PRO-C174

WhiteHat Jr
Live Online Coding for Kids

BARCODE MARKER BASED AR

What is our GOAL for this MODULE?

We learned about barcode markers and created Augmented reality based on barcode markers.

What did we ACHIEVE in the class TODAY?

- Learned to use barcode markers for web based Augmented reality in A-Frame.
- Learned to show atomic molecules in Augmented reality using barcode markers.

Which CONCEPTS/CODING BLOCKS did we cover today?

- 3X3 Barcode Markers
- ngrok to run the application.

How did we DO the activities?

1. Specify the detection mode and matrix code attributes.

```
<head>
    <meta charset="utf-8" />
    <title>AR Chemistry</title>
    <meta name="viewport" content="width=device-width, user-scalable=no, minimum-scale=1.0, maximum-scale=1.0" />
    <script src="https://aframe.io/releases/1.0.1/aframe.min.js"></script>
    <script src="https://rawgit.com/jeromeetienne/AR.js/master/aframe/build/aframe-ar.min.js"></script>
    <script src="./js/Compounds.js"></script>
</head>

<body style="margin : 0px; overflow: hidden;">

    <!-- Add detectionMode and matrixCodeType to tell AR.js to recognize barcode markers -->

    <a-scene id="main-scene" vr-mode-ui="enabled: false"
        arjs="sourceType: webcam; debugUIEnabled: false; detectionMode: mono_and_matrix; matrixCodeType: 3x3;" embedded>
        <a-entity id="camera" camera></a-entity>

    </a-scene>
</body>
```

2. Create compoundList.json & elementColors.json files and add data about the Na & Cl element.

compoundList.json

```
{
  "0": {
    "element_name": "Na",
    "barcode_value": 2,
    "number_of_electron": 1
  },

  "1": {
    "element_name": "Cl",
    "barcode_value": 5,
    "number_of_electron": 7
  }

}
```

elementColors.json

```
{
  "Na": "#8000FF",
  "Cl": "#4CAF50"
}
```

3. Register the **atoms** component atoms to write functions:

   - **getCompounds()** to get the element data: name, number of electrons and barcode value.
   - **getElementColors()** to get the color of the element, and
   - **createAtoms()** to create the 3D atomic structure(nucleus and free electrons) of the elements.

```
AFRAME.registerComponent("atoms", {
    init: async function() {


    },
    getCompounds: function() {
      //Get compound details from the JSON file


    },
    getElementColors: function() {
      //Get color of the element from the JSON file


    },
    createAtoms: async function(element) {


      //Add BARCODE marker


      //Add atom card


      //Add nucleus of the


      //Create orbit for electron rvolution


      //Create revolving electrons

    }
});
```

4. Attach the **atoms** component to the <a-scene> element.

```
<!-- Add detectionMode and matrixCodeType to tell AR.js
<a-scene id="main-scene" vr-mode-ui="enabled: false"
  arjs="sourceType: webcam; debugUIEnabled: false;
  detectionMode: mono_and_matrix; matrixCodeType: 3x3;"
  embedded
  atoms
  <a-entity id="camera" camera></a-entity>
</a-scene>
```

5. Write the functions **getCompounds()** and **getElementColors()** to fetch details of the elements and colors. Use the **fetch()** function to get the details from the JSON file.

```
getCompounds: function() {
  return fetch("js/compoundList.json")
    .then(res => res.json())
    .then(data => data);
},

getElementColors: function() {
  return fetch("js/elementColors.json")
    .then(res => res.json())
    .then(data => data);
},
```

6. Call **getCompunds()** to fetch the values of elements which can then be passed to call the **createAtoms()** function.

```
init: async function () {

  //Get the compund details of the element
  var compounds = await this.getCompounds();

  var barcodes = Object.keys(compounds);

  barcodes.map(barcode => {
    var element = compounds[barcode];

    //Call the function
    this.createAtoms(element);
  });

},
```

