# DECO 7230 Interactive Prototype III statement

## 1. Statement of Originality

I promise that the following things are created by me:

1. Every image used was drawn and generated by me.
2. Every sound used was created & recorded by me (except the background music).
3. Every 3D or 2D asset used was created by me
4. Every code is written by me (except the code in the reference)
5. All the text is original content generated by me.

## 2. Basic Details:

| | |
|---|---|
| Full name | Yifan Wu |
| Student number | 45203811 |
| Practical session | DECO7230 p02 |
| Tutor name | Alexander Balson (workshop)<br>Jason Weigel(practical) |
| Document title | Heal the world |

## 3. The Concept

### Link to a previous description of the concept：

https://ixd-deco2300.uqcloud.net/blogs/journal-week3-5d4d525a69e70

### One sentence “blurb”:

This strategy game uses the grid transformation to balance the resources of the old and new planets and stimulate people's environmental awareness.

### The background of this game:

The environment of the Earth is almost destroyed after 3000 years. Fortunately, people discovered a new planet called X-planet. Its environment is similar to Earth. People are planning to move to x-planet to relieve the destruction speed of the Earth.

## Then the mechanism of the game:

Players transmit the Earth's resources and population to planet-x, reducing the pressure on the Earth and giving it more time to recover. At the same time, players must ensure that the planet-x resources are not used up. The player must complete the operation within fifteen times.
When the user performs an addition operation, the data of population will be changed from two to seven, the data of population will be changed from one to five and the data of construction and pollution will be changed from one to three. The same algorithm is followed when the value is reduced.

**Conditions for the player to win:**
If players meet any conditions below, they will win the game.

- The population of the earth is lower than 21%;
- The resources of the earth are higher than 80%;
- The pollution of the earth is lower than 25%;
- The population of the x-planet is higher than 88%;
- The construction of the x-planet is higher than 54%;

**Conditions for player failure:**
If players meet any conditions below, they will lose the game.

- The population of the earth is higher than 99%;
- The resources of the earth are lower than 9%;
- The construction of the earth is lower than 26%;
- The pollution of the earth is higher than 56%;
- The population of the x-planet is lower than 7%;
- The resources of the x-planet are lower than 29%;
- The construction of the x-planet is lower than 19%;
- The pollution of x-planet is higher than 53%;

## The specific interaction process:

Firstly, players should enter the gate by the switch.
Secondly, they will enter the main interface and see the instruction.
Thirdly, they will enter the main interface.
The left planet is the Earth, and the right one is the x-planet. The number displayed in the red box is the number of remaining operations. They can input plus or minus instructions to transmit the resources between two planets.
Fourthly, players can restart the game by clicking the restart button.
All manipulations of users follow the rules in the mechanism above. The picture and the round status bar of these two planets will have different changes when the game wins or fails.

