Topic	FLIGHT SIMULATION DESIGN	
Class Description	**Students create a flight simulation. Students will learn how to use keyboard events to control 3D models.**	
Class	**C153**	
Class time	**45 mins**	
Goal	• Create a flight simulation scene. • Implement keyboard events to control the flight.	

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**

WARM-UP SESSION - 05 mins

CONTEXT

- **Implementing Javascript events listeners in the flight simulation.**
- **Controlling the flight and terrain with the arrow keys.**

Teacher starts slideshow from slides 1 to 11 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 4** **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**
The ____ method allows you to add event listeners on any HTML DOM object. A. read() B. on() C. addEventListener() D. tick()	**C**
An event is created every time something happens to the _____ on the webpage. A. element B. document C. window D. object	**A**
Continue the WARM-UP session	

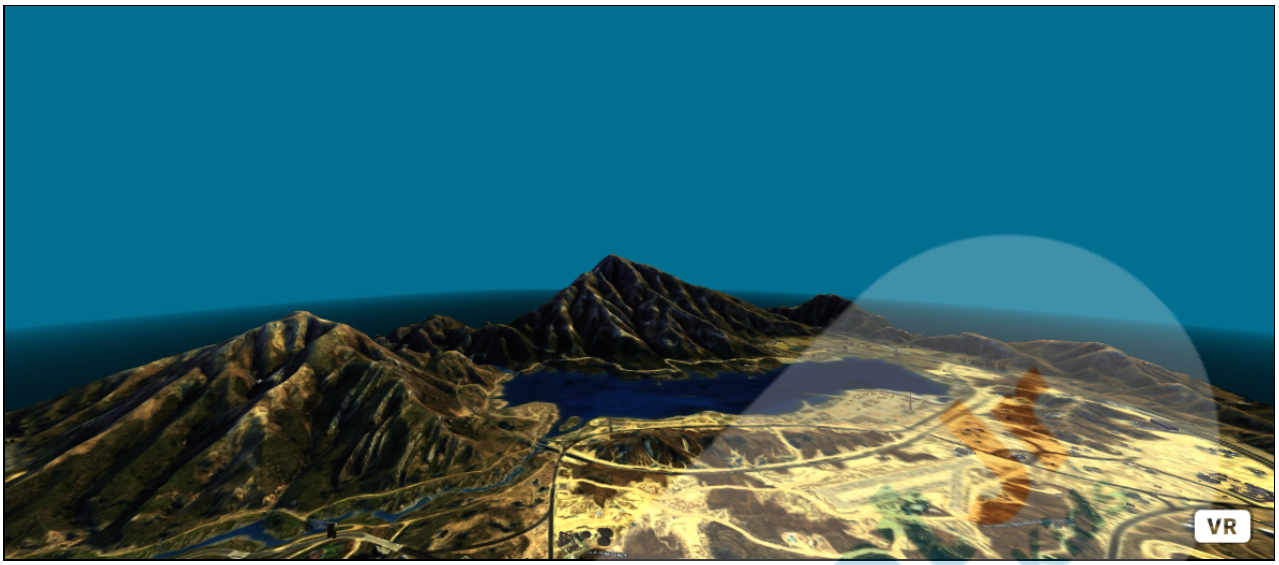
Activity details		Solution/Guidelines
Run the presentation from slide 5 to slide 11 to set the problem statement. **Following are the WARM-UP session deliverables:** • **Implementing Javascript events listeners in the flight simulation.** • **Controlling the flight and terrain with the arrow keys.**		Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.
Teacher ends slideshow		
TEACHER-LED ACTIVITY - 15 mins		
Teacher Initiates Screen Share		
CHALLENGE • **Create a flight simulation with the help of 3D models.** • **Control the objects with the help of keyboard events.**		
Step 2: Teacher-led Activity (20 mins)	What do you think can be used to create a flight simulation scene? Yes, great! To start with, let's load the gLTF models in the A-Frame scene. ***[Teacher Activity 1]*** *The teacher downloads code.* Do you remember the asset management system in A-Frame?	**ESR**: Flight and a ground. **ESR**: Yes.

