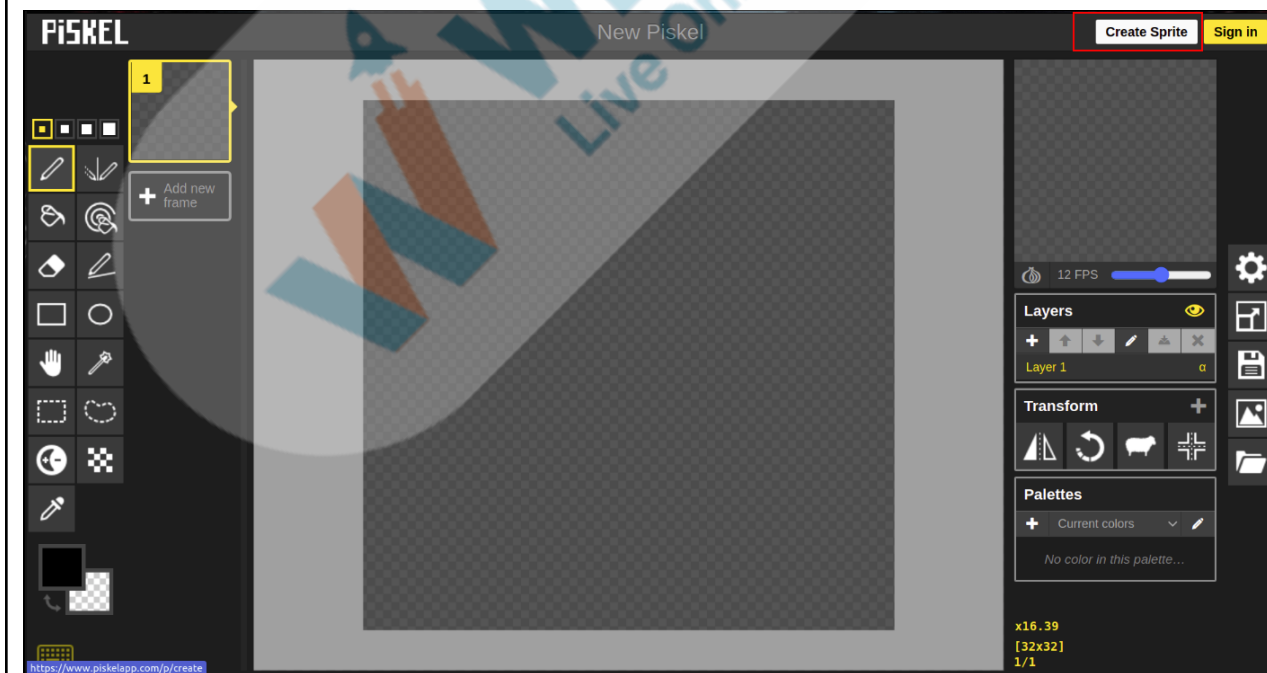
The program is easy to **use** and offers many features, such as the ability to live preview your animations and the ability to save and export your images and GIFs to easily distribute and share.

Student observes.

