SASTA OLIPHANT SCIENCE AWARDS

SOUTH AUSTRALIAN SCIENCE TEACHERS ASSOCIATION

Highly Commended

Programming, Apps & Robotics

Year 9-10

Jacinta Atterton
Thien-An Dang

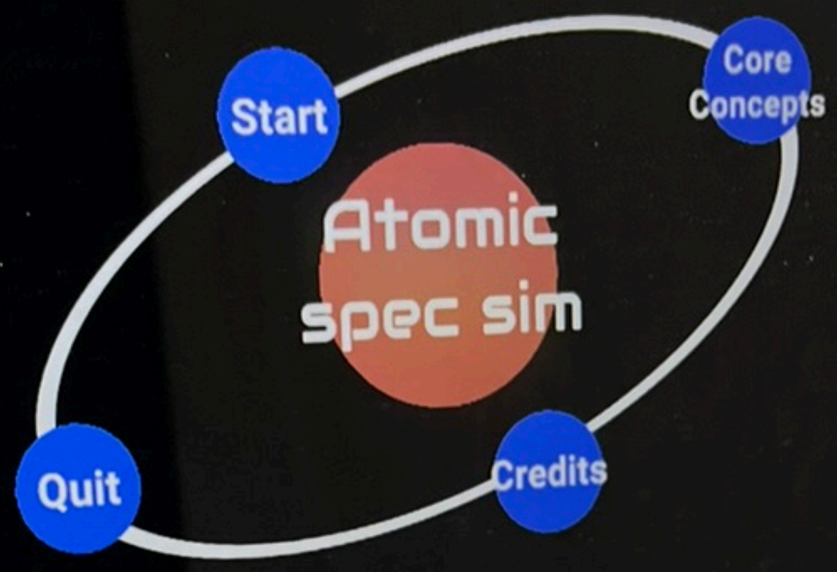
Thomas More College

ROWE SCIENTIFIC PTY. LTD. www.rowe.com.au

Australian Government
Department of Defence

Government of South Australia
Department for Education

SASTA South Australian Science Teachers Association


Start
Core Concepts
Atomic spec sim
Quit
Credits

Atomic spectroscopy simulator report

Aim of entry

The aim of this simulation is to create a program that will demonstrate the scientific concept, visual spectroscopy. This will serve the purpose of providing a visual demonstration of the concept for students and scientists alike, making it easier to understand visual spectroscopy.

Type of device required to run program

Any device that can run unity

Instructions on using entry

Open the program

Press start

Select element

<u>If selecting preset element:</u>

Program inserts value(s)

Program calculates the equation

Press next

Results displayed

<u>If selecting custom:</u>

Insert eBase and eLevel values

Press next

Program calculates the equation

Press next

Results displayed

Program

SceneManager.cs

```
using UnityEngine;
using UnityEngine.SceneManagement;
using System.Collections;

public class SceneChanger : MonoBehaviour
{
   public ElementData hydrogenData;
   // if more elements add here

   public void scene_changer(string scene_name)
   {
      SceneManager.LoadScene(scene_name);
   }

   public void quit()
   {
      Application.Quit();
   }

   public void SelectHydrogen()
   {
      ElementSelection.SelectedElement = hydrogenData;
      SceneManager.LoadScene("Equation");
   }

   public void SelectCustom()
```

```
    {
        SceneManager.LoadScene("Custom inputs");
    }
}
```

ElementSelection.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ElementSelection : MonoBehaviour
{
    public static ElementData SelectedElement;
}
```

ElementData.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

[CreateAssetMenu(fileName = "NewElement", menuName = "Element data")]
public class ElementData : ScriptableObject
{
    public string elementName;
    public float eLevel1;
    public float eLevel2;
    public float eLevel3;
    public float eLevel4;
```

```
    public float eLevel5;
    public float eLevel6;
    public float eBase;
}
```

EquationDisplay.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.Events;

public class EquationDisplay : MonoBehaviour
{

    public Text displayText;
    public Text elementDataText;

    public float e = 1.602f * Mathf.Pow(10, -19); // Charge of an electron
    public float h = 6.62607015f * Mathf.Pow(10, -34);// Planck's constant

    // Start is called before the first frame update
    void Start()
    {
        ElementData element = ElementSelection.SelectedElement;

        SpectralData.frequencies.Clear();
        SpectralData.isCustomElement = element.elementName == "Custom";
```

```
    float[] energyLevels = { element.eLevel1, element.eLevel2, element.eLevel3,
element.eLevel4 };
    float baseLevel = element.eBase;
    float maxFreq = 0f;

    string result = $"Equations for {element.elementName}:\n\n";

    float c = 3f * Mathf.Pow(10, 8); // speed of light in m/s

    foreach (float level in energyLevels)
    {
      if (level == 0f) continue; // skips level if its value is 0

      float diff = level - baseLevel; // in eV
      float E = diff * e; // in joules
      float f = E / h; // frequency in Hz
      float wavelength = (c / f) * Mathf.Pow(10, 9); // convert into nm