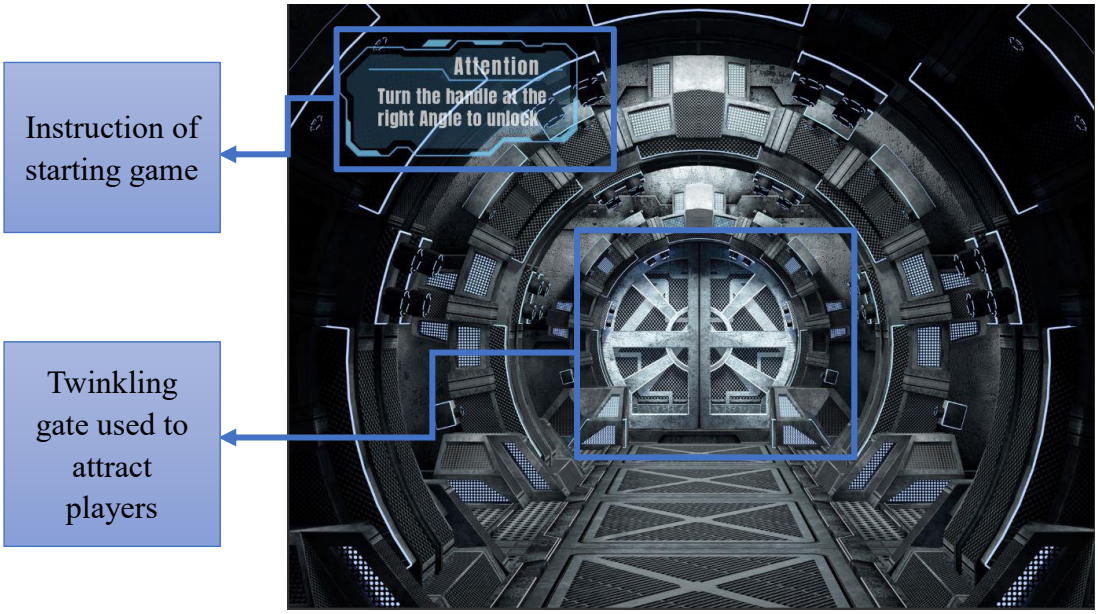



Figure 1 Initial interface

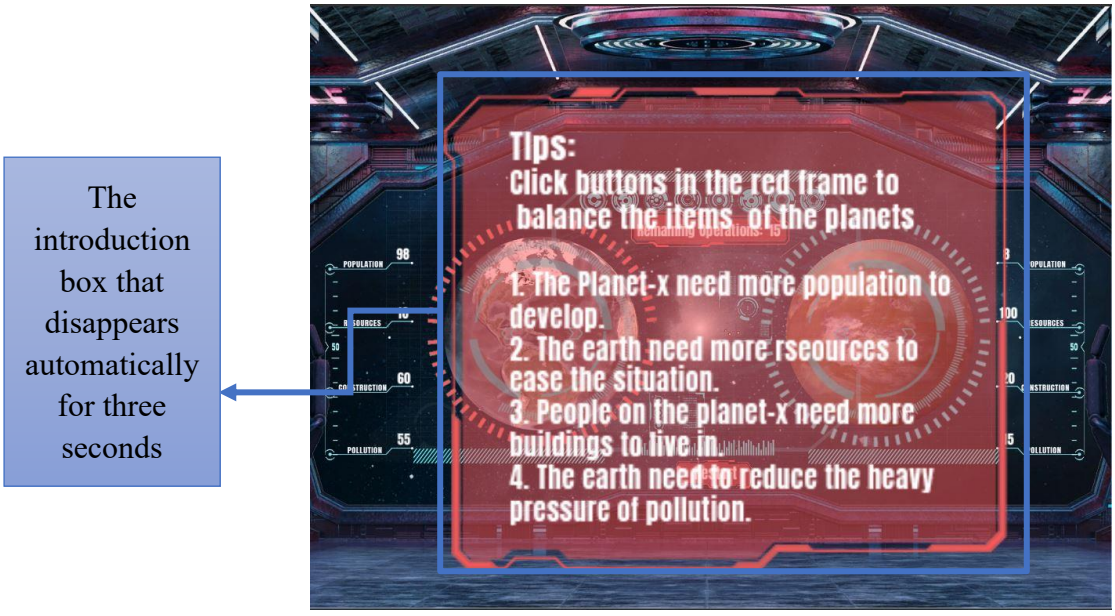



Figure 2 Tutorial

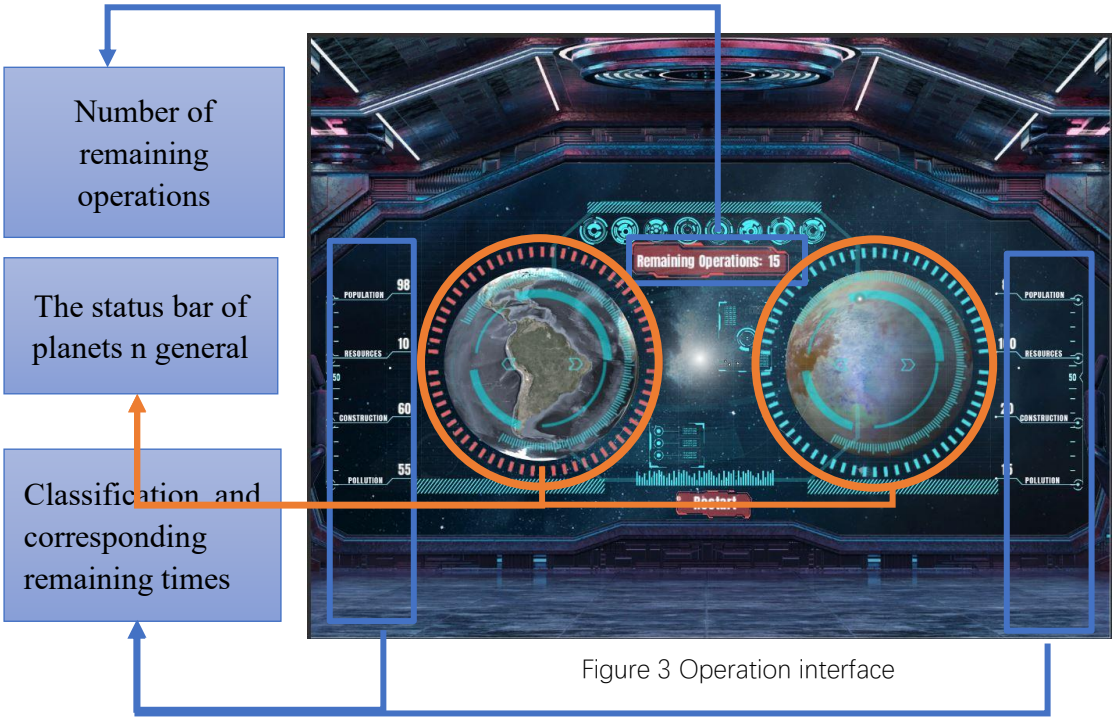



Figure 3 Operation interface

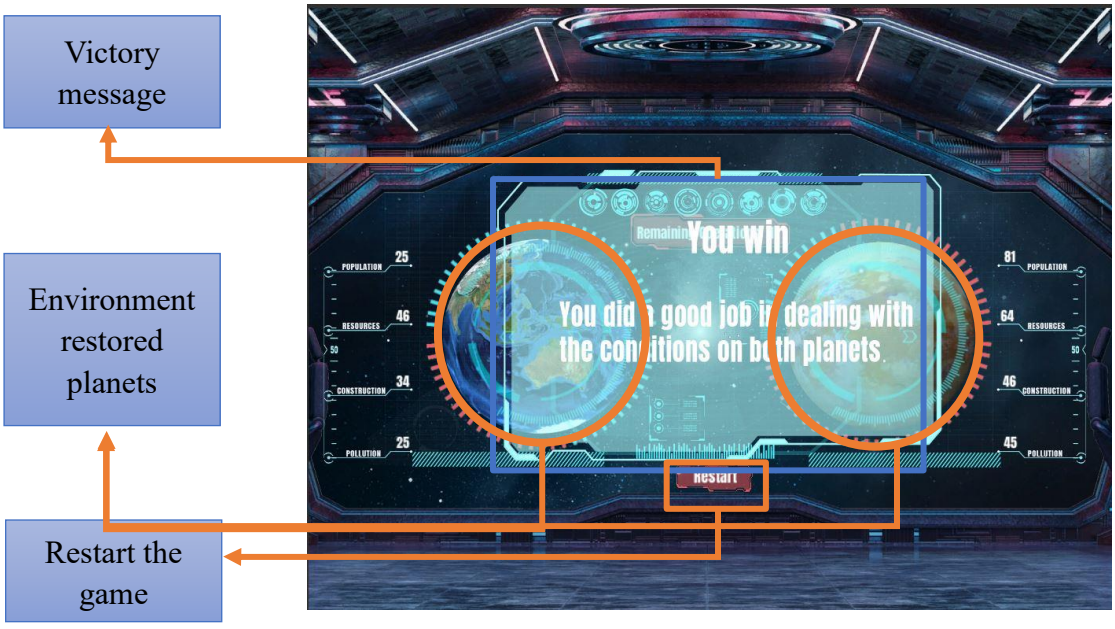



Figure 4 Victory interface

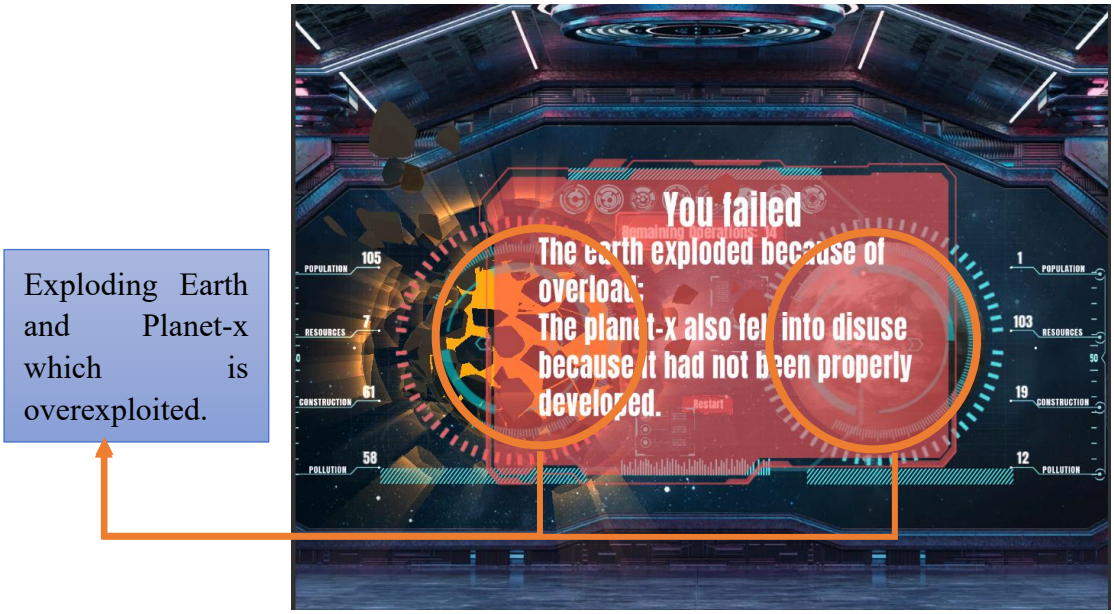



Figure 5 Failure interface

## 4. Recap

1. Firstly, I have made some fine-tuning of the game's **winning and losing standards**. For example, when the user wins, the population of the earth is lower than 21% rather than 18%. In addition, the player produces a certain range each time the click is made. These changes are in order to increase the player's playing time and reduce the difficulty of the game, which can make the game more playable.
2. Secondly, I used a **new starry sky map** as the background. The number of stars on this map is small and the brightness is very low. These are to **highlight** the subject object. [Tester2 Q4] [Tester4 Q5] [Tester4 A4]
3. Thirdly, I **removed all the plus and minus button** to simplify the interface [Tester1 Q5]
4. Fourthly, I **extended the instruction time** to let the player read longer [Tester1 Q10] and I added the game's **detailed** win-and-loss mechanism to let the player clearly understand the next goal [Tester2 Q10] [Tester3 Q10] [Tester4 Q10]
5. Fifthly, I added a **victory& failure interface** to inform the player of the end of the game and the game results [Tester3 Q5] [Tester3 A6]
6. Sixthly, I added round **status bar with vittual style** to let players know the changes that can be made in each step of their operation [Tester1 Q4] [Tester4 Q4]
7. Seventhly, I added the **restart button** to provide a shortcut for the player to play again.
8. Eighthly, I design a **big box to contain all of the Arduino elements** and use cubes to symbolize the resources which could be transmitted between planets (details in figure 22 The physical part of Arduino)

After integrating the feedback from testers, I made the current Interactive Prototype III and will test it again in the test below.

## 5. The Purpose of this testing round

### (1)A description of the aspects that are to be evaluated

Generally, this test is more detailed in the basic interface and the interaction flow than the last prototype. The content of the test mainly includes visual effects, function realization and physical interaction (operation of physical installment).

### (2) What hope to find out through people using the prototype

Firstly, it will test the playability of the game from the specific aspect that whether the rule of the game is clear enough.
Secondly, it will test whether the ending interface of the game is acceptable.
Thirdly, it will test if the data visualization is obvious such as the data changes show on the planet.
Fourthly, it will test users' attitudes about the mechanism of the physical installment.
Fifthly, it will test whether the game introduction is detailed and eye-catching enough.
Sixthly, it will test the interestingness and stability of the visual part.
Seventhly, it will test the fluency of the control process that users throw the paper cube into the tube by physical installment.

### (3) The items need to be included & excluded in the construction of the prototype

The appearance and the interaction way of the physical device should be concerned because the user uses the physical installment to operate in this version, but the playability in the interaction should still be the first thing to be considered.

# 6. The Form of the prototype

### (1) General Description

This prototype that I use to test my hypothesis is the Interactive Prototype II.
The software I use is Unity and Visual Studio. The physical interaction used is Arduino. The modeling part is done with Unity with the reference picture and model (see reference).

The player control element of the physical interaction is photoresistor and potentiometer. By uploading code from the Arduino application to the Arduino physical side and combining the code edited by the Visual Studio, the Unity and physical parts of Arduino will be linked to implementing the game interface via Arduino.

In terms of physical interaction, I design a big box to contain all of the Arduino elements such as the button which has the function of a switch, the potentiometer has the function of a door handle, the photosensitive resistor. The other elements of the installment are the paper cube which symbolizes one kind of resource that transmits between the planets and four operation tubes for users to throw cubes. The photoresistors are embedded in the inner wall of the tube. The three sides of the box facade are opened with holes that can be closed at any time to facilitate adjustment at any time.

### (2) Screenshots

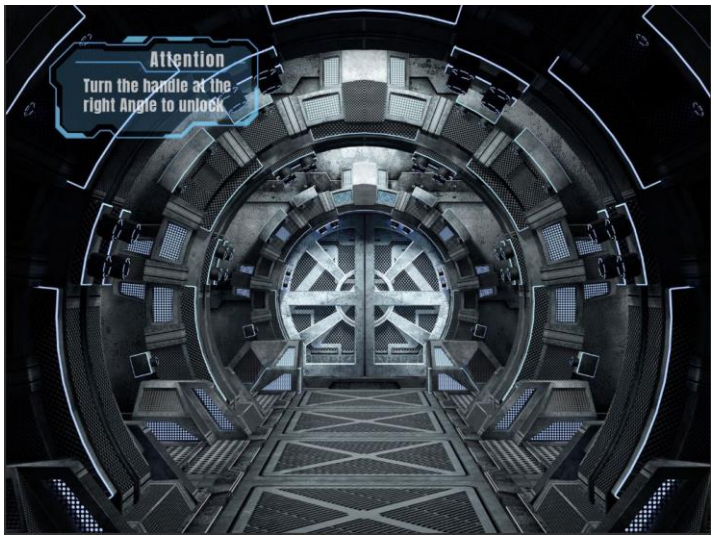



Figure 6 initial/restart interface

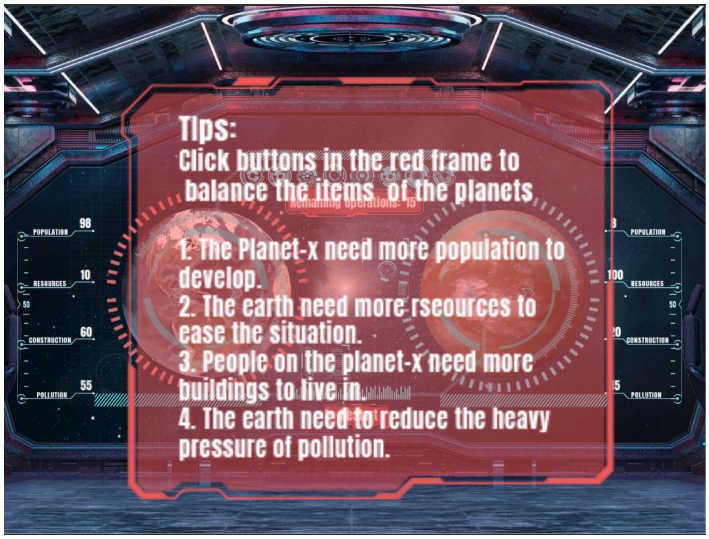



Figure 7 Instruction


Figure 8 Initial status


Figure 9 Victory interface

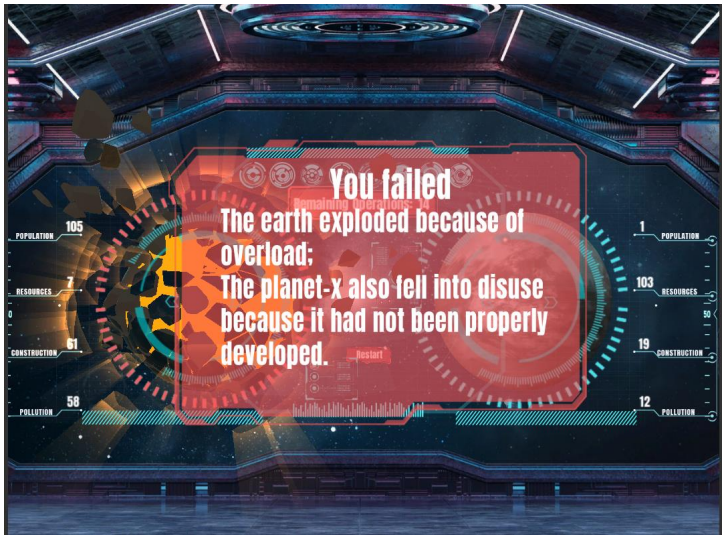
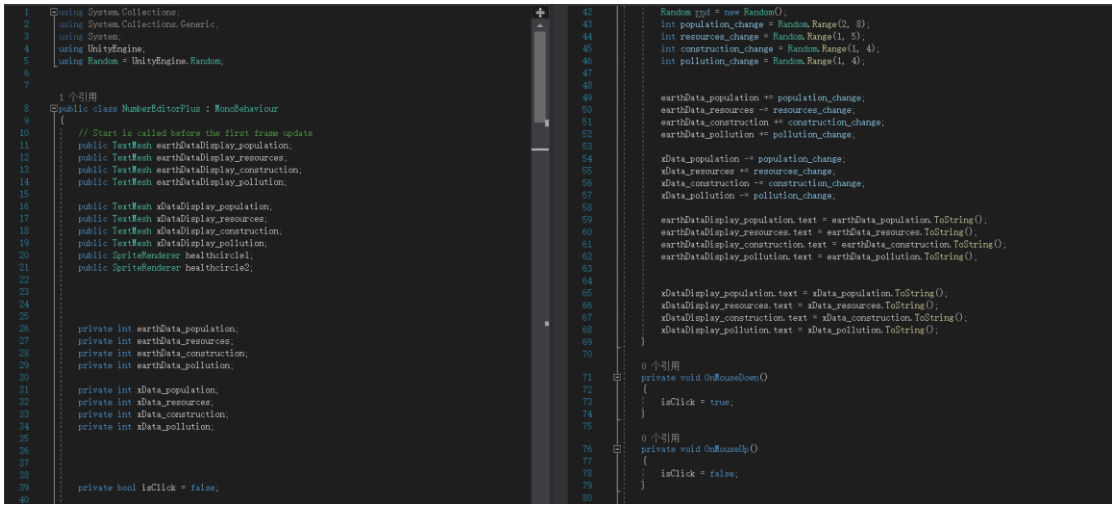