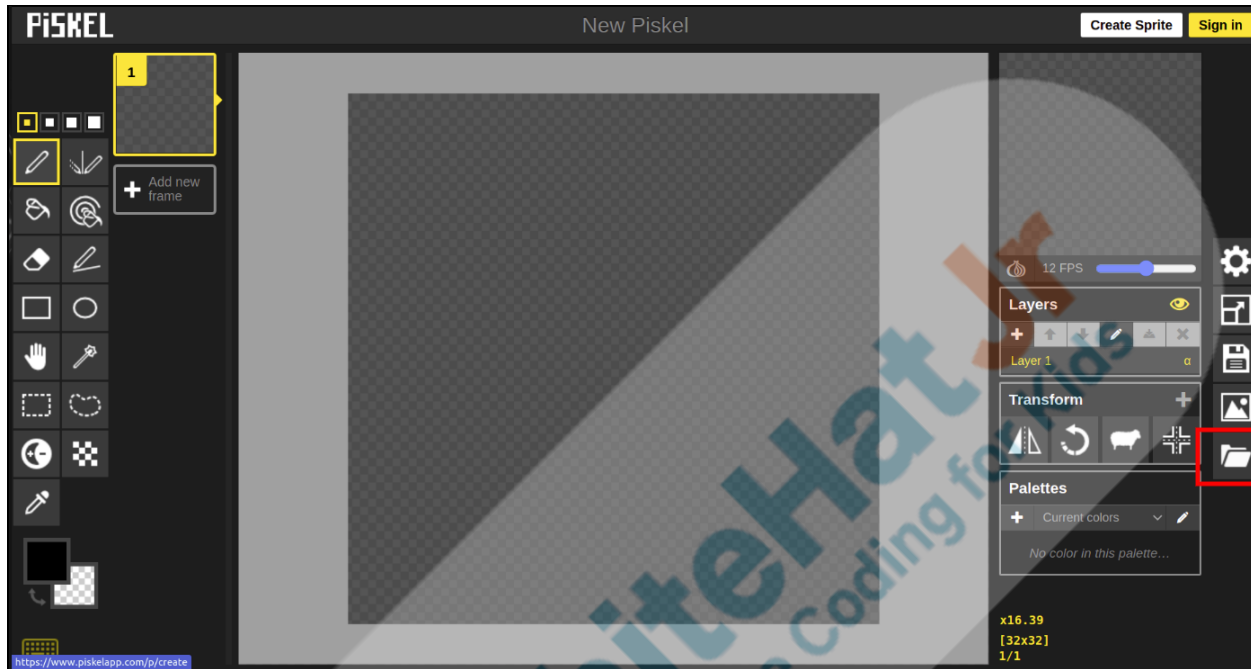


Step 1) Click on Create Sprite option to open a new tab to create a SpriteSheet.

Student observes and learns.