	Let's create <a-asset-item> and set an id for the terrain map and render the model with the help of the id. ***Note**: Teachers should toggle between the code and the output screen to show the difference.*	

```
<!--Asset Management-->
<a-assets timeout="10000">
  <a-asset-item
    id="terrainMap"
    src="./assets/models/terrain/scene.gltf"
  ></a-asset-item>

</a-assets>
```

```
<!-- Terrain  -->
<a-entity
  id="terrain"
  gltf-model="#terrainMap"
  position="0 0 0"
  scale="0.3 0.3 0.3"
>
</a-entity>
```

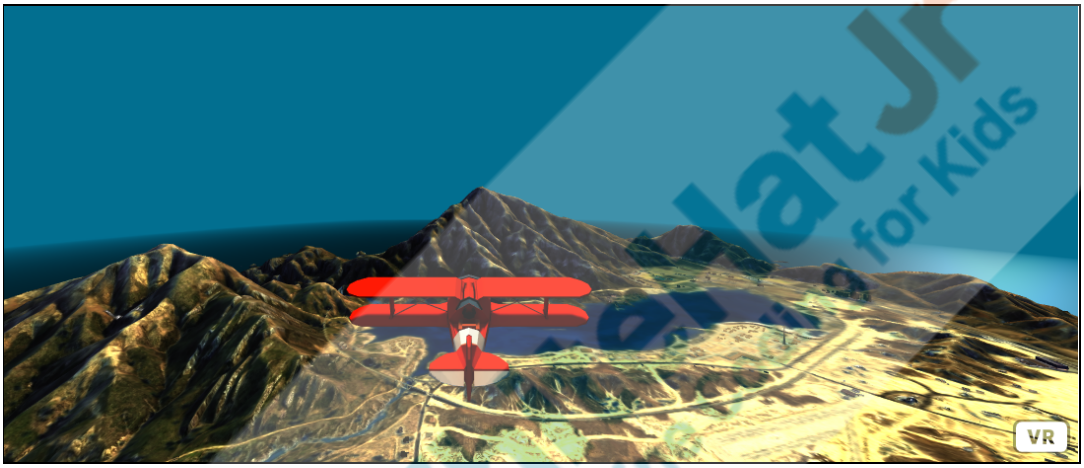


	Now, we have got our terrain. What's next? Yes. Let's add that too. *The teacher creates <a-asset-item> for the plane model and renders the plane the screen using <a-entity>.* *The teacher sets the position, rotation, and scale to set the orientation of the gLTF model.*	**ESR**: The flight.

```
  <a-asset-item
    id="plane"
    src="./assets/models/airplane/scene.gltf"
  ></a-asset-item>

</a-assets>
```

```
<!-- Plane -->
<a-entity
  id="plane_model"
  gltf-model="#plane"
  position="0 0 15"
  scale="1 1"
  rotation="0 90 0"
>
</a-entity>
```



	Now, we got our models in the scene. What can be done next?	**ESR**: Move the flight.
	Yes. How can we move the flight forward?	**ESR**: Move the terrain backward.
	Amazing!	
	What should we use to move the terrain map backward?	**ESR**: The animation component.
	Yes, great!	

	The teacher adds the animation component for the 'position' property and sets the position for the z-axis. We can keep the animation loop true to repeat the animation cycle infinitely.	

```
<!-- Terrain  -->
<a-entity
  id="terrain"
  gltf-model="#terrainMap"
  position="0 0 0"
  scale="0.3 0.3 0.3"
  animation="property: position; to: 0 0 1000;easing:linear; loop: true; dur: 150000"
>
</a-entity>
```

	Now the flight seems to be moving forward. We should be able to control the flight movement in the left and right directions. What should we do to move the flight left and right? Yes, exactly! What should we do about that? We can write a component that can control the terrain map from left to right and vice-versa whenever the arrow key is pressed. What do you think should be updated to move left/right?	**ESR**: The terrain should move left and right too. **ESR**: Varied. **ESR**: The position/rotation.

	Instead of position, we should update the rotation attribute of the terrain to create a turning effect. Let's take a variable to control the speed of rotation under "schema" with a default value of 0. Now, in which function should we update the rotation attribute? Yes, great! The reason is to keep rotating the map in that direction. *The teacher writes the code in **Rotation.js** to update the rotation attribute.* *Make sure to add the **Rotation.js** file in <head>.*	**ESR**: .tick() method

```
Code > components > JS Rotation.js > ...
//Terrain Rotation
AFRAME.registerComponent("terrain-rotation-reader", {
  schema: {
    speedOfRoation: { type: "number", default: 0 }
  },
  tick: function () {
    var mapRotation = this.el.getAttribute("rotation");
    mapRotation.y += this.data.speedOfRoation;

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

```
<head>

  <script src="https://aframe.io/releases/1.0.4/aframe.min.js"></script>

  <script src="./components/Rotation.js"></script>

</head>
```

This will continuously rotate the terrain map.

But we should only start rotation when the left and right arrow keys are pressed.

The teacher adds the event listener code for the "keydown" event.

Each key has specific codes that can be accessed using the "key" attribute of the event "e".

	Write the condition for the “ArrowRight” and “ArrowLeft” keycodes. Update the rotation value to certain angles to avoid absurd rotation.	

-Code > components > JS Rotation.js > ...