Figure 10 Failure interface

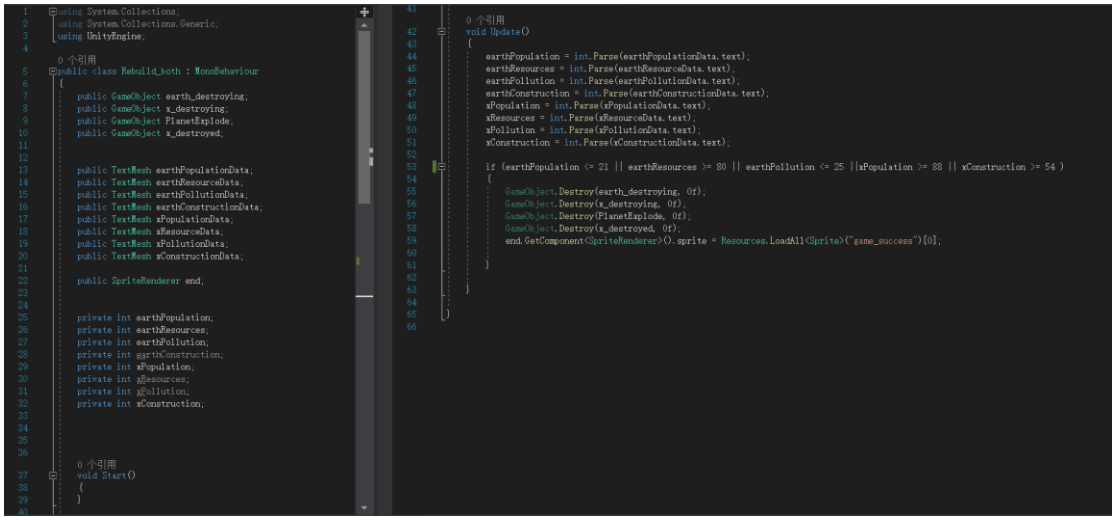
About above interfaces, I have explained in the specific interaction interface in the Concept part.

The main code of the game is in the bellow. They are the code of Arduino controller, the code of plus/minus buttons, the code of the remaining number, the code of making the gate disappear, the code of restarting the game and the code of changing planet depends on the outcome of the game. The hardest part is coping with the signal from the Arduino. (see Figure 21 code of Arduino Controller)



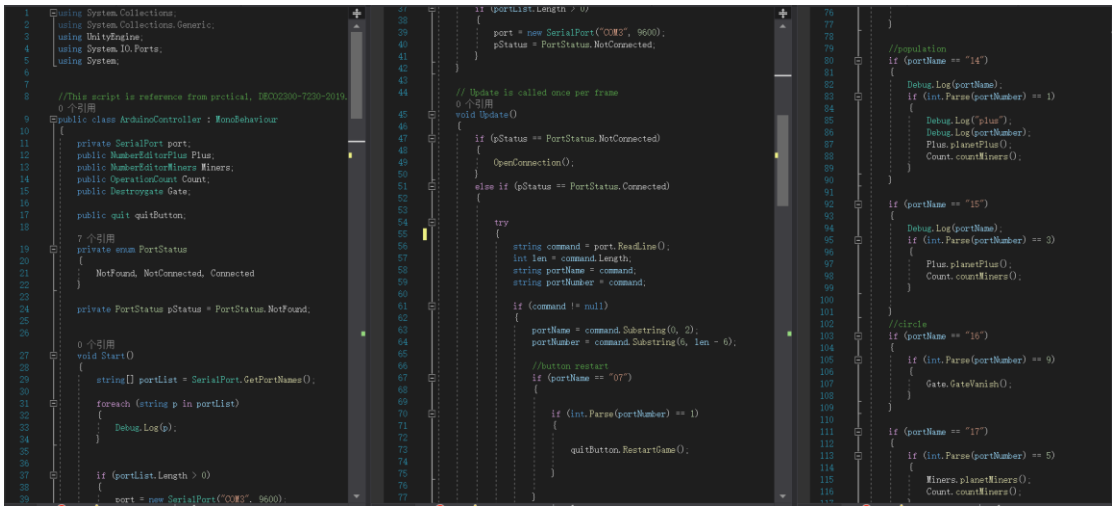
```
using System.Collections;
using System.Collections.Generic;
using System;
using UnityEngine;
using Random = UnityEngine.Random;


1 个引用
public class NumberEditorPlus : MonoBehaviour
{
    // Start is called before the first frame update
    public TextMesh earthDataDisplay_population;
    public TextMesh earthDataDisplay_resources;
    public TextMesh earthDataDisplay_construction;
    public TextMesh earthDataDisplay_pollution;

    public TextMesh xDataDisplay_population;
    public TextMesh xDataDisplay_resources;
    public TextMesh xDataDisplay_construction;
    public TextMesh xDataDisplay_pollution;
    public SpriteRenderer healthcircle1;
    public SpriteRenderer healthcircle2;




    private int earthData_population;
    private int earthData_resources;
    private int earthData_construction;
    private int earthData_pollution;

    private int xData_population;
    private int xData_resources;
    private int xData_construction;
    private int xData_pollution;




    private bool isClick = false;

    Random rnd = new Random();
    int population_change = Random.Range(2, 8);
    int resources_change = Random.Range(1, 5);
    int construction_change = Random.Range(1, 4);
    int pollution_change = Random.Range(1, 4);


    earthData_population += population_change;
    earthData_resources -= resources_change;
    earthData_construction += construction_change;
    earthData_pollution += pollution_change;

    xData_population -= population_change;
    xData_resources += resources_change;
    xData_construction -= construction_change;
    xData_pollution -= pollution_change;

    earthDataDisplay_population.text = earthData_population.ToString();
    earthDataDisplay_resources.text = earthData_resources.ToString();
    earthDataDisplay_construction.text = earthData_construction.ToString();
    earthDataDisplay_pollution.text = earthData_pollution.ToString();


    xDataDisplay_population.text = xData_population.ToString();
    xDataDisplay_resources.text = xData_resources.ToString();
    xDataDisplay_construction.text = xData_construction.ToString();
    xDataDisplay_pollution.text = xData_pollution.ToString();
}

0 个引用
private void OnMouseDown()
{
    isClick = true;
}

0 个引用
private void OnMouseUp()
{
    isClick = false;
}
```

Figure 11 The code of plus function



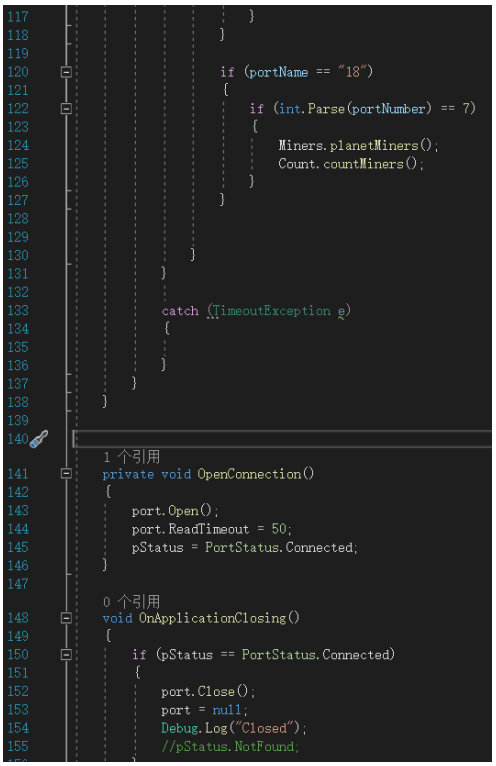
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

0 个引用
public class Rebuild_both : MonoBehaviour
{
    public GameObject earth_destroying;
    public GameObject x_destroying;
    public GameObject PlanetExplode;
    public GameObject x_destroyed;


    public TextMesh earthPopulationData;
    public TextMesh earthResourceData;
    public TextMesh earthPollutionData;
    public TextMesh earthConstructionData;
    public TextMesh xPopulationData;
    public TextMesh xResourceData;
    public TextMesh xPollutionData;
    public TextMesh xConstructionData;

    public SpriteRenderer end;


    private int earthPopulation;
    private int earthResources;
    private int earthPollution;
    private int earthConstruction;
    private int xPopulation;
    private int xResources;
    private int xPollution;
    private int xConstruction;




    0 个引用
    void Start()
    {
    }

    0 个引用
    void Update()
    {
        earthPopulation = int.Parse(earthPopulationData.text);
        earthResources = int.Parse(earthResourceData.text);
        earthPollution = int.Parse(earthPollutionData.text);
        earthConstruction = int.Parse(earthConstructionData.text);
        xPopulation = int.Parse(xPopulationData.text);
        xResources = int.Parse(xResourceData.text);
        xPollution = int.Parse(xPollutionData.text);
        xConstruction = int.Parse(xConstructionData.text);

        if (earthPopulation <= 21 || earthResources >= 80 || earthPollution <= 25 ||xPopulation >= 88 || xConstruction >= 54 )
        {
            GameObject.Destroy(earth_destroying, 0f);
            GameObject.Destroy(x_destroying, 0f);
            GameObject.Destroy(PlanetExplode, 0f);
            GameObject.Destroy(x_destroyed, 0f);
            end.GetComponent<SpriteRenderer>().sprite = Resources.LoadAll<Sprite>("game_success")[0];