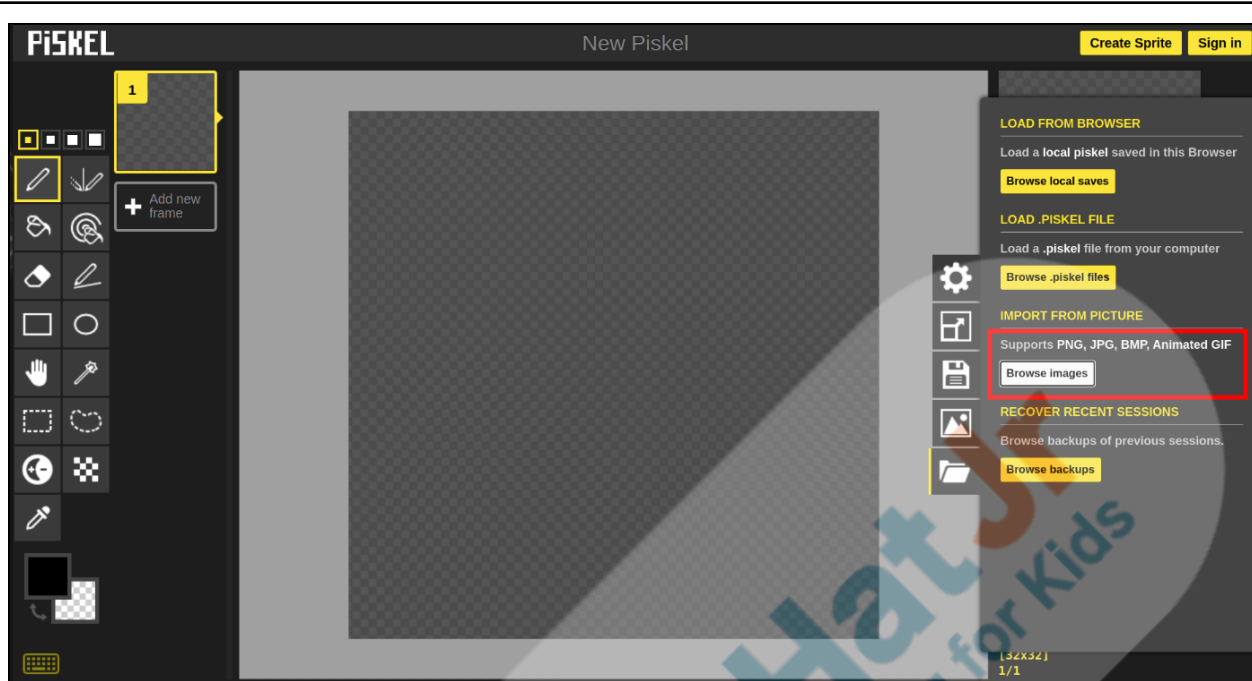


Step 2) Click on the import button to import the GIF you want to create the sprite sheet.



Step 3) Click on Browse Image and select the GIF file.
In our case, we'll select the sailing ship GIF.

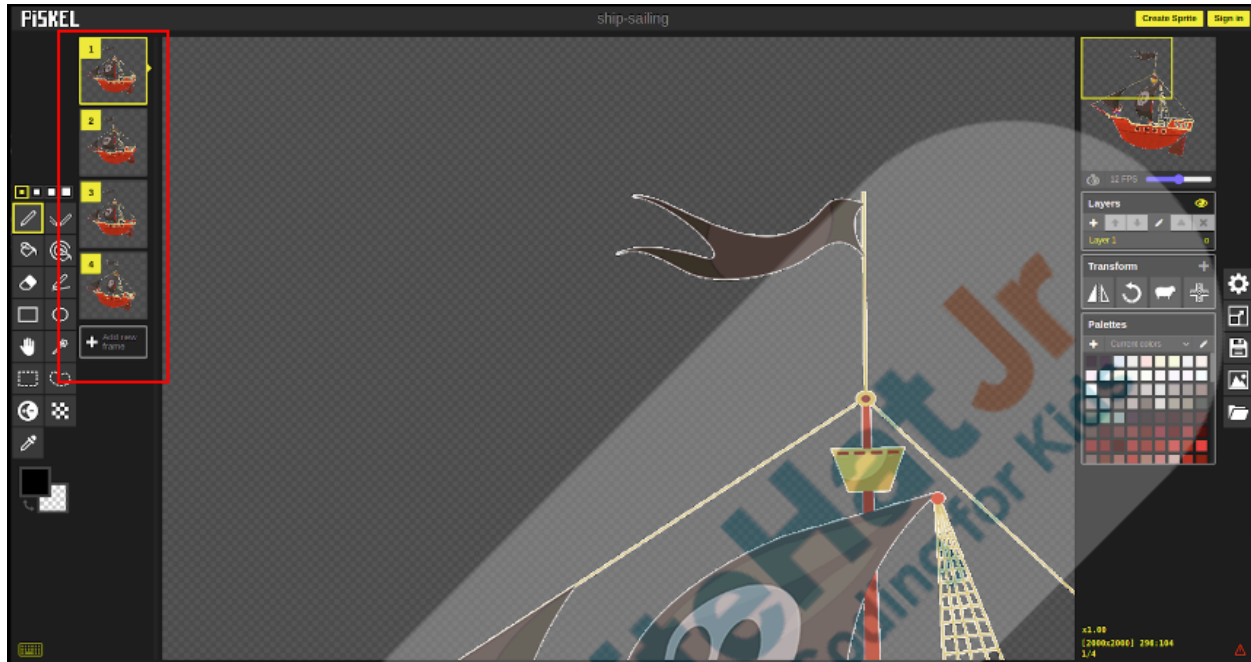
Student observes and learns.



Step 4) Import the GIF image which we downloaded as an image.

