Topic	A-Frame DOM & JavaScript EVENT LISTENERS	
Class Description	**Students learn about the A-Frame DOM structure and JavaScript event listeners. Students will learn how entities can be controlled by triggering events on the web page. Students will write a program in A-Frame where the entity will behave based on the event triggered.**	
Class	**C152**	
Class time	**45 mins**	
Goal	• Learn about the A-Frame DOM structure. • Learn about the JavaScript event listener. • Control A-Frame entities based on the events fired on the entity.	

Resources Required	• Teacher Resources:
	  ○ Visual Studio Code Editor
	  ○ laptop with internet connectivity
	  ○ earphones with mic
	  ○ notebook and pen
	• Student Resources:
	  ○ Visual Studio Code Editor
	  ○ laptop with internet connectivity
	  ○ earphones with mic
	  ○ notebook and pen	

Class structure	**Warm-Up** **Teacher-led Activity** **Student-led Activity** **Wrap-Up**	**05 mins** **15 mins** **20 mins** **05 mins**

CONTEXT

- **Understand A-Frame DOM structure.**
- **Learn about JavaScript Event Listeners.**
- **Control A-Frame Entity by triggering the event listener on the window.**

Teacher starts slideshow  **from slides 1 to 13**

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

Activity details	Solution/Guidelines
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** **Following are the WARM-UP session deliverables:** • **Greet the student.** • **Revision of previous class activities.** • **Quizzes**	**ESR**: Hi, thanks, Yes I am excited about it! Click on the slide show tab and present the slides

Q&A Session

Question	Answer
Which of the following is used to get the current values of the position attribute? A. this.el.getAttribute() B. this.el.setAttribute() C. setattribute() D. getattribute	**A**
Which of the following is used to update the value of the position attribute? A. this.el.getAttribute() B. this.el.setAttribute() C. setattribute() D. getattribute	**B**

<table>
<tr><th colspan="3">Continue the WARM-UP session</th></tr>
<tr><th colspan="2">Activity details</th><th>Solution/Guidelines</th></tr>
<tr><td colspan="2">Run the presentation from slide 4 to slide 13 to set the problem statement.

Following are the WARM-UP session deliverables:
• Appreciate the student.
• Understand A-Frame DOM structure.
• Learn about JavaScript Event Listener</td><td>Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.</td></tr>
<tr><td colspan="3">Teacher ends slideshow</td></tr>
<tr><th>Class Steps</th><th>Teacher Action</th><th>Student Action</th></tr>
<tr><td>Step 1:
Warm-Up
(5 mins)</td><td>Hi <Student Name>!
How are you?

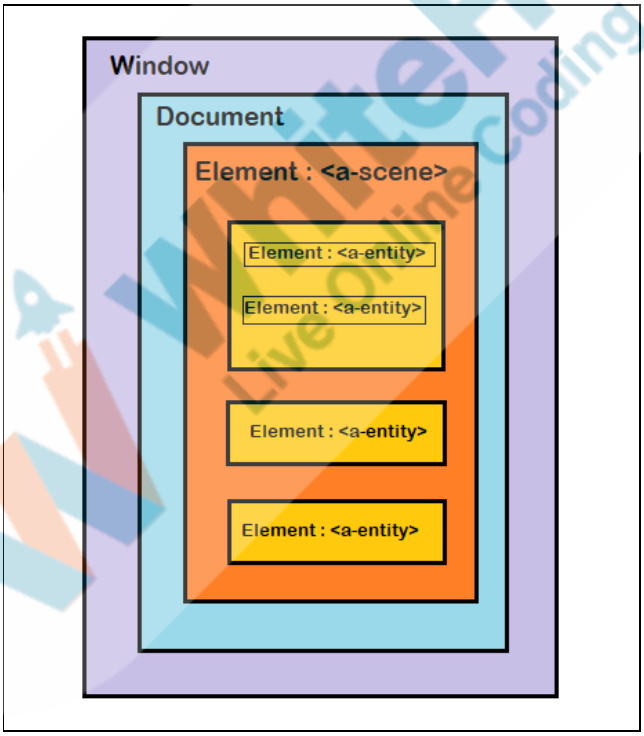
That’s great!</td><td>ESR: I am good!</td></tr>
<tr><td></td><td>Do you remember what we learned in the last class?