        }
    }
}
```

Figure 13 The code of minus function



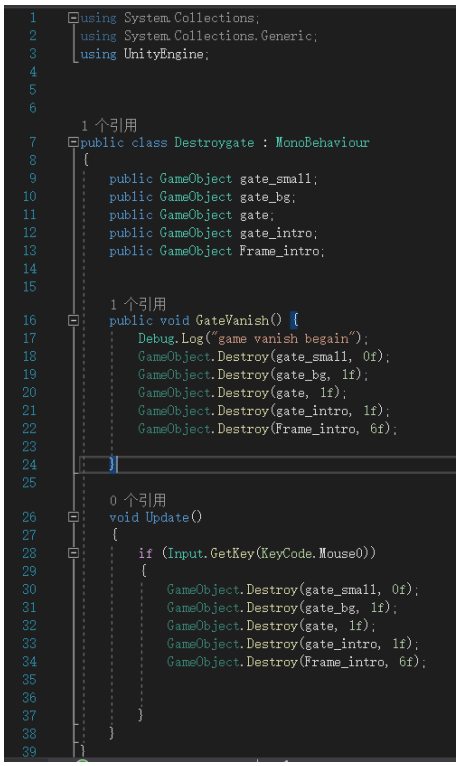
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using System.IO.Ports;
using System;


//This script is reference from prctical, DECO2300-7230-2019.
0 个引用
public class ArduinoController : MonoBehaviour
{
    private SerialPort port;
    public NumberEditorPlus Plus;
    public NumberEditorMiners Miners;
    public OperationCount Count;
    public Destroygate Gate;

    public quit quitButton;

    7 个引用
    private enum PortStatus
    {
        NotFound, NotConnected, Connected
    }

    private PortStatus pStatus = PortStatus.NotFound;


    0 个引用
    void Start()
    {
        string[] portList = SerialPort.GetPortNames();

        foreach (string p in portList)
        {
            Debug.Log(p);
        }


        if (portList.Length > 0)
        {
            port = new SerialPort("COM3", 9600);
            pStatus = PortStatus.NotConnected;
        }
    }

    // Update is called once per frame
    0 个引用
    void Update()
    {
        if (pStatus == PortStatus.NotConnected)
        {
            OpenConnection();
        }
        else if (pStatus == PortStatus.Connected)
        {

            try
            {
                string command = port.ReadLine();
                int len = command.Length;
                string portName = command;
                string portNumber = command;

                if (command != null)
                {
                    portName = command.Substring(0, 2);
                    portNumber = command.Substring(6, len - 6);

                    //button restart
                    if (portName == "07")
                    {

                        if (int.Parse(portNumber) == 1)
                        {

                            quitButton.RestartGame();

                        }
                    }


                    //population
                    if (portName == "14")
                    {
                        Debug.Log(portName);
                        if (int.Parse(portNumber) == 1)
                        {

                            Debug.Log("plus");
                            Debug.Log(portNumber);
                            Plus.planetPlus();
                            Count.countMiners();
                        }
                    }

                    if (portName == "15")
                    {
                        Debug.Log(portName);
                        if (int.Parse(portNumber) == 3)
                        {

                            Plus.planetPlus();
                            Count.countMiners();
                        }
                    }
                    //circle
                    if (portName == "16")
                    {
                        if (int.Parse(portNumber) == 9)
                        {

                            Gate.GateVanish();
                        }
                    }

                    if (portName == "17")
                    {
                        if (int.Parse(portNumber) == 5)
                        {

                            Miners.planetMiners();
                            Count.countMiners();
                        }
```

Figure 14 Arduino Controller1



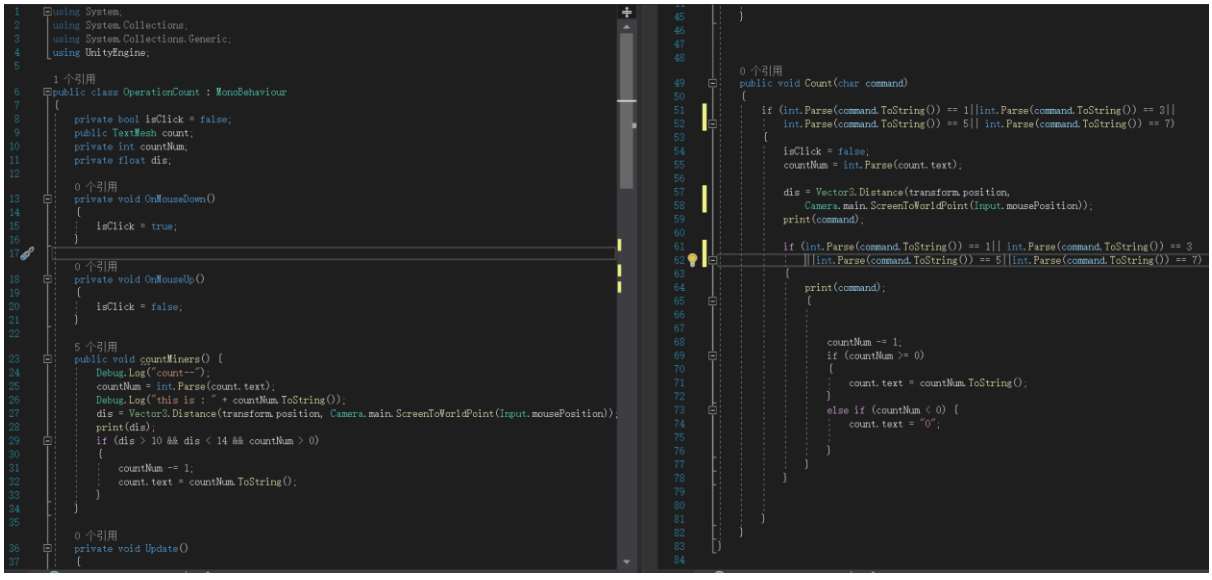
```
                    }
                }

                if (portName == "18")
                {
                    if (int.Parse(portNumber) == 7)
                    {
                        Miners.planetMiners();
                        Count.countMiners();
                    }
                }


            }
        }

        catch (TimeoutException e)
        {

        }
    }
}

1 个引用
private void OpenConnection()
{
    port.Open();
    port.ReadTimeout = 50;
    pStatus = PortStatus.Connected;
}

0 个引用
void OnApplicationClosing()
{
    if (pStatus == PortStatus.Connected)
    {
        port.Close();
        port = null;
        Debug.Log("Closed");
        //pStatus.NotFound;
```

Figure 14 Arduino Controller1



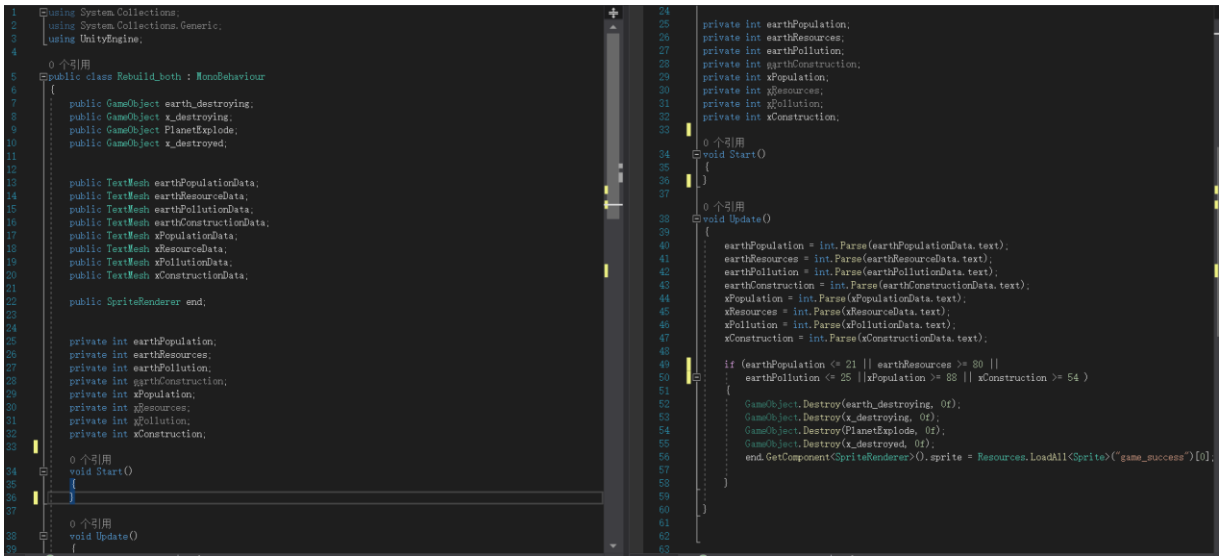
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;



1 个引用
public class Destroygate : MonoBehaviour
{
    public GameObject gate_small;
    public GameObject gate_bg;
    public GameObject gate;
    public GameObject gate_intro;
    public GameObject Frame_intro;


    1 个引用
    public void GateVanish() {
        Debug.Log("game vanish begain");
        GameObject.Destroy(gate_small, 0f);
        GameObject.Destroy(gate_bg, 1f);
        GameObject.Destroy(gate, 1f);
        GameObject.Destroy(gate_intro, 1f);
        GameObject.Destroy(Frame_intro, 6f);

    }

    0 个引用
    void Update()
    {
        if (Input.GetKey(KeyCode.Mouse0))
        {
            GameObject.Destroy(gate_small, 0f);
            GameObject.Destroy(gate_bg, 1f);
            GameObject.Destroy(gate, 1f);
            GameObject.Destroy(gate_intro, 1f);
            GameObject.Destroy(Frame_intro, 6f);


        }
    }
}
```

Figure 15 The code of making gate disappear



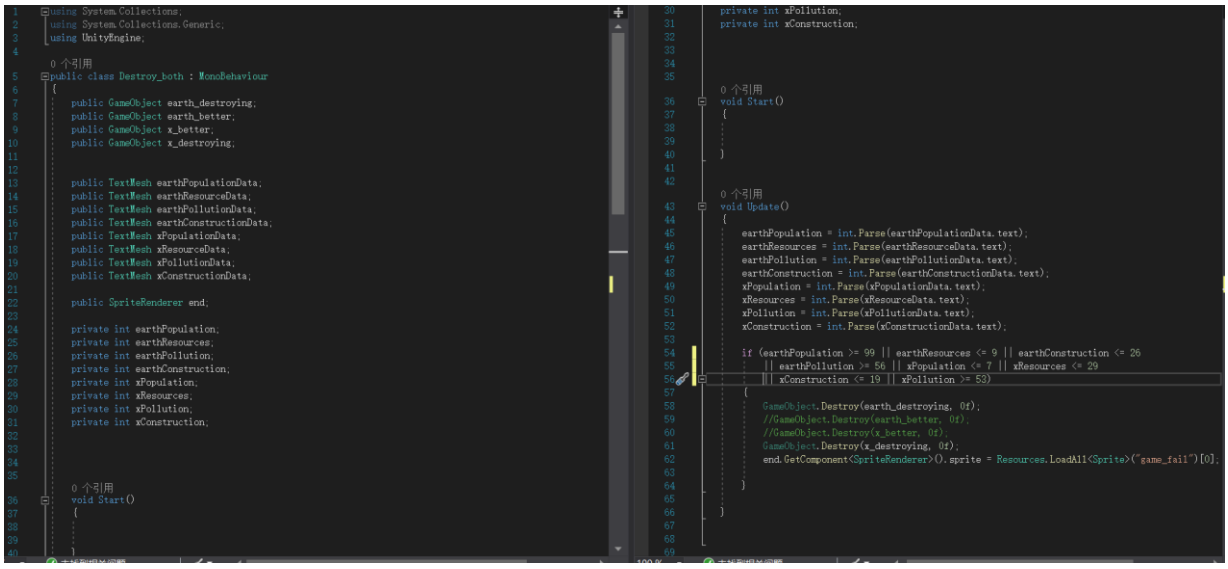
```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

1 个引用
public class OperationCount : MonoBehaviour
{
    private bool isClick = false;
    public TextMesh count;
    private int countNum;
    private float dis;