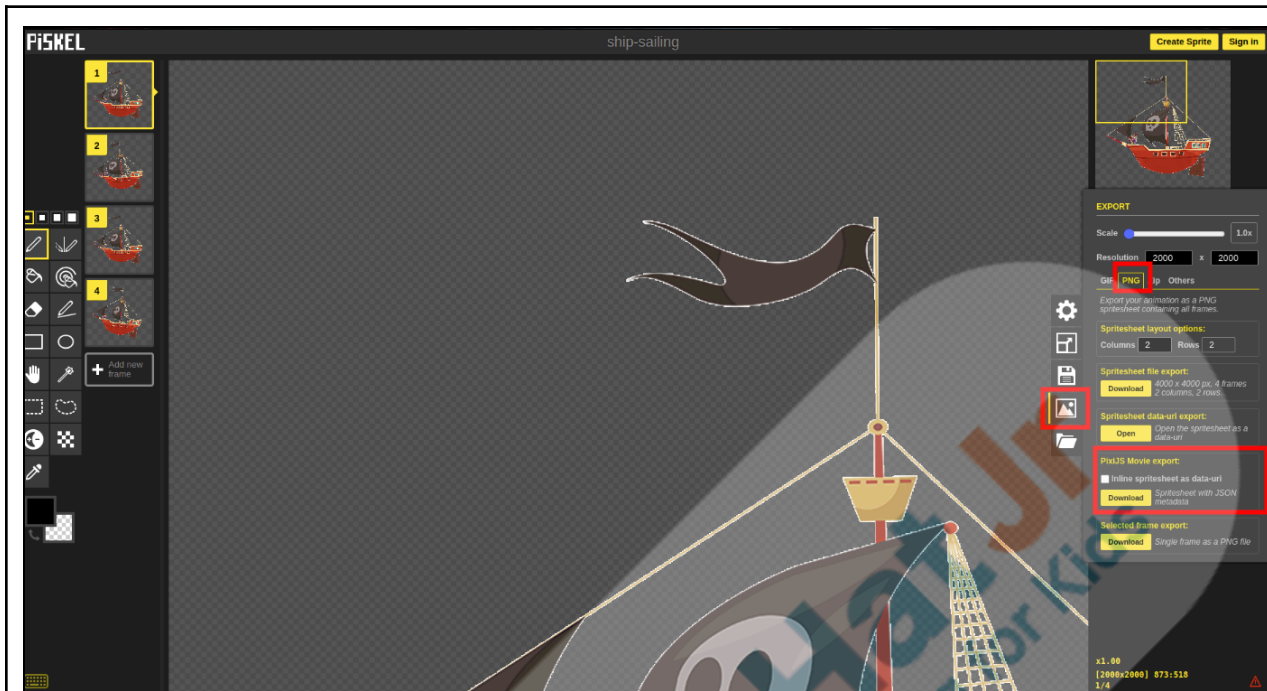
Student observes and learns.

You'll see that the GIF is split into 4 parts.



Now it's time to export the image.
Step 5) Click on the image button, then click on **PNG**, then select the PIXi Movie js export option.

Student observes and learns.





Student observes and learns.

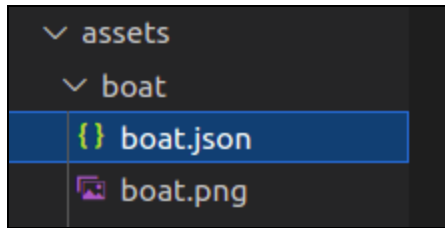
You'll see that a zip file gets downloaded.

This zip file contains a JSON file and a PNG file.

Now we go back to the VS Code and create a new folder called as boat inside the assets folder.

And extract the zip file in it.

Name	Size	Type
 ship-sailing.json	880 bytes	JSON docu..
 ship-sailing.png	1.8 MB	PNG image



If you see the PNG file, you'll see that it has 4 images of different actions of the boat.

Student observes and learns.