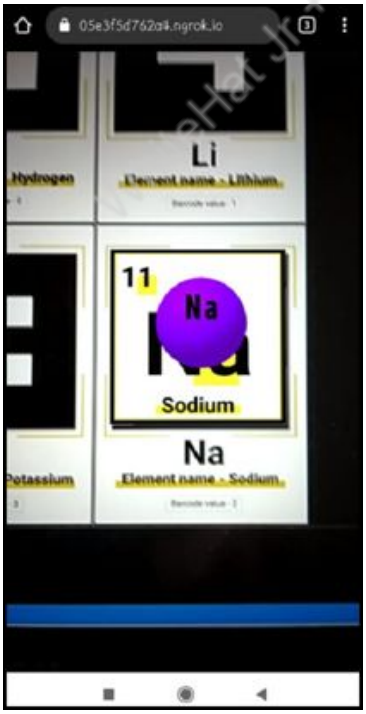
7. Create the **marker**, **atoms**, **nucleus**, **orbit** and **electrons** entity and add these in the scene.

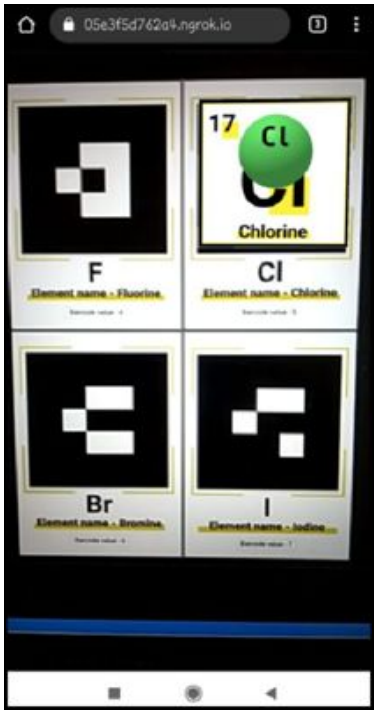
```
//Element data
var elementName = element.element_name;
var barcodeValue = element.barcode_value;
var numOfElectron = element.number_of_electron;

//Get the color of the element
var colors = await this.getElementColors();

//Scene
var scene = document.querySelector("a-scene");

//Add marker entity for BARCODE marker
var marker = document.createElement("a-marker");

marker.setAttribute("id", `marker-${barcodeValue}`);
marker.setAttribute("type", "barcode");
marker.setAttribute("element_name", elementName);
marker.setAttribute("value", barcodeValue);

scene.appendChild(marker);
```

```
var atom = document.createElement("a-entity");
atom.setAttribute("id", `${elementName}-${barcodeValue}`);
marker.appendChild(atom);

//Add atom card
var card = document.createElement("a-entity");
card.setAttribute("id", `card-${elementName}`);
card.setAttribute("geometry", {
  primitive: "plane",
  width: 1,
  height: 1
});

card.setAttribute("material", {
  src: `./assets/atom_cards/card_${elementName}.png`
});

card.setAttribute("position", { x: 0, y: 0, z: 0 });
card.setAttribute("rotation", { x: -90, y: 0, z: 0 });

atom.appendChild(card);
```

```
//Add nucleus
var nucleusRadius = 0.2;
var nucleus = document.createElement("a-entity");
nucleus.setAttribute("id", `nucleus-${elementName}`);
nucleus.setAttribute("geometry", {
  primitive: "sphere",
  radius: nucleusRadius
});

nucleus.setAttribute("material", "color", colors[elementName]);
nucleus.setAttribute("position", { x: 0, y: 1, z: 0 });

nucleus.setAttribute("rotation", { x: 0, y: 0, z: 0 });

var nucleusName = document.createElement("a-entity");
nucleusName.setAttribute("id", `nucleus-name-${elementName}`);
nucleusName.setAttribute("position", { x: 0, y: 0.21, z: -0.06 });
nucleusName.setAttribute("rotation", { x: -90, y: 0, z: 0 });
nucleusName.setAttribute("text", {
  font: "monoid",
  width: 3,
  color: "black",
  align: "center",
  value: elementName
});

nucleus.appendChild(nucleusName);

atom.appendChild(nucleus);
```