    0 个引用
    private void OnMouseDown()
    {
        isClick = true;
    }

    0 个引用
    private void OnMouseUp()
    {
        isClick = false;
    }

    5 个引用
    public void countMiners() {
        Debug.Log("count--");
        countNum = int.Parse(count.text);
        Debug.Log("this is : " + countNum.ToString());
        dis = Vector3.Distance(transform.position, Camera.main.ScreenToWorldPoint(Input.mousePosition));
        print(dis);
        if (dis > 10 && dis < 14 && countNum > 0)
        {
            countNum -= 1;
            count.text = countNum.ToString();
        }
    }

    0 个引用
    private void Update()
    {
    }


    0 个引用
    public void Count(char command)
    {
        if (int.Parse(command.ToString()) == 1||int.Parse(command.ToString()) == 3||
            int.Parse(command.ToString()) == 5|| int.Parse(command.ToString()) == 7)
        {
            isClick = false;
            countNum = int.Parse(count.text);

            dis = Vector3.Distance(transform.position,
                Camera.main.ScreenToWorldPoint(Input.mousePosition));
            print(command);

            if (int.Parse(command.ToString()) == 1|| int.Parse(command.ToString()) == 3
                ||int.Parse(command.ToString()) == 5||int.Parse(command.ToString()) == 7)
            {
                print(command);
                {


                    countNum -= 1;
                    if (countNum >= 0)
                    {
                        count.text = countNum.ToString();
                    }
                    else if (countNum < 0) {
                        count.text = "0";

                    }
                }
            }


        }
    }
}
```

Figure 16 the code of the remaining number



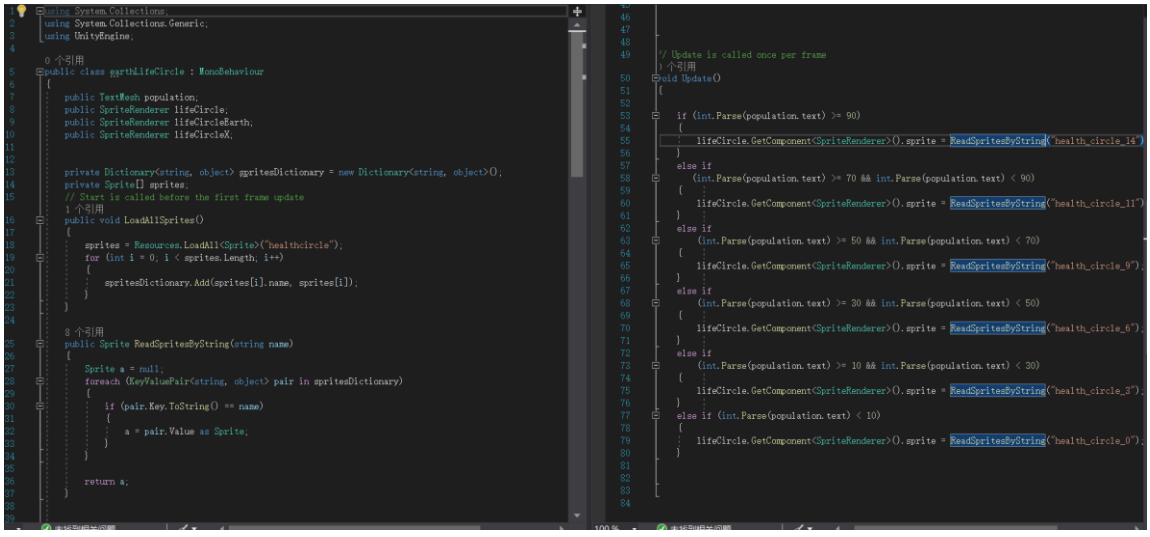
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

0 个引用
public class Rebuild_both : MonoBehaviour
{
    public GameObject earth_destroying;
    public GameObject x_destroying;
    public GameObject PlanetExplode;
    public GameObject x_destroyed;


    public TextMesh earthPopulationData;
    public TextMesh earthResourceData;
    public TextMesh earthPollutionData;
    public TextMesh earthConstructionData;
    public TextMesh xPopulationData;
    public TextMesh xResourceData;
    public TextMesh xPollutionData;
    public TextMesh xConstructionData;

    public SpriteRenderer end;


    private int earthPopulation;
    private int earthResources;
    private int earthPollution;
    private int earthConstruction;
    private int xPopulation;
    private int xResources;
    private int xPollution;
    private int xConstruction;

    0 个引用
    void Start()
    {
    }

    0 个引用
    void Update()
    {
        earthPopulation = int.Parse(earthPopulationData.text);
        earthResources = int.Parse(earthResourceData.text);
        earthPollution = int.Parse(earthPollutionData.text);
        earthConstruction = int.Parse(earthConstructionData.text);
        xPopulation = int.Parse(xPopulationData.text);
        xResources = int.Parse(xResourceData.text);
        xPollution = int.Parse(xPollutionData.text);
        xConstruction = int.Parse(xConstructionData.text);

        if (earthPopulation <= 21 || earthResources >= 80 ||
            earthPollution <= 25 ||xPopulation >= 88 || xConstruction >= 54 )
        {
            GameObject.Destroy(earth_destroying, 0f);
            GameObject.Destroy(x_destroying, 0f);
            GameObject.Destroy(PlanetExplode, 0f);
            GameObject.Destroy(x_destroyed, 0f);
            end.GetComponent<SpriteRenderer>().sprite = Resources.LoadAll<Sprite>("game_success")[0];
        }
    }
}
```

Figure 17 The code of changing planet-1



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

0 个引用
public class Destroy_both : MonoBehaviour
{
    public GameObject earth_destroying;
    public GameObject earth_better;
    public GameObject x_better;
    public GameObject x_destroying;


    public TextMesh earthPopulationData;
    public TextMesh earthResourceData;
    public TextMesh earthPollutionData;
    public TextMesh earthConstructionData;
    public TextMesh xPopulationData;
    public TextMesh xResourceData;
    public TextMesh xPollutionData;
    public TextMesh xConstructionData;

    public SpriteRenderer end;

    private int earthPopulation;
    private int earthResources;
    private int earthPollution;
    private int earthConstruction;
    private int xPopulation;
    private int xResources;
    private int xPollution;
    private int xConstruction;




    0 个引用
    void Start()
    {

    }


    0 个引用
    void Update()
    {
        earthPopulation = int.Parse(earthPopulationData.text);
        earthResources = int.Parse(earthResourceData.text);
        earthPollution = int.Parse(earthPollutionData.text);
        earthConstruction = int.Parse(earthConstructionData.text);
        xPopulation = int.Parse(xPopulationData.text);
        xResources = int.Parse(xResourceData.text);
        xPollution = int.Parse(xPollutionData.text);
        xConstruction = int.Parse(xConstructionData.text);

        if (earthPopulation >= 99 || earthResources <= 9 || earthConstruction <= 26
            || earthPollution >= 56 || xPopulation <= 7 || xResources <= 29
            || xConstruction <= 19 || xPollution >= 59)
        {
            GameObject.Destroy(earth_destroying, 0f);
            //GameObject.Destroy(earth_better, 0f);
            //GameObject.Destroy(x_better, 0f);
            GameObject.Destroy(x_destroying, 0f);
            end.GetComponent<SpriteRenderer>().sprite = Resources.LoadAll<Sprite>("game_fail")[0];

        }
    }
}
```

Figure 18 The code of changing planet-2



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

0 个引用
public class earthLifeCircle : MonoBehaviour
{
    public TextMesh population;
    public SpriteRenderer lifeCircle;
    public SpriteRenderer lifeCircleEarth;
    public SpriteRenderer lifeCircleX;


    private Dictionary<string, object> spritesDictionary = new Dictionary<string, object>();
    private Sprite[] sprites;
    // Start is called before the first frame update
    1 个引用
    public void LoadAllSprites()
    {
        sprites = Resources.LoadAll<Sprite>("healthcircle");
        for (int i = 0; i < sprites.Length; i++)
        {
            spritesDictionary.Add(sprites[i].name, sprites[i]);
        }
    }

    8 个引用
    public Sprite ReadSpritesByString(string name)
    {
        Sprite a = null;
        foreach (KeyValuePair<string, object> pair in spritesDictionary)
        {
            if (pair.Key.ToString() == name)
            {
                a = pair.Value as Sprite;
            }
        }

        return a;
    }


// Update is called once per frame
1 个引用
void Update()
{

    if (int.Parse(population.text) >= 90)
    {
        lifeCircle.GetComponent<SpriteRenderer>().sprite = ReadSpritesByString("health_circle_14");
    }
    else if
        (int.Parse(population.text) >= 70 && int.Parse(population.text) < 90)
    {
        lifeCircle.GetComponent<SpriteRenderer>().sprite = ReadSpritesByString("health_circle_11");
    }
    else if
        (int.Parse(population.text) >= 50 && int.Parse(population.text) < 70)
    {
        lifeCircle.GetComponent<SpriteRenderer>().sprite = ReadSpritesByString("health_circle_9");
    }
    else if
        (int.Parse(population.text) >= 30 && int.Parse(population.text) < 50)
    {
        lifeCircle.GetComponent<SpriteRenderer>().sprite = ReadSpritesByString("health_circle_6");
    }
    else if
        (int.Parse(population.text) >= 10 && int.Parse(population.text) < 30)
    {
        lifeCircle.GetComponent<SpriteRenderer>().sprite = ReadSpritesByString("health_circle_3");
    }
    else if (int.Parse(population.text) < 10)
    {
        lifeCircle.GetComponent<SpriteRenderer>().sprite = ReadSpritesByString("health_circle_0");
    }
}
```

Figure 19 The code of round status bar

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

1 个引用
public class quit : MonoBehaviour
{
    public GameObject Button;

    1 个引用
    public void RestartGame()
    {
        SceneManager.LoadScene(SceneManager.GetActiveScene().name); // loads current scene
    }
}
```

Figure 20 The code of restarting game

```
const int photocell_1 = A0;
int photocell_1Read = 0;
int photocell_1Data = 0;

int photocell_2 = A1;
int photocell_2Read = 0;
int photocell_2Data = 0;

int photocell_3 = A3;
int photocell_3Read = 0;
int photocell_3Data = 0;

int photocell_4 = A4;
int photocell_4Read = 0;
int photocell_4Data = 0;

int sensor = A2;
int sensorRead = 0;
int sensorData = 0;

int pushButton =7;
int buttonState =0;
//int beforeState =0;
//int pressTime =0;


//This script is reference fr
class Photocell {
  int pin;
```

```
int prevReading = 500;

byte messageToSendUp;
byte messageToSendDown;


public:
  void init(int pinNum, byte messageUp, byte messageDown) {
  this->pin = pinNum;
    pinMode(this->pin, INPUT);
    this->messageToSendUp = messageUp;
    this->messageToSendDown = messageDown;
  }

  void loop() {
  int reading = analogRead(this->pin);

    if (reading >= this->prevReading + 100) {
      this->prevReading = reading;
      delay(200);
      Serial.print(pin);
      Serial.print(" is ");
      Serial.println(this->messageToSendUp);
    } else if (reading <= this->prevReading - 100) {
      this->prevReading = reading;
      delay(200);
      Serial.print(pin);
      Serial.print(" is ");
      Serial.println(this->messageToSendDown);
    }
```

