catalogue

I am a student in China, so this document is obtained by using Google to translate the Chinese document. If there is something strange, you can also check it against the Chinese document, or contact me at silmont@foxmail.com

Thanks a lot!

Chapter I Software Configuration Description

.1 1 Development system configuration

OS name:	Microsoft Windows 10 Professional Edition
OS edition:	10.0.19043
System Type:	x64-based PC
processor:	Intel64 Family 6 Model 158 Stepping 10 GenuineIntel ~2201 Mhz
graphics processing unit:	Intel(R) UHD Graphics 630
	NVIDIA GeForce GTX 1060

1.2 Development Environment Configuration

Unity:	2020.3.23f1c1 (.NET Standard 2.0)
VS Code:	1.66.2 (user setup)
Node.js:	16.13.0

Table 3.1 Minimum operating requirements of Unity 2020LTS

	Windows	macOS	Linux (supported in the Preview)
operate system edition	Windows 7 (SP1+), Windows 10 and Windows 11, 64-bit versions only.	High Sierra 10.13+	Ubuntu 20.04, Ubuntu 18.04, and CentOS 7
CPU	X64 Schema (supporting the SSE2 instruction set)	X64 Schema (supporting the SSE2 instruction set)	X64 Schema (supporting the SSE2 instruction set)
figure API	The DX10, DX11, and DX12 are compatible with the GPU	Metal-compatible type Intel and AMD GPU	OpenGL 3.2 + or Vulkan compatible Nvidia and AMD GPU
other ask	Hardware supplier Officially supported drivers	Drivers officially supported by the Apple	The Gnome desktop environment, the Nvidia official proprietary graphics driver, or the AMD Mesa graphics driver, running on an X11 window system.Other configuration and consumer environments provided, and supported distributions (kernel, synthesizer, etc.)
	For all operating systems, workstation or laptop models support Unity Editor, running without a simulation, container, or compatibility layer.		

Chapter II　Main functions and performance indicators

2.1 Main functions

In general, this design needs to complete the calculation of object state changes according to certain rules.Specifically, it can be disassembled as the following points:

- Calculate whether the receiver meets the reaction condition at some point
- If fit, calculate what reactions should occur
- Allow users to customize the reaction rules
- Allows users to customize the element and receive categories
- Provide unified environmental condition management

2.2 Performance indicators

This design serves the game development, and the main performance index is the game frame rate.The higher the frame rate, the better the performance, and the worse the performance.

Chapter III　The User Manual and Tutorials

3.1 Overview of the interaction mode and process

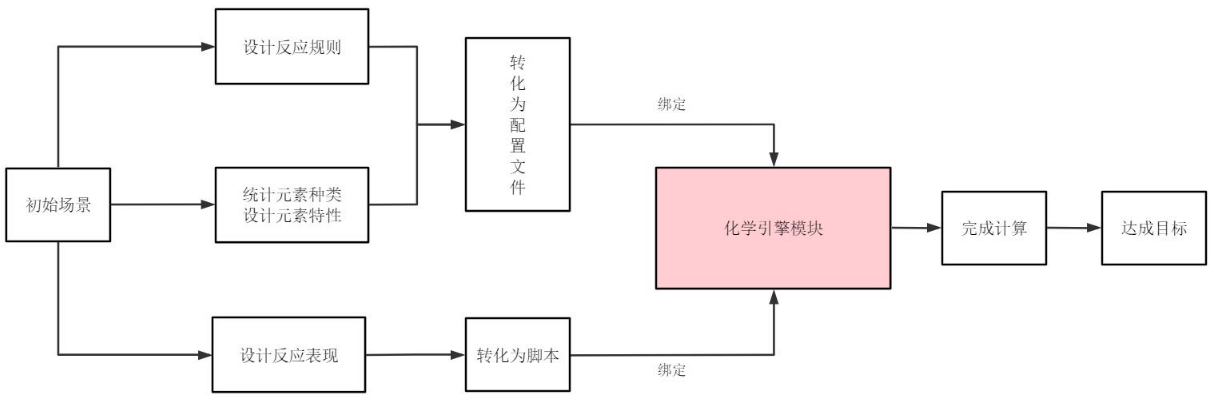


Figure 3.1, Interaction flow

The user first determines the type of elements, element characteristics, reaction rules and reaction performance, and transforms them into the corresponding format.Where, the reaction rule and the element type according to.Format storage of the json file, element features can be configured directly in the Inspector panel, and reaction performance needs to be converted into a code language, using the agreed function name.

After that, the above files are bound to the module, and after the game is started, the module is automatically calculated and called.

3.2 Quick Tutorial Manual