      SpectralData.frequencies.Add(f);

      result += $"eLevel = {level}, eBase = {baseLevel}, Diff = {diff}, E = {E:E2} J,
Wavelength = {wavelength:F2} nm\n";
    }

    string elementInfo = $"eLevel 1: {element.eLevel1}, eLevel 2: {element.eLevel2},
eLevel 3: {element.eLevel3}, eLevel 4: {element.eLevel4}, eBase: {element.eBase}";

    displayText.text = result;
```

```
        elementDataText.text = elementInfo;
    }
}
```

ElementImageDisplay.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class ElementImageDisplay : MonoBehaviour
{
    public Image targetImage;

    void Start()
    {
        ElementData currentElement = ElementSelection.SelectedElement;
        if (currentElement != null && currentElement.elementImage !=null)
        {
            targetImage.sprite = currentElement.elementImage;
        }
    }
}
```

CustomElementInput.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
```

```
using UnityEngine.SceneManagement;


public class CustomElementInput : MonoBehaviour
{
   public InputField levelField;
   public InputField baseField;


   public void SubmitAndContinue()
   {
      ElementData custom = ScriptableObject.CreateInstance <ElementData>();
//creates custom data object


      // assign values
      custom.eLevel1 = float.Parse(levelField.text);
      custom.eBase = float.Parse(baseField.text);
      custom.elementName = "Custom";


      // store data
      ElementSelection.SelectedElement = custom;


      // go to equation scene
      SceneManager.LoadScene("Equation");
   }
}
}
```

SpectralData.cs

```
using System.Collections;
```

```
using System.Collections.Generic;
using UnityEngine;

public static class SpectralData
{
   public static List<float> frequencies = new List<float>();
   public static bool isCustomElement;
}
```

fToRGB.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class fToRGB : MonoBehaviour
{
   public Image barImage;
   public RectTransform barTransform;
   public RectTransform referenceImageTransform;

   void Start()
   {

      barImage.gameObject.SetActive(true);

      float refWidth = referenceImageTransform.rect.width;
      Vector3 basePos = referenceImageTransform.position;
```

```
    foreach (float f in SpectralData.frequencies)
    {
        Color c = FrequencyToRGB(f);
        barImage.color = c;

        float offset = GetPositionOffset(f);

        barTransform.anchoredPosition = new Vector3(basePos.x + offset, basePos.y,
basePos.z);
    }
  }

  // determining colour based on frequency

  Color FrequencyToRGB(float freq)
  {
    float wavelength = (3e8f / freq) * 1e9f;

    // return colors - most are built in colors
    if (wavelength >= 380 && wavelength <= 410)
        return new Color(0.627451f, 0.1254902f, 0.9411765f); // violet rgb
    else if (wavelength <= 434)
        return Color.blue;
    else if (wavelength <= 495)
        return Color.cyan;
    else if (wavelength <= 570)
        return Color.green;
```

```
    else if (wavelength <= 590)
        return new Color(1f, 0.5f, 0f); // orange rgb
    else if (wavelength <= 620)
        return Color.red;
    else if (wavelength <= 750)
        return new Color(0.6f, 0f, 0f); // deep red rgb
    else
        return Color.white;
}

// Map frequency to a specific position
float GetPositionOffset(float wavelength, float totalWidth)
{
    // Divide reference image into evenly spaced spectral bands
    float sectionWidth = totalWidth / 7f;

    if (wavelength >= 380f && wavelength <= 410f)
        return -3f * sectionWidth;
    else if (wavelength <= 434f)
        return -2f * sectionWidth;
    else if (wavelength <= 495f)
        return -1f * sectionWidth;
    else if (wavelength <= 570f)
        return 0f; // center
    else if (wavelength <= 590f)
        return 1f * sectionWidth;
    else if (wavelength <= 620f)
        return 2f * sectionWidth;
```

```
        else if (wavelength <= 750f)
            return 3f * sectionWidth;
        else
            return 0f; // fallback to center
    }

}
```

External support

Brendon Huu Nguyen Ngo (student)

Nathan Ackan (teacher)

Bibliography

https://www.youtube.com/watch?v=Kv-hRvEOjuA&t=649s

https://www.youtube.com/watch?v=AznXSVx2xX0&t=629s

https://discussions.unity.com/t/using-createassetmenu-and-scriptableobject/692127

https://docs.unity3d.com/6000.1/Documentation/Manual/class-Mathf.html

https://discussions.unity.com/t/foreach-loop-with-an-array-of-gameobjects/28102

https://docs.unity3d.com/6000.1/Documentation/ScriptReference/Color.html