```
  }

};
Photocell Photocell_1;
Photocell Photocell_2;
Photocell Photocell_3;
Photocell Photocell_4;

void setup(){
  Serial.begin(9600);
  pinMode(pushButton, INPUT);
  Photocell_1.init(A0, 0, 1);
  Photocell_2.init(A1, 2, 3);
  Photocell_3.init(A3, 4, 5);
  Photocell_4.init(A4, 6, 7);
  }

void loop(){
  photocell_1Read = analogRead(photocell_1);
  Serial.print(photocell_1);
  Serial.print(" is ");
  Serial.println(photocell_1Read);

  photocell_2Read = analogRead(photocell_2);
  Serial.print(photocell_2);
  Serial.print(" is ");
  Serial.println(photocell_2Read);

  photocell_3Read = analogRead(photocell_3);
  Serial.print(photocell_3);
```

```
photocell_3Read = analogRead(photocell_3);
Serial.print(photocell_3);
Serial.print(" is ");
Serial.println(photocell_3Read);

photocell_4Read = analogRead(photocell_4);
Serial.print(photocell_4);
Serial.print(" is ");
Serial.println(photocell_4Read);


sensorRead = analogRead(sensor);
sensorData = map(sensorRead, 0, 1023, 8, 9);
Serial.print(sensor);
Serial.print(" is ");
Serial.println(sensorData);

int buttonState = digitalRead(pushButton);
Serial.print(0);
Serial.print(pushButton);
Serial.print(" is ");
Serial.println(buttonState);
delay(100);