```javascript
//Terrain Rotation
AFRAME.registerComponent("terrain-rotation-reader", {
  schema: {
    speedOfRoation: { type: "number", default: 0 }
  },
  init: function () {
    window.addEventListener("keydown", (e) => {
      if (e.key === "ArrowRight") {
        if (this.data.speedOfRoation < 0.1) {
          this.data.speedOfRoation += 0.01;
        }
      }
      if (e.key === "ArrowLeft") {
        if (this.data.speedOfRoation > -0.1) {
          this.data.speedOfRoation -= 0.01;
        }
      }
    });
  },
  tick: function () {
    var mapRotation = this.el.getAttribute("rotation");

    mapRotation.y += this.data.speedOfRoation;

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

	Now attach the component to the entity.	

```
<!-- Terrain  -->
<a-entity
  id="terrain"
  gltf-model="#terrainMap"
  position="0 0 0"
  scale="0.3 0.3 0.3"
  animation="property: position; to: 0 0 1000;easing:linear; loop: true; dur: 150000"
  terrain-rotation-reader
>
</a-entity>
```

	We can see the terrain map move whenever the keys are pressed. You will be creating controls for the flight.	

Teacher Stops Screen Share

	Now it's your turn. Please share your screen with me.	

STUDENT-LED ACTIVITY - 20 mins

ACTIVITY

- **Create a flight simulation scene.**
- **Implement the Javascript keyboards to control flight movements.**

Teacher starts slideshow from slides 12 to 14

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

Step 3: Student-Led Activity (20 mins)	*Encourage the student to do the activities and come up with their own solution.* *Guide the student to download the code.* ***[Student Activity 1]*** *Guide the student to write the component to control flight movements.* *Take a variable to control the rotation of the flight with the default value set to 0.*	

```
//Plane rotation component
AFRAME.registerComponent("plane-rotation-reader", {
  schema: {

    speedOfRoation: { type: "number", default: 0 },

  },
```

	Now, what should we do next? Great! *Guide the student to write the code for the keydown event listener and write the conditions for different arrows to rotate the flight to a particular limit in all directions.*	**ESR**: Add keydown event.

Code > components > JS Rotation.js > ...

```
init: function () {
  window.addEventListener("keydown", (e) => {
    //get the data from the attributes
    this.data.speedOfRoation = this.el.getAttribute("rotation");

    //variables to store data
    var planeRotation = this.data.speedOfRoation;

    //control the attributes with the Arrow Keys
    if (e.key === "ArrowRight") {
      if (planeRotation.x < 10) {
        planeRotation.x += 0.5;
        this.el.setAttribute("rotation", planeRotation);
      }
    }
    if (e.key === "ArrowLeft") {
      if (planeRotation.x > -10) {
        planeRotation.x -= 0.5;
        this.el.setAttribute("rotation", planeRotation);
      }
    }
    if (e.key === "ArrowUp") {
      if (planeRotation.z < 20) {
        planeRotation.z += 0.5;
        this.el.setAttribute("rotation", planeRotation);
      }
    }
    if (e.key === "ArrowDown") {
      if (planeRotation.z > -10) {
        planeRotation.z -= 0.5;
        this.el.setAttribute("rotation", planeRotation);
      }
    }
  });
}
});
```

	Now you can attach the component to the flight entity. *Guide the student to attach the component to the entity.*	

```
<!-- Plane -->
<a-entity
  id="plane_model"
  gltf-model="#plane"
  position="0 0 15"
  scale="1 1"
  rotation="0 90 0"
  plane-rotation-reader
>
</a-entity>
```