8. Test the output using ngrok.





9. Add the entity to create an orbit path.

```
var orbitAngle = -180;

for (var num = 1; num <= numOfElectron; num++) {
  var orbit = document.createElement("a-entity");
  orbit.setAttribute("geometry", {
    primitive: "torus",
    arc: 360,
    radius: 0.28,
    radiusTubular: 0.001
  });

  orbit.setAttribute("material", {
    color: "#ff9e80",
    opacity: 0.3
  });

  orbit.setAttribute("position", {
    x: 0,
    y: 1,
    z: 0
  });

  orbit.setAttribute("rotation", {
    x: 0,
    y: orbitAngle,
    z: 0
  });

  orbitAngle += 45;

  atom.appendChild(orbit);

}
```

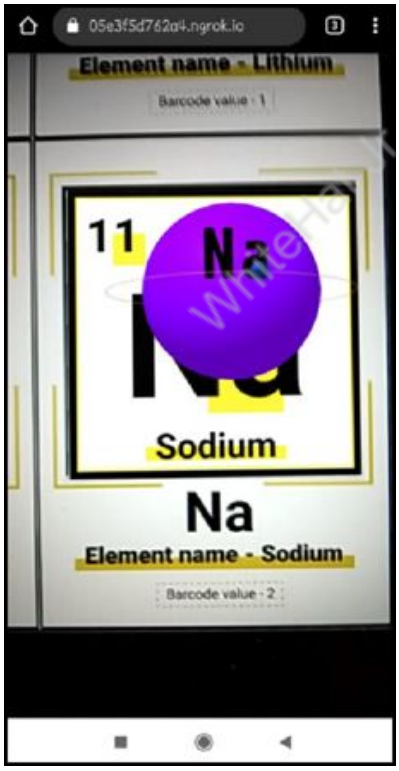
10. Create the revolving electron.

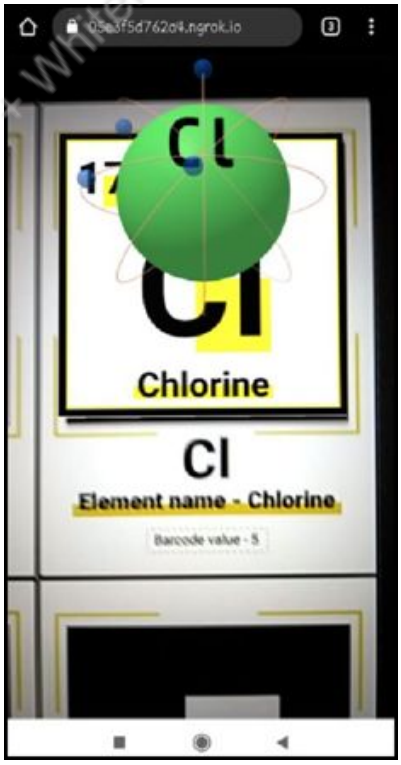
```
var electronAngle = 30;
```

```
//<a-entity>
var electronGroup = document.createElement("a-entity");
electronGroup.setAttribute("id", `electron-group-${elementName}`);

electronGroup.setAttribute("rotation", {
    x: 0,
    y: 0,
    z: electronAngle
});

electronAngle += 65;

//animation
electronGroup.setAttribute("animation", {
    property: "rotation",
    to: `0 0 -360`,
    loop: "true",
    dur: 3500,
    easing: "linear"
});

orbit.appendChild(electronGroup);
```

```
//electron
var electron = document.createElement("a-entity");
electron.setAttribute("id", `electron-${elementName}`);
electron.setAttribute("geometry", {
    primitive: "sphere",
    radius: 0.02
});

electron.setAttribute("material", { color: "#0d47a1", opacity: 0.6 });

electron.setAttribute("position", {
    x: 0.2,
    y: 0.2,
    z: 0
});

electronGroup.appendChild(electron);
```

11. Test the code using ngrok.





We learned about the atomic structure of elements and rendered them in Augmented reality.

What's NEXT?

In the next class, we will learn about finding distance between markers and the formation of compounds between elements in augmented reality.

EXTEND YOUR KNOWLEDGE:

- You can refer to the link below to explore more about A-Frame
  A-Frame