Photocell_1.loop();
Photocell_2.loop();
Photocell_3.loop();
Photocell_4.loop();
}
```

Figure 21 The code in Arduino

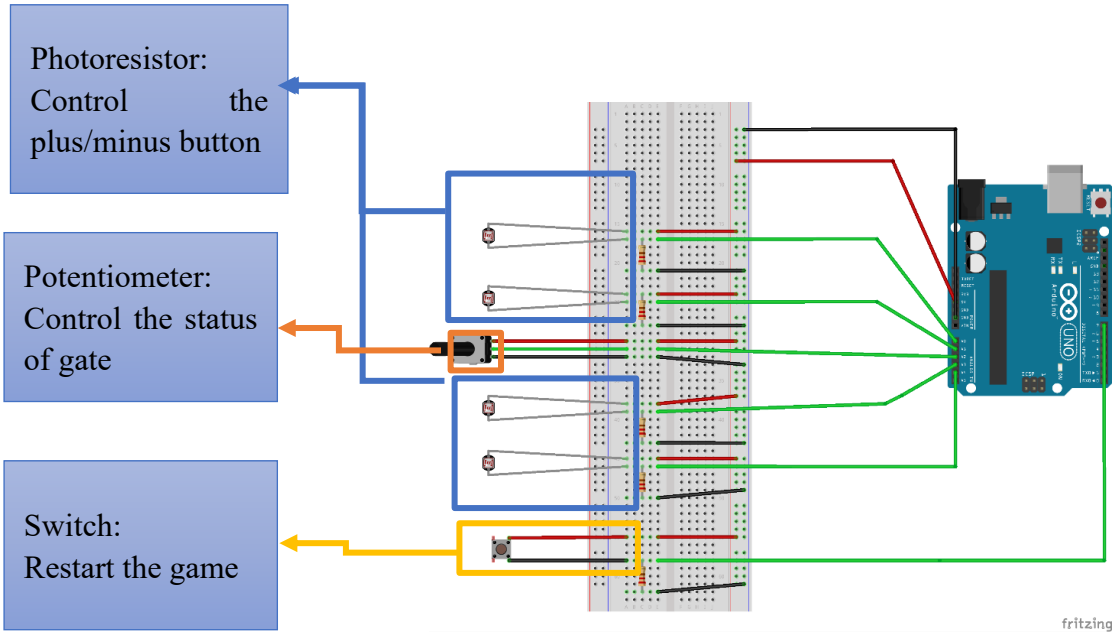



Figure 22 The physical part of Arduino

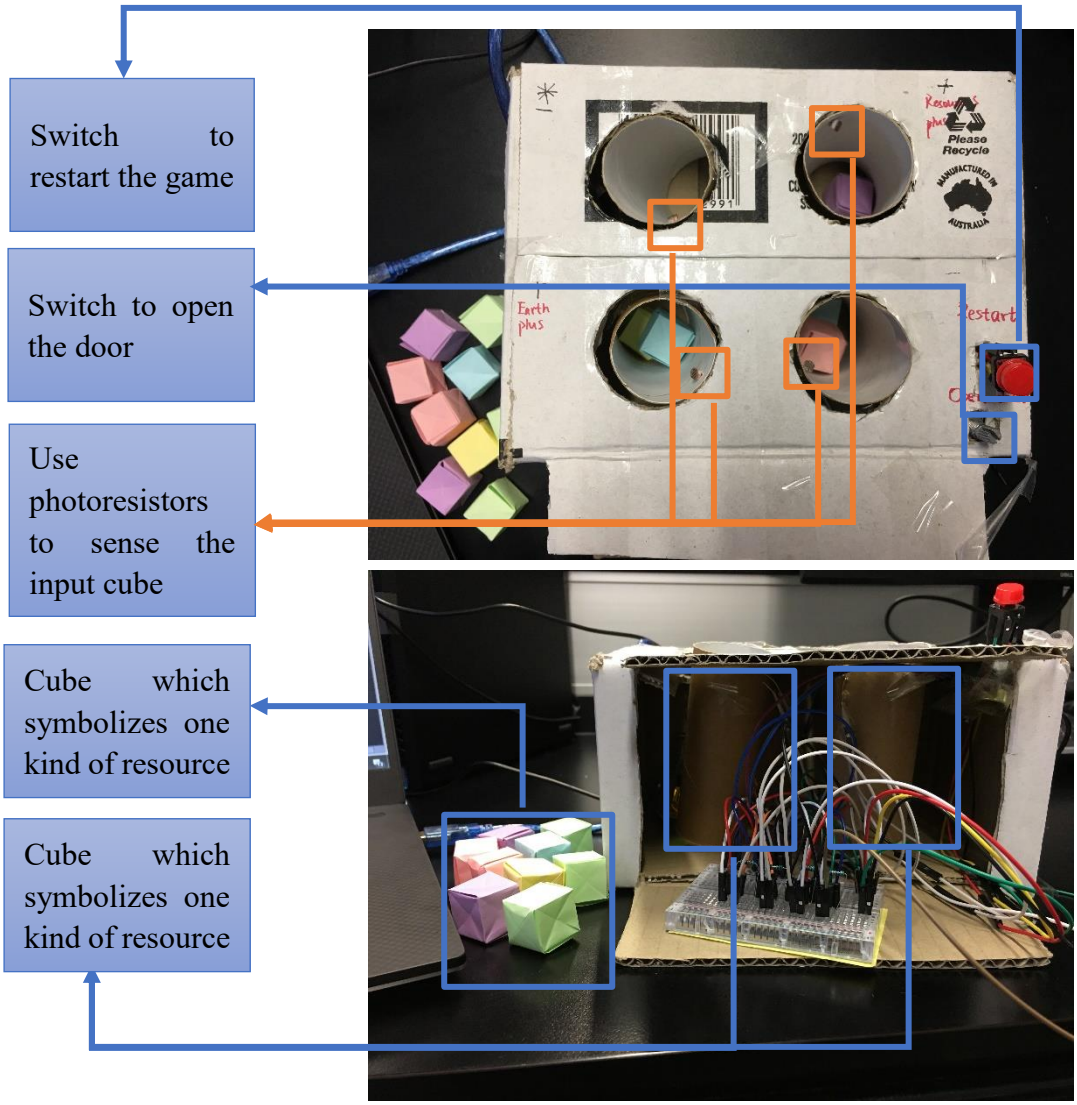



Figure 23&24 The physical part of Arduino

**(3) Interaction diagrams**

The first step is to read the instruction and open the gate by screwing the potentiometer. (step1 in the interaction flow).

The second step is to read the detailed instruction and set the number of resources to be transferred by touching the photoresistor. (step2 in the interaction flow).

The third step is to observe the remaining number of times in the center and the round status bar around the planets to realize their real-time progress in the game (step3 in the interaction flow).

After repeating this step fifteen times, players can judge their game results through changes in planets (step4 in the interaction flow).

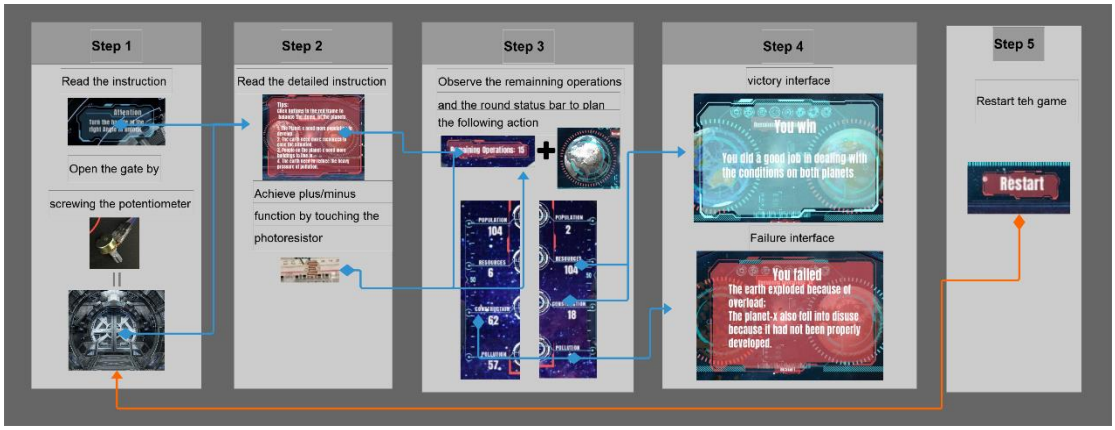



Figure 125 Interaction flow

# 7. Testing Approach

## The evaluation method

In order to connect the test before and after and use the evaluation method in front, I would keep using the methods of observation and interview in prototype III.

I use an interview in this phase because the interview can communicate more directly, which is conducive to building trust with the tester.

Due to the complexity of the prototype, the player's operating ratio is more complicated before, so that the information obtained from the observation will produce some errors. For example, the operation of the physical installment could be influenced by some other factors such as the repeated press. It could be device delay, improper operation or the user has the habit of repeated pressing.

However, the data got by observation can intuitively reflect the user's problem, so it is more reliable. By setting the accurate observation goals in advance, the observation can yield a lot of useful information such as tester preference and the place that cannot run smooth.

In terms of the questionnaire, it is not detailed enough to contain the information of this prototype. In addition. it is difficult for the investigator to understand whether the tester is filling out or being perfunctory. Secondly, the tester may be affected by the opinions of most people, which may affect the authenticity of the investigation. Thirdly, people are usually not willing to fill in open-ended questions or be perfunctory, so there will be a shortage of information in this area.

Therefore, Interviews are the primary method of testing, and observation is used to aid analysis and let users know some basic information to save more time for the interview. By getting detailed enough information about personal opinions, and any misunderstanding can be easily corrected.

Also, the fully exchanged opinions can correct the outcome get by the observation before and the higher response rate make it more efficient.

## Testing agenda

### Observation

1. Observe the speed at which the user understands the physical installment.
2. Observe the throwing distance and the success rate of operation of users.
3. Observe whether the user notices the principle of the prototype.
4. Observe the interval of user operations.
5. Observe how often the user restarts.
6. Observe the interval and times of pressing buttons.

### Interview

Questions that may be involved:

1. Are you clear with the rule after starting the game?
2. How do you evaluate the success/failure interface of the game?
3. What do you think about the mechanism of the prototype?
4. What is the most attractive item to you / what makes you confused?
5. What do you think about the features in the game? What do you think is the place that affects your gaming experience?
6. What do you think of the way you open the door?
7. Can you find the corresponding photoresistor accurately? Is it sensitive/slow enough?
8. Do you think the prototype interface is easy?
9. Do you think game instruction is helpful?
10. If you divide the game playability into 1 to 5 stars, which one would you choose?

| Tester number | Question number | Answer |
|---|---|---|
| Tester1 | Q1 | |
| | Q2 | |
| | Q3 | |
| | Q4 | |
| | Q5 | |
| | Q6 | |
| | Q7 | |
| | Q8 | |
| | Q9 | |
| | Q10 | |

## Testing Process

1. Let the tester play the game freely.
2. Observe the testers according to the points in the observation.
2. Communicate with the user (communication content contains prepared questions)
3. The tester gives an opinion and exchanges views based on the design method.
4. Record the above content and form a written document.

## Evaluation Method:

1. The method of conducting the evaluation

I have designed some questions to do observation and interview.
During the process of observation, I will recommend testers to experience the game freely and do the observation at the same time by the established plan.
When I do the interview, I can not only cover the prepared questions but also the questions derived from the observation. In addition, I will concentrate more on their experience and their expression because such face-to-face and free communication is the advantage of this method.

2. Description of the prototype

This prototype allows players to control the transfer of resources between two planets via the plus/minus button. All operations must be completed within fifteen times, and then the player's win or loss will be determined based on the data of the four aspects of the two planets. The key point of this prototype is to balance the resources between the two planets, which is also the most challenging part of the game. There are also some traps in the game to improve the difficulty of the game. For example, if the player clicks on the earth's population at the beginning of the game, then the game will end soon. This setting allows the player to understand more deeply about the background of the game (earth population explosion) Through this simple strategy arrangement, this game provides players with a moderately difficult play experience.

In terms of Arduino, players control the game by touching photoresistors, screwing the potentiometer and pressing the button. Practical interaction is achieved by capturing the signals of Arduino, transforming them into corresponding code and using different grammatical structures.

All of the Arduino elements are contained in a box such as a button which has the function of a switch, the potentiometer has the function of a door handle, the photosensitive resistor. The other elements of the installment are cubes and four operation tubes for users to put in.

3. How it achieves the evaluation outcomes

After experiencing the game, the tester got the complete information of the prototype and I could get some initial conclusion from the observation. I will ask something that I am confused about within the observation by interview. By the combination of these two evaluation methods, I can Ensure fast and accurate feedback from users.

# 8. Evaluation Outcomes & Reflection

## The results of the evaluation:

It can be seen from the evaluation results that this prototype has corrected some problems in the previous prototype. For example, the background of the game is no longer conspicuous, the button is removed to highlight the subject, the player will have feedback every time the operation, the time of instruction has increased and there will be a prompt interface at the end of the game. However, some problems still exist.

Firstly, there are some problems with the design of the physical installment.
The **variable resistors** should be **supported** by something, or else it is not well controlled.
There are **delays** and **false touches** in the **photoresistor**.
The **box** needs to be **bigger** to prevent some Arduino elements to go out of the box.
There is a **description** of the corresponding position of **each element** in the prototype.
The automatic **cube collection function** should be added.
The **plus and minus** functions should be realized by **one tube**.

In addition, there are some problems with the game function.
The **background** of the game should be shown before entering the door.
The **rebirth point** should be placed on the **inside** of the door.
**More time** are needed to read the **instruction**.

## The raw data of the evaluation:

**Observation**

| Tester number | Action number | Description |
|---|---|---|
| Tester1 | A1 | Spent 30 seconds to try all elements. |
| | A2 | Put cubes far away from the tube and the success rate is about 1/10. |
| | A3 | Notice the photoresistor immediately. |
| | A4 | Within 5 seconds. |
| | A5 | Restart 1 time and then success |
| | A6 | Nearly no interval and success in 1 time. |
| Tester2 | A1 | Spent 5 minutes to try all elements. |
| | A2 | Put cubes just on the surface of the photoresistor and the success rate is 100%. |
| | A3 | Have not notice the photoresistor till the end of the game. |

| | | |
|---|---|---|
| | A4 | About 10 seconds. |
| | A5 | Restart 3 time and then success. |
| | A6 | Keep pressing and success in 5 times. |
| Tester3 | A1 | Spent 1 minutes to try all elements. |
| | A2 | Put cubes close to the tube and the success rate is about 1/3. |
| | A3 | Notice the photoresistor after putting in 2 tubes. |
| | A4 | Within 2 seconds. |
| | A5 | No restart and success in the first time. |
| | A6 | Three seconds interval and success immediately. |
| Tester4 | A1 | Start the game in 10 seconds without figure out the function of every elements. |
| | A2 | Put cubes far away from the tube and the success rate is about 1/8. |
| | A3 | Notice the photoresistor immediately. |
| | A4 | Within 2 seconds. |
| | A5 | Restart 5 time and still fail. |
| | A6 | No interval and success in 6 times. |

**Interview**

| Tester number | Question number | Answer |
|---|---|---|
| Tester1 | Q1 | Quite clear, I can remember them. |
| | Q2 | Clear enough. |
| | Q3 | The device is very interesting, the method of using the device is unique. |
| | Q4 | The way the photoresistor is installed is very interesting. I didn't expect it to be used like this. |
| | Q5 | There is a delay in the photoresistor. |
| | Q6 | The current element is reminiscent of the door handle in the context of the installment, which is very creative. |
| | Q7 | There is no description on the installment interface so I can just try one by one. The reaction is slightly slower. |
| | Q8 | When I was throwing paper cubes, the arm will block the photosensitive resistor and cause mis-touch, and the control is unstable. |
| | Q9 | It is useful enough to introduce the rules of the game. |
| | Q10 | Prototype is very interesting, but maybe due to the design of the installment, I mean the tube, the game progress is not smooth, I give 4 stars. |
| Tester2 | Q1 | The time of the instruction in the game is still too short. |

| | | |
|---|---|---|
| | Q2 | Timely and eye-catching enough for me, especially the big "failure". |
| | Q3 | It would be easier if the addition and subtraction were achieved through a single cylinder. |
| | Q4 | The game's planet status bar is very interesting. |
| | Q5 | There should have something to support the switch, or it cannot be controlled easily. |
| | Q6 | The form of turning back and forth let me think of opening a safe, a bit mysterious and curious. |
| | Q7 | After several attempts, the corresponding photoresistors were found because they were not arranged according to the rules. The response is within my acceptance. |
| | Q8 | The entrances of several inputs are too close, and it is easy to block the effect of the photosensitive resistor, and the box is too small for some elements to be revealed. |
| | Q9 | The instruction needs to stay longer, because although I finish reading it, I will forget it during the game. |
| | Q10 | 4 stars, the function is basically realized, so it is satisfactory. |
| Tester3 | Q1 | I think it is clear by following the instruction. |
| | Q2 | I can get the ending message of the game immediately by the pop-up window. |
| | Q3 | Very interesting and creative. |
| | Q4 | The planet of the game is very eye-catching. |
| | Q5 | I don't know the background of the game before entering the door, it's a bit confusing; the prototype interface should be explained. |
| | Q6 | Very creative. |
| | Q7 | Need to try; The reaction is OK. |
| | Q8 | It should have a description of the corresponding elements of each input port in the physical part. |
| | Q9 | The rules just disappeared when I finished reading, just enough for me. |
| | Q10 | 4 stars, strategy games are not challenging for me, but I think the difficulty is enough for most people. |
| Tester4 | Q1 | I need the instruction always be there. |
| | Q2 | Very timely, although I have not seen a successful interface. |
| | Q3 | Quite cool. |
| | Q4 | The description of the game is very conspicuous. |
| | Q5 | Restart will be better if it can be set behind the door. |
| | Q6 | The idea is very good, but it is inconvenient to operate it again after each restart. |

| | | |
|---|---|---|
| | Q7 | I haven't found out until the game is finished. The response is not sensitive. |
| | Q8 | It is inconvenient to take out the squares, it would be more convenient if the cubes can be collected automatically. |
| | Q9 | The instruction needs to stay at the top of the interface because it is too complicated for me to remember. |
| | Q10 | 3 stars, although I like the interface of the game, the difficulty of the game is too high. |

## The changes will make in response to the evaluation:

From the evaluation above, the instruction should not disappear and set on the top of the interface. In this way, it will help the player and would not obstruct the other functions [Tester2 Q1] [Tester4 Q1] [Tester2 Q9] [Tester4 Q9].