If you see the JSON file, you'll see that it has frame data along with the image name and position.



```
{
  "frames": {
    "ship-sailing0.png": {
      "frame": {
        "x": 0,
        "y": 0,
        "w": 2000,
        "h": 2000,
        "rotated": false,
        "trimmed": false,
        "spriteSourceSize": {
          "x": 0,
          "y": 0,
          "w": 2000,
          "h": 2000
        },
        "sourceSize": {
          "w": 2000,
          "h": 2000
        }
      },
      "ship-sa
```

```
iling1.png":{"frame":{"x":2000,"y":0,"w":2000,"h":2000},"rotated":false,"trimmed":false,"spriteSourceSize":{"x":0,"y":0,"w":2000,"h":2000},"sourceSize":{"w":2000,"h":2000}},"ship-sailing2.png":{"frame":{"x":0,"y":2000,"w":2000,"h":2000},"rotated":false,"trimmed":false,"spriteSourceSize":{"x":0,"y":0,"w":2000,"h":2000},"sourceSize":{"w":2000,"h":2000}},"ship-sailing3.png":{"frame":{"x":2000,"y":2000,"w":2000,"h":2000},"rotated":false,"trimmed":false,"spriteSourceSize":{"x":0,"y":0,"w":2000,"h":2000},"sourceSize":{"w":2000,"h":2000}}},"meta":{"app":"https://github.com/piskelapp/piskel/","version":"1.0","image":"ship-sailing.png","format":"RGBA8888","size":{"w":4000,"h":4000}}}
```

This JSON has some extra data which is of no use to us so we'll clean the JSON.

We just need the name of the image and the positions of the frames.

As we see now, the structure of data is that there is a main object denoted by {}, inside this object that is a key named **frames** and has another object as its value.

So we'll change the data structure as we'll have the main object and inside it we'll have frames as **key** and an array as its value. This array will contain all the data.

Student observes and learns.

```
assets > boat > {} boat.json > ...
1  {
2    "frames": [
3      {
4        "position": { "x": 0, "y": 0, "w": 2000, "h": 2000 },
5        "name": "ship-sailing0.png"
6      },
7      {
8        "position": { "x": 2000, "y": 0, "w": 2000, "h": 2000 },
9        "name": "ship-sailing1.png"
10     },
11     {
12       "position": { "x": 0, "y": 2000, "w": 2000, "h": 2000 },
13       "name": "ship-sailing2.png"
14     },
15     {
16       "position": { "x": 2000, "y": 2000, "w": 2000, "h": 2000 },
17       "name": "ship-sailing3.png"
18     }
19   ]
20 }
```

Now we have the images ready. Similarly we can create the images for the broken boat and use them in the game.

To create the spritesheet for broken boat

1. Let's start by creating the sprite sheet for the broken boat.
2. First download the image from [Student Activity 3](#).
3. Open the image from the Piskel app from the [Student Activity 2](#).
4. Click on the Create Sprite button.
5. Click on the import button and upload the broken boat GIF.
6. Click on export button, click on PNG format and select the **Pixi Js** movie.
7. Your zip will be downloaded.

Activity	Activity Name	Links
Teacher Activity 1	Previous class code	https://github.com/pro-whitehatjr/PRO-C26-Reference
Teacher Activity 2	Piskel link	https://www.piskelapp.com/p/create/sprite
Teacher Activity 3	Sailing ship GIF	https://github.com/whitehatjr/PiratesInvasionStage-4.5/blob/main/assets/ship-sailing.GIF
Teacher Activity 4	Reference Link	https://github.com/pro-whitehatjr/PRO-C27-Reference
Student Activity 1	Boilerplate code	https://github.com/pro-whitehatjr/PRO-C27-SA
Student Activity 2	Piskel link	https://www.piskelapp.com/p/create/sprite
Student Activity 3	Broken ship GIF	https://github.com/whitehatjr/PiratesInvasionStage-4.5/blob/main/assets/broken-ship-01.GIF
Teacher Reference visual aid link	Visual aid link	https://s3-whjr-curriculum-uploads.whjr.online/b65c9024-7398-4209-9796-464d13954ce3.html
Teacher Reference In-class quiz	In-class quiz	https://s3-whjr-curriculum-uploads.whjr.online/b9b1d029-9368-4b71-bafa-89076eed8a80.pdf
Project Solution	Epic Archery Stage-6	https://github.com/pro-whitehatjr/V3_Project_Solution_C27