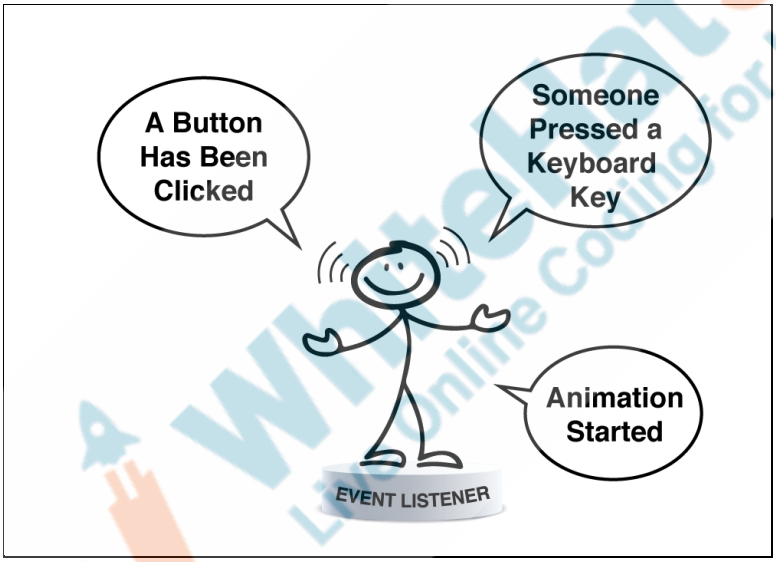
Great!

Components save us from re-writing the logic again and again.
Writing components also helps us to structure the code.</td><td>ESR:
We learned about the A-Frame custom components to move a box from one position to another.</td></tr>
</table>

	Any ideas on what we will be learning in today's class? Today we will be learning about the A-Frame DOM structure and JavaScript event listener. We will be discussing that in a while.	**ESR**: Varied.
	But before we can understand what that means, can you tell me what's been missing out in the virtual environment scenes we have been creating till now? In all the scenes that we have created till now, the elements of the scene cannot be controlled by us! We, as a programmer, should be able to control what's going inside a computer, right? Today we'll see how we can control entities in A-Frame based on some events created on the web page.	**ESR**: Varied. **ESR**: Yes!
	Now, let's get started.	
TEACHER-LED ACTIVITY - 15 mins		
Teacher Initiates Screen Share		
CHALLENGE • **Create a component to move the A-Frame Entity.** • **Add a JavaScript Event Listener to move the box.**		
Step 2: Teacher-led Activity	Do you remember how we used some p5 DOM elements while making a multiplayer car racing game?	**ESR**: Yes!

(15 mins)	Do you remember what DOM stands for? Amazing! Basically, this model or design considers web pages as documents and follows a certain structure to represent each element of this document. This model connects web pages (which are displayed using HTML) with programming languages such as Javascript. This helps us to access and manipulate the element data on the web pages. Since A-Frame is just HTML, we can control the scene and its entities using JavaScript and HTML DOM. There is something called events and event listeners in JavaScript. With JavaScript events and event listeners and HTML DOM, A-Frame entities and components can be connected with one another easily.	**ESR**: Document Object Model.

	To understand what events are and how event listeners work, let's first understand the A-Frame DOM structure. In the browser window, the A-Frame HTML is the document under which the <a-scene> element is created; followed by that, one or more <a-entity> is created. ***[Teacher Activity 1 Illustration]***	
		
	And an **event is created** every time **something happens to the elements on the webpage.**	

	For example, the most common event that you have created is the click event! Every time you click with your mouse on the web page, some event listener responds to it. Similarly, pressing a key or starting an animation on the web page are events. ***[Teacher Activity 1 Illustration]***	
		
	Let's understand that with an example. Do you remember we had written a custom A-Frame component to move the box? Now let's move the box when we click on the screen.	**ESR**: Yes.

	We can use the **addEventListener()** method on the browser **window element**. And then increment the moveX value only when the mouse is clicked on the screen. The position attribute will be updated once the moveX value is incremented after the click.	

```
// Registering component in box-component.js
AFRAME.registerComponent('move-box-on-click', {
  schema: {
    moveX: { type: "number", default: 0 },
  },
  tick: function () {

    window.addEventListener("click", (e) => {

      this.data.moveX = this.data.moveX +0.005;

    });
    var pos = this.el.getAttribute("position");
    pos.x = pos.x + this.data.moveX;
    this.el.setAttribute("position", { x: pos.x, y: pos.y, z: pos.z });
  }
});
```

```
<a-entity position="0 0 0" rotation="0 0 0" move-box-on-click>

  <a-box>

  </a-box>

</a-entity>
```

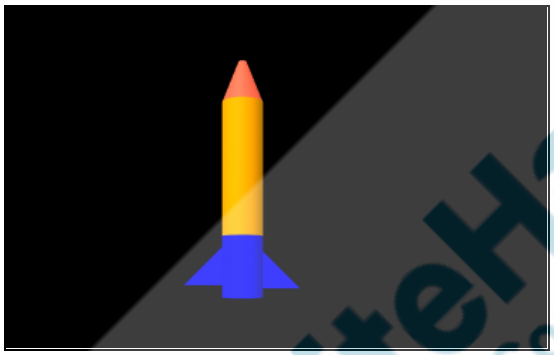
	That seemed pretty simple to do, right? Great! Now since you have learned about the event listeners, you will be simulating a rocket launch.	**ESR**: Yes!
Teacher Stops Screen Share		
Teacher starts slideshow  **from slides 14 to 20** Refer to speaker notes and follow the instructions on each slide.		
	Now it's your turn. Please share your screen with me.	
Teacher ends slideshow 		
STUDENT-LED ACTIVITY - 20 mins		
• **Ask the student to press the ESC key to come back to the panel.** • **Guide the student to start a screen share.** • **The teacher gets into fullscreen.**		
ACTIVITY • **Learn about the A-Frame DOM.** • **Learn about the JavaScript Event Listener.** • **Use the click event on the window in the Virtual Simulation.**		
Step 3: Student-Led Activity (20 mins)	*The teacher guides the student to open the code from the previous class.* *[Student Activity 1]*	**ESR**: A rocket!

	What do you think you will need to create a rocket launch simulation scene? How would you create a rocket?	**ESR**: We can use 3D models.
	Yes, that's one of the options. Can you tell me what happens when a rocket is launched?	**ESR**: When a rocket is launched, as it goes up in space, the parts of the rocket detach.
	Yes. Amazing! To do this, we will need multiple parts which can fall down once clicked on the screen, so instead of using a model, we will need to create our own shape for the rocket. Let's create a rocket shape. What do you think are the primitives you can use? Great! Let's start with <a-cone> for part1 and <a-cylinder> for part2. *Guide the student to set the id, position, color, radius-bottom, radius-top for the <a-cone> and id, position, color, radius, height, and color <a-cylinder> to set the orientation of the shape.*	**ESR**: Cone, cylinders, and triangles.

```
<!--Part 1-->
<a-cone  id="part1"  position="0 1.5 0" color="tomato" radius-bottom="0.5" radius-top="0.1" >
</a-cone>

<!--Part 2-->
<a-cylinder  id="part2"  position="0 -1 0" color="orange" radius="0.5" height="4" >
</a-cylinder>
```

	Great! Now, let's make the bottom part with a cylinder and a few triangles. *Guide the student to create <a-cylinder>.* Now let's add a few triangles at the bottom. How do you draw triangles in 2D? Yes. Great! We can use <a-triangle> with vertex-a, vertex-b, vertex-c to set 3 different vertices. These 3 points will be in 3D. *Guide the student to set the position, scale, rotation and vertices attributes in <a-triangle>.* ***Note**: Let the student come with their own values for the proper orientation.*	**ESR**: To draw a triangle we need 3 vertices.

```
<!--Part 3-->

<a-cylinder id="part3"  position="0 -3 0" radius="0.5" color="blue" height="1.5" >

  <a-triangle position="-0.93 -0.95 -0.94" scale="0.5 0.5 1" color="blue"
  vertex-a="1 1 1" vertex-b="1 3 1" vertex-c="-1 1 1"></a-triangle>

  <a-triangle position="-0.11 -0.026 -0.94" rotation="0 0 270" scale="0.5 0.5 1" color="blue"
  vertex-a="1 1 1" vertex-b="1 3 1" vertex-c="-1 1 1"></a-triangle>

</cylinder>
```



Now we have got the shape, let's move it upwards.

Remember how to write a component to move a box. We can write the same component for the shape we have drawn.

Guide the student to write the A-Frame component called "move", to move the rocket shape in the Y direction.

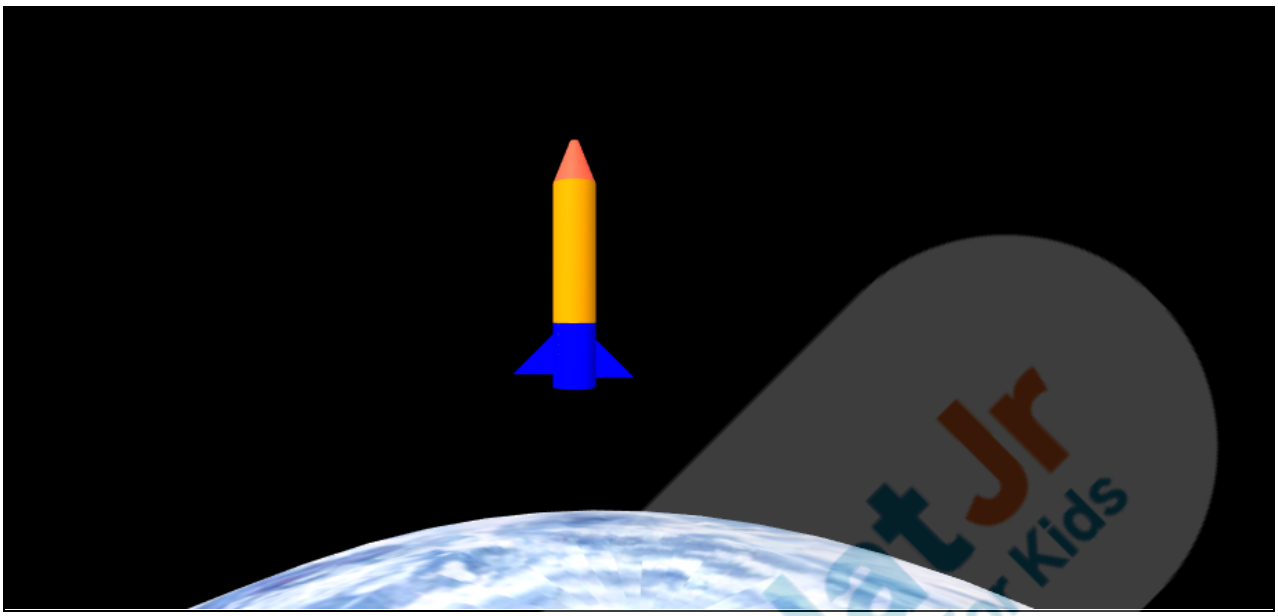
```
AFRAME.registerComponent("move", {
  schema: {
    moveY: { type: "number", default: 1 },
  },

  tick: function () {
    var pos = this.el.getAttribute("position");
    this.data.moveY = this.data.moveY + 0.01;

    pos.y = this.data.moveY;

    this.el.setAttribute("position", { x: pos.x, y: pos.y, z: pos.z });
  },
});
```

	What should we do next? Yes. We can do that too, or we can apply it to the entire shape to move the shape altogether. Let's apply the component to the entire shape. To do that, we should put all the parts under one entity. *Guide the student to make all parts as the child of <a-entity> and add the "move" component.*	**ESR**: We can apply this to every shape.

```
<a-entity id="rocket" position="0 0 0" rotation="0 0 0" move>

  <!--Part 1-->
  <a-cone  id="part1"  position="0 1.5 0" color="tomato" radius-bottom="0.5" radius-top="0.1" >
  </a-cone>

  <!--Part 2-->
  <a-cylinder  id="part2"  position="0 -1 0" color="orange" radius="0.5" height="4" >
  </a-cylinder>

  <!--Part 3-->

  <a-cylinder id="part3"  position="0 -3 0" radius="0.5" color="blue" height="1.5" >

    <a-triangle position="-0.93 -0.95 -0.94" scale="0.5 0.5 1" color="blue"
    vertex-a="1 1 1" vertex-b="1 3 1" vertex-c="-1 1 1"></a-triangle>

    <a-triangle position="-0.11 -0.026 -0.94" rotation="0 0 270" scale="0.5 0.5 1" color="blue"
    vertex-a="1 1 1" vertex-b="1 3 1" vertex-c="-1 1 1"></a-triangle>

  </cylinder>

</a-entity>
```

	Great! Now, what can we do to simulate a scene similar to the launch? To do this we should be able to see the shape move upwards along the far view of the Earth as it goes higher and higher, right? Okay, let's quickly create the sphere entity again for the Earth view.	**ESR**: Varied. **ESR**: Yes.

```
<!--Earth-->
<a-entity position="0 0 0" rotation="0 0 0"
  animation="property: rotation; to: 0 360 0;easing:linear; loop: true; dur: 1000000">
  <a-sphere position="0 -10 10" radius="10"
  src="https://cdn.glitch.com/850dd813-e5e9-4dd0-9751-38981ae74172%2Fearth.jpg?v=1604312404511">
  </a-sphere>
</a-entity>
```



	Great! Now let's write the custom component to move the camera position. How can the position of the camera be changed to have a zoom-out effect? *Guide the student to write a component to zoom out from the camera view and attach the component to <a-camera>.*	**ESR**: z-axis.

```
AFRAME.registerComponent("camera-zoom-out", {
  schema: {
    moveZ: { type: "number", default: 10 },
  },

  tick: function () {
    this.data.moveZ = this.data.moveZ + 0.01;

    var pos = this.el.getAttribute("position");

    pos.z = this.data.moveZ;

    this.el.setAttribute("position", { x: pos.x, y: pos.y, z: pos.z });
  },
});
```

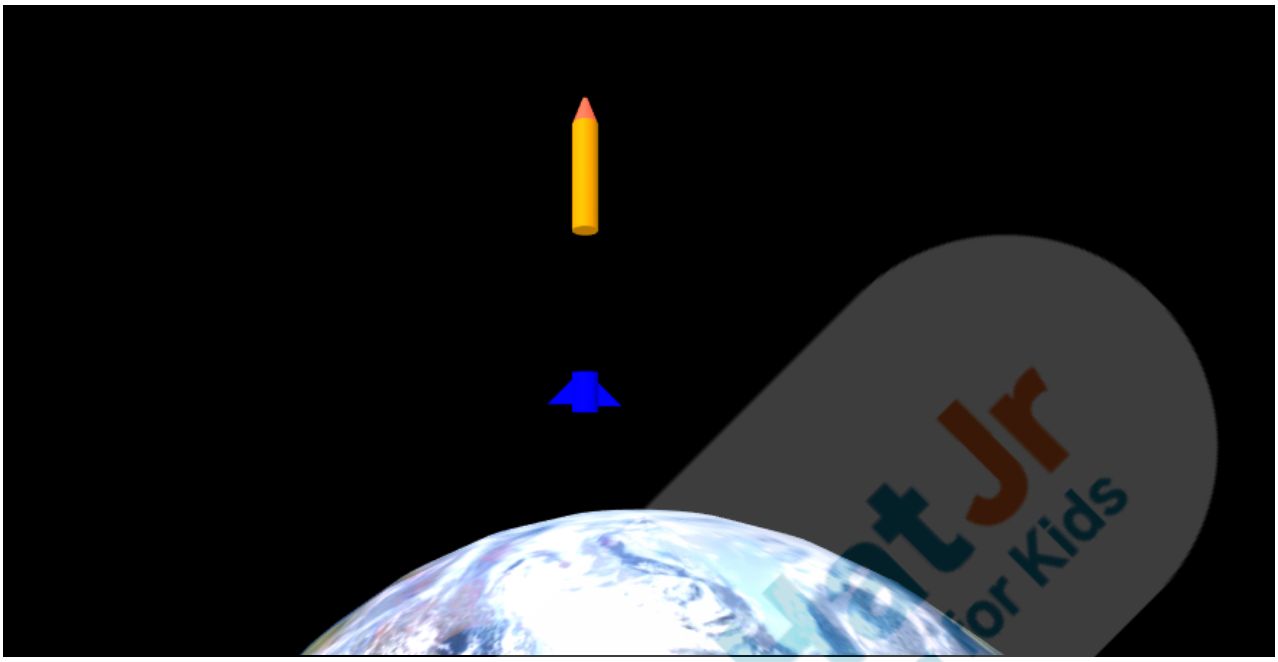
```
<a-camera wasd-controls-enabled="false" position="0 1.6 0" camera-zoom-out>
</a-camera>
```

	Great! Now we can see the rocket flying up and also the view is changing as it goes up! Now, what's next? Amazing! Let's again write a component and add a window click event which will make the third part of the rocket shape fall down on click. *Guide the student to write a component called "fall-down" to update the position attribute of the 3rd part of the rocket shape in the*	ESR: Making the part fall down on a click!

	negative direction and attach the component to the part3 cylinder entity. *<a-cylinder id="part3" …..></a-cylinder>.*	

```
AFRAME.registerComponent("fall-down", {
  schema: {
    moveY: { type: "number", default: 0 },
  },
  tick: function () {

    window.addEventListener("click", (e) => {

      this.data.moveY = this.data.moveY - 0.001;

    });
    var pos = this.el.getAttribute("position");
    pos.y = pos.y + this.data.moveY;
    this.el.setAttribute("position", { x: pos.x, y: pos.y, z: pos.z });
  }
});
```

```
<!--Part 3-->

<a-cylinder id="part3"  position="0 -3 0" radius="0.5" color="blue" height="1.5" fall-down>

  <a-triangle position="-0.93 -0.95 -0.94" scale="0.5 0.5 1" color="blue"
  vertex-a="1 1 1" vertex-b="1 3 1" vertex-c="-1 1 1"></a-triangle>

  <a-triangle position="-0.11 -0.026 -0.94" rotation="0 0 270" scale="0.5 0.5 1" color="blue"
  vertex-a="1 1 1" vertex-b="1 3 1" vertex-c="-1 1 1"></a-triangle>

</cylinder>
```



	That's really amazing! You created a very simple simulation with the help of components and triggering a window event.	

Teacher Guides Student to Stop Screen Share

FEEDBACK

- **Compliment the student for her/his effort in the class.**
- **Encourage the student to think and come up with their own solutions.**

Teacher starts slideshow **from slide 21 to slide 31**

Activity details	Solution/Guidelines
Run the presentation from slide 21 to slide 31 **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 with us.

Quiz time - Click on in-class quiz	
Question	**Answer**
What does DOM stand for? A. Domain Object Model B. Document Oriented Model C. Document Object Model D. Domain Oriented Model	**C**
_____ is created every time something happens to the elements on the webpage. A. An object B. An event C. An action D. The component	**B**
Which primitives were used to create the rocket? A. <a-cone> B. <a-cylinder> C. <a-square> D. <a-triangle>	**A**
• **End the quiz panel**	
You get a "hats-off". Alright. I will look forward to seeing you create your own component.	*Make sure you have given at least 2 Hats Off during the class for:* 

<table>
<tr><td></td><td></td><td> 
</td></tr>
<tr><td>Project Overview</td><td>ADDING CLICK EVENT

Goal of the Project:

In this project, you have to write a component to add click events.

Story:
Jess' showcase went great with the design, but the client has requested to see the car model from all the views. Help Jess write an event-driven component to see all the sides of the car model on the click.

Write a component to add click event listeners to see all sides of the model.

I am excited to see the on-screen click event.

Bye!</td><td></td></tr>
<tr><td colspan="3">Teacher ends slideshow  </td></tr>
<tr><td colspan="3">Teacher Clicks 
</td></tr>
<tr><td>Additional Activities I</td><td>Encourage the student to write reflection notes in their reflection journal using markdown.</td><td>The student uses the markdown editor to write</td></tr>
</table>

	Use these as guiding questions: ● What happened today? ○ Describe what happened. ○ The 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?	*their reflections in a reflection journal.*

Activity	Activity Name	Links
Teacher Activity 1	A-Frame DOM and JavaScript Event Listener Illustration	https://curriculum.whitehatjr.com/PRO+Asset/A-Frame+DOM+and+Event+Listerners.pdf
Teacher Activity 2	Reference Code	https://github.com/whitehatjr/PRO-C152-Activity-Reference
Student Activity 1	Previous Class Code	https://github.com/whitehatjr/PRO-C152-Student-Actitvity
Project Solution Link	Adding Click Event	https://github.com/whitehatjr/PRO-VR-C152
Teacher Ref. Visual Aid Link	Visual Aid link	https://curriculum.whitehatjr.com/Visual+Project+Asset/PRO_VD/PRO_C152_withcues.html
Teacher Ref. In-Class Quiz	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/f4c4bf13-6e22-404e-a433-0e8b31ede727.pdf