Secondly, the rebirth point should be placed on the inside of the door so that it will not be “inconvenient to operate it again after each restart"[Tester4 Q6] [Tester4 Q5].

Thirdly, some tester reflects that the background of the game should be shown before entering the door. [Tester3 Q5] In fact, I hope to create a sense of mystery through this form, causing the curiosity of the players. Players can only understand the truth by cracking the password (turning the variable resistor). Therefore, I would keep the setting.

Fourthly, there could have some changes to the variable resistor. It should be supported by something. [Tester3 Q5] In fact, I used tape to secure it above the box, but the part was loose due to too many testing. But this still proves that the firmness is still insufficient, so it should be glued to the box firmly, and the bottom is supported by wooden strips.

Fifthly, someone wants me to simplify the interface of the installment. The plus and minus functions should be realized by one tube [Tester1 Q3]. I have tried to superimpose the translucent plastic sheet above the photoresistor and got the figure that the minimum difference between the two transparent ones is 20. When the plastic sheet is placed up, the transmittance is reduced by 20, and the reduction of the plastic sheet is the opposite. Although the idea is very good, the code is difficult to achieve, so I will keep this design.

Sixthly, the automatic cube collection function should be added [Tester4 Q8]. This function is limited by the size of the box and this issue is realized by one tester [Tester2 Q8]. I plan to put all the elements in a bigger box so that I can add an internet pocket at the bottom of each tube and bring it to a common container so that players can quickly collect paper cubes when they restart the game.

Seventhly, a description of the corresponding position of each element in the installment should be added.[Tester3 Q8] At the end of the setting, each tube is arranged in the order of the stars, but due to the bumps on the road, the corresponding line in the Arduino is broken, and the order of the photoresistors in the tube is disordered when it is connected again. I will draw the picture corresponding to earth and planet-x on the box, and the corresponding elements such as the switch (variable resistance), the restart button and the plus/minus function of the photoresistor. More importantly, the wire should also be attached to the inside of the box or the tube to increase its stability.

Eighthly, there are delays and false touches in the photoresistor.
[Tester1 Q8] [Tester2 Q8] This is related to the lack of light in the location I tested. I think it can be improved by adding a fixed light source. In addition, I plan to pack the parts in a large box and have all the tubes in a row, so you don't have to worry about blocking the light during operation.

**Reference of Interactive Prototype III:**

Images Used: Sky Photography, Vadim Sadovski; retrieved from shutterstock.com (https://www.shutterstock.com/zh/image-photo/cosmic-landscape-beautiful-science-fiction-wallpaper-783052066) Last Accessed 9/9/2019

Images Used: Set of planet earth, Twin Design; retrieved from shutterstock.com(https://www.shutterstock.com/zh/image-photo/set-planet-earth-globe-different-angles-134839019) Last Accessed 25/9/2019

Images Used: Radar screen, Uladzislau Baryschchyk; retrieved from shutterstock.com(https://www.shutterstock.com/zh/image-vector/radar-screen-vector-illustration-your-design-589828655) Last Accessed 25/9/2019

Images Used: Spaceship Interior Round. 3D illustration, Apex Infinity Games; retrieved from shutterstock.com(https://www.shutterstock.com/zh/image-illustration/spaceship-interior-round-3d-illustration-344364185) Last Accessed 25/9/2019

Images Used: Set of Sci Fi Modern User Interface Elements, Oniks Astarit; retrieved from shutterstock.com(https://www.shutterstock.com/zh/image-vector/set-sci-fi-modern-user-interface-1432399046) Last Accessed 25/9/2019

Images Used: Futuristic virtual graphic touch user interface, PALERMO89; retrieved from shutterstock.com (https://www.shutterstock.com/zh/image-vector/futuristic-virtual-graphic-touch-user-interface-661354531)Last Accessed 25/9/2019

Images Used: HUD ELEMENT PACK 1, firmanadhib; retrieved from shutterstock.com (https://www.shutterstock.com/zh/image-vector/hud-element-pack-1-338431427)Last Accessed 25/9/2019

Images Used: 3d illustration of futuristic design spaceship interior, medvedsky.kz; retrieved from shutterstock.com (https://www.shutterstock.com/zh/image-illustration/3d-illustration-futuristic-design-spaceship-interior-375587404)Last Accessed 25/9/2019

Images Used: Saturn's moon Titan, Elena11; retrieved from shutterstock.com (https://www.shutterstock.com/zh/image-photo/saturns-moon-titan-set-six-different-1287020371)Last Accessed 25/9/2019

Images Used: Dark spaceship interior with large window view on space 3D rendering, sdecoret; retrieved from shutterstock.com (https://www.shutterstock.com/zh/image-illustration/dark-spaceship-interior-large-window-view-1439669642)Last Accessed 25/9/2019

Code Snippets:

1. (Planet Explosion) retrieved from Rivermill Studios: Planet Explosion (https://assetstore.unity.com/packages/vfx/particles/fire-explosions/planet-explosion-138968) Last accessed 25/9/2019

2. (pongphotoresistor.ino) retrieved from DECO2300-7230-2019.slack.com: pongphotoresistor.ino(https://app.slack.com/client/TLL1JHLBH/CLHMDAT28/thread/CLHMDAT28-1569402795.031200) Last Accessed 25/9/2019

3. (Pong3 .zip) retrieved from DECO2300-7230-2019.slack.com: Starting Unity project for Week 6 Practicals(https://app.slack.com/client/TLL1JHLBH/CLHMDAT28)Last accessed

25/9/2019
4. (Arduino.variable resistance) retrieved from Youlairenzhenjiao: Practical 06-Analog Signal input (https://www.youtube.com/watch?v=cp8ETY29HVk&list=PLdckmk1Jf8MYOED98iY13wdGi52h-O69X&index=9)Last accessed 25/9/2019

5.（read image from multiple image for sprite）retrived from CSDN Blog: unity – read Sprite images(https://blog.csdn.net/m0_37583098/article/details/73302263) Last accessed 23/10/2019
6. (Creating a Restart Button) retrived from answers.unity.com: Creating a Restart Button (https://answers.unity.com/questions/1261937/creating-a-restart-button.html) Last accessed 23/10/2019
7. (pongbuttons . ino) retrieved from DECO2300-7230-2019.slack.com: Starting Unity project for Week 6 Practicals(https://app.slack.com/client/TLL1JHLBH/CLHMDAT28)Last accessed 24/10/2019
8. (Arduino.variable resistance) retrieved from Youlairenzhenjiao: Practical 05-Button Debounce input (https://www.youtube.com/watch?v=cp8ETY29HVk&list=PLdckmk1Jf8MYOED98iY13wdGi52h-O69X&index=9)Last accessed 24/10/2019

Audio Used:
Explosion track: Planet Explosion by Rivermill Studios in Planet Explosion.

Font Used:
Popular Pairings with Anton, Vernon Adams; retrieved from google.com (https://fonts.google.com/specimen/Anton?selection.family=Anton)Last accessed 25/9/2019

**Reference of Interactive Prototype 1:**
planet
explosion
Images Used: Plus, Icon Flat Vector Illustration, Farencia; retrieved from shutterstock.com (https://www.shutterstock.com/zh/image-vector/plus-icon-flat-vector-illustration-1432778753) Last Accessed 9/9/2019

Images Used: Blue Pink and White Andromeda Galaxy Way, Miriam Espacio; retrieved from pexels.com (https://www.pexels.com/photo/sky-space-milky-way-stars-110854/)Last Accessed 9/9/2019

Images Used: forest Icon, Rashad Ashur; retrieved from: shutterstock.com

(https://www.shutterstock.com/zh/image-vector/forest-icon-389418292) Last Accessed 9/9/2019

Images Used: People Icon - Vector Group of Business Person in Glyph Pictogram illustration Symbol, TukTuk Design; retrieved from shutterstock.com (https://www.shutterstock.com/zh/image-vector/people-icon-vector-group-business-person-414369094) Last Accessed 9/9/2019

Images Used: trash icon in trendy flat design, pambudi; retrieved from shutterstock.com (https://www.shutterstock.com/zh/image-vector/trash-icon-trendy-flat-design-1185282175) Last Accessed 9/9/2019

Animation used: Planet Explosion, Rivermill Studios; retrieved from unity.com (https://assetstore.unity.com/packages/vfx/particles/fire-explosions/planet-explosion-138968) Last Accessed 25/9/2019

Model used: Health and Progress Bars Pro, Martin Nerurkar; retrieved from unity.com (https://assetstore.unity.com/packages/tools/gui/health-and-progress-bars-pro-82405) Last Accessed 9/9/2019

Model used: 2D Space Kit, Brett Gregory; retrieved from unity.com (https://assetstore.unity.com/packages/tools/gui/health-and-progress-bars-pro-82405) Last Accessed 9/9/2019

# 9. Appendix

**Testing agenda in Interactive Prototype III:**

**Observation**

1. Observe the time it takes to turn the switch when entering the door and the mood when opening the door.
2. Observe the success/failure times of the game.
3. Observe whether the testers touch the corresponding photoresistor to control resource transfer.
4. Observe the time testers spent transferring resources.
5. Observe whether testers choose to play again after the failure.
6. Observe the testers' reaction to the success/failure of the game.

**Interview**

Questions that may be involved:

1. Are the conditions for the success of the game clear?
2. How do you evaluate the success/failure interface of the game?
3. If you divide the game playability into 1 to 5 stars, which one would you choose?
4. What is the most attractive item to you / what makes you confused?
5. What features do you want to add?
6. What do you think of the way you open the door?
7. Can you find the corresponding photoresistor accurately? Is it sensitive/slow enough?
8. Is control stable? What are the objective factors in the operation?
9. If there are more photoresistors, can you get used to it, such as 16?
10. Do you think game instruction is helpful?

**Observation**

| Tester number | Action number | Description |
|---|---|---|
| Tester1 | A1 | Spent 10 seconds and anxious. |
| | A2 | Fail after 3 clicks. |
| | A3 | Touch the wrong photoresistor. |
| | A4 | Within 10 seconds. |
| | A5 | Choose to play again and success. |
| | A6 | Not convinced and then be happy. |
| Tester2 | A1 | Spent 5 seconds and calm. |
| | A2 | Win after 15 clicks. |
| | A3 | Touch the correct photoresistor. |
| | A4 | About 2 minutes and a half on recognizing the words. |
| | A5 | Quit the game. |
| | A6 | Happy |
| Tester3 | A1 | Spent 2 seconds and calm. |

| | | |
|---|---|---|
| | A2 | Win after 15 clicks. |
| | A3 | Touch the correct photoresistor. |
| | A4 | Within 50 seconds. |
| | A5 | Quit the game. |
| | A6 | Stunned for a while and then show a little happy. |
| Tester4 | A1 | Spent 30 seconds and depressed. |
| | A2 | Fail after 5 clicks. |
| | A3 | Touch the wrong photoresistor. |
| | A4 | Pause before each step for about 3 seconds and stare at the screen. |
| | A5 | Choose to play again and success. |
| | A6 | Depressed and then be happy. |

**Interview**

| Tester number | Question number | Answer |
|---|---|---|
| Tester1 | Q1 | I figure it out in the second chance. |
| | Q2 | Good job, especially the explosion effect. |
| | Q3 | 3 stars. |
| | Q4 | The unexpected explosion for the first time is surprising. The planet could have some change when the number is changed. |
| | Q5 | No more functions, it is enough now. If the button could be optimized, it could be better. |
| | Q6 | It is interesting to set it as a coded lock. |
| | Q7 | No, but they are sensitive enough. |
| | Q8 | Generally good expect the wire could interrupt me. |
| | Q9 | 16 is too exaggerated, you should optimize it. |
| | Q10 | I cannot read it in time… |
| Tester2 | Q1 | I think it is quite clear. |
| | Q2 | The rebirth interface is too stable. |
| | Q3 | 4 stars. Nice work. |
| | Q4 | The background could be darker to highlight the words on it. |
| | Q5 | I think the button is useless now. Maybe you can make room for the planets and the words. |
| | Q6 | It is a good interaction. |
| | Q7 | Yes, sensitive. |
| | Q8 | I think it is stable enough. |
| | Q9 | No, too much. |

| | | |
|---|---|---|
| | Q10 | Not quite helpful. I think it is too general. |
| Tester3 | Q1 | I think it is clear by following the instruction. |
| | Q2 | I can figure out the win effect, but it could be better if it is an animation. |
| | Q3 | 4 stars! |
| | Q4 | I like the cockpit window, it is quite cool. |
| | Q5 | The success/failure instruction should be added to remind the players that they have finished the game. |
| | Q6 | Interesting and clear by the instruction. |
| | Q7 | Yes, quite sensitive. |
| | Q8 | It is stable, but I am always worried that the photoresistors could be influenced during the game even it does not influence. |
| | Q9 | No, it is terrible. You should reduce its number conversely. |
| | Q10 | I can figure out the basic rule by it, but still confused. |
| Tester4 | Q1 | A little bit hard…I can not figure it out after two attempts. |
| | Q2 | Explosion is awesome. |
| | Q3 | 4 stars. I think it is interesting even I fail twice. |
| | Q4 | I favor the HUD setting, but the interface should be more active when the player is still in the game rather than just the explosion effect. |
| | Q5 | More detailed instruction should be added, and the buttons are annoying because they have no function and always interfere with my vision. |
| | Q6 | Too hard to open because it is not sensitive enough. |
| | Q7 | Not but sensitive. |
| | Q8 | Not stable. The potentiometer is too hard to open. |
| | Q9 | No more photoresistors…Maybe you can try the control keypad? |
| | Q10 | I struggle to see them, and the content is not enough. Maybe you should let them stay longer, clearer and add more details. |