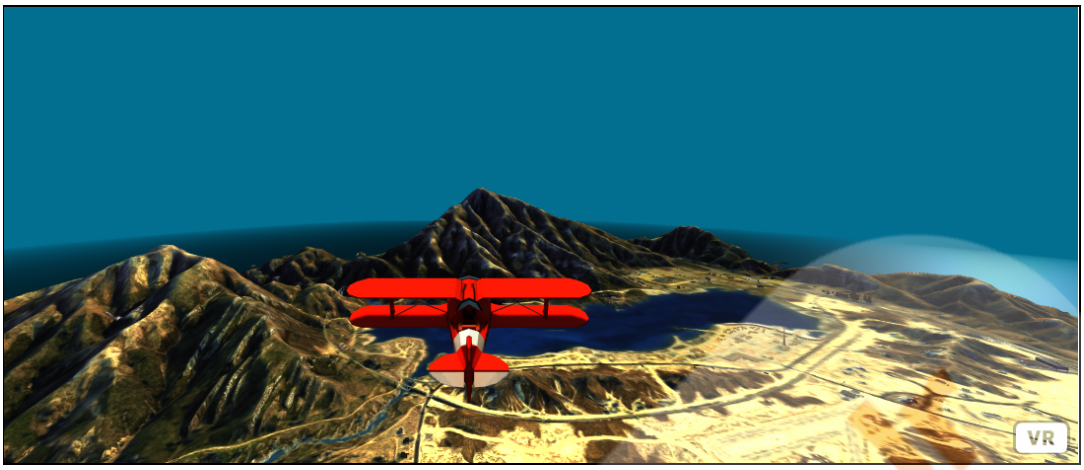
	Now, what else can we do? We can update the position attribute as the flight ascends (goes up) and descends (goes down). How would you do that? Amazing! Let's take a variable to update the position attribute to move it up and down when the up and down arrow key is pressed respectively.	**ESR**: Varied. **ESR**: Take a variable in the schema to control position.

```
AFRAME.registerComponent("plane-rotation-reader", {
    schema: {
      speedOfRoation: { type: "number", default: 0 },
      speedOfAscent: { type: "number", default: 0 }
    },
    init: function () {
      window.addEventListener("keydown", (e) => {

        //get the data from the attributes
        this.data.speedOfRoation = this.el.getAttribute("rotation");
        this.data.speedOfAscent = this.el.getAttribute("position");

        var planeRotation = this.data.speedOfRoation;
        var planePosition = this.data.speedOfAscent;

        //control the attributes with the Arrow Keys
        if (e.key === "ArrowRight") {
          if (planeRotation.x < 10) {
            planeRotation.x += 0.5;
            this.el.setAttribute("rotation", planeRotation);
          }
        }
        if (e.key === "ArrowLeft") {
          if (planeRotation.x > -10) {
            planeRotation.x -= 0.5;
            this.el.setAttribute("rotation", planeRotation);
          }
        }
        if (e.key === "ArrowUp") {
          if (planeRotation.z < 20) {
            planeRotation.z += 0.5;
            this.el.setAttribute("rotation", planeRotation);
          }
          if (planePosition.y < 2) {
            planePosition.y += 0.01;
            this.el.setAttribute("position", planePosition);
          }
        }
        if (e.key === "ArrowDown") {
          if (planeRotation.z > -10) {
            planeRotation.z -= 0.5;
            this.el.setAttribute("rotation", planeRotation);
          }
          if (planePosition.y > -2) {
            planePosition.y -= 0.01;
            this.el.setAttribute("position", planePosition);
          }
        }
      });
```



	Great! That's really interesting to see. You have done a good job in creating a control simulation.	

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 15 to slide 24

Activity details	Solution/Guidelines
Run the presentation from slide 15 to slide 24 **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 should we use to move the terrain map backward? A. gltf-model B. scale C. animation component D. position	**D**
In which function should we update the rotation attribute to keep rotating the map in that direction? A. tynker B. tick C. thunkable D. tinkerbell	**B**
A variable is used to update the _____ to move it up and down when the up and down arrow key is pressed respectively. A. rotation attribute B. position attribute C. translation attribute D. vertical attribute	**B**
• **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:* Creatively Solved Activities +10 Great Question +10

		Strong Concentration +10
Project Overview	**SCUBA DIVING SIMULATION** **The goal of the Project:** In this project, you will create a virtual underwater scene for the scuba diver to explore the world under the ocean. **Story:** Your friend always wanted to go scuba diving, but he is terrified of water and the high waves in the ocean. He always wished to travel under the ocean without going underwater. Write an A-Frame program to create a virtual underwater ocean scene and add the scuba diver's movement component. I am very excited to see how to create a virtual ocean for your friend. Bye!	
Teacher ends slideshow		
Teacher Clicks	✖ End Class	

Additional Activities	*Encourage the student to write reflection notes in their reflection journal using markdown.*
	
	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?
		The student uses the markdown editor to write their reflections in a reflection journal.

Activity	Activity Name	Links
Teacher Activity 1	Teacher Boilerplate	https://github.com/whitehatjr/PRO-C153-Teacher-Boilerplate
Teacher Activity 2	Teacher Reference Code	https://github.com/whitehatjr/PRO-C153-Teacher-Reference-Code/
Teacher Activity 3	Output Reference	https://curriculum.whitehatjr.com/PRO+Asset/898276df32b94341af5622b74721b0c8.mp4
Student Activity 1	Flight Simulation Stage 1	https://github.com/whitehatjr/PRO-C153-Student-Activity/
Project Solution Link	Scuba Diving Simulation	https://github.com/whitehatjr/PRO-C153-Project
Teacher Ref. Visual Aid Link	Visual Aid link	https://curriculum.whitehatjr.com/Visual+Project+Asset/PRO_VD/PRO_C153_withcues.html
Teacher Ref.	In-Class Quiz	https://docs.google.com/document/d/12t07XXfb

In-Cass Quiz		https://s3-whjr-curriculum-uploads.whjr.online/f402305f-14ff-465a-bd78-ff68ea6e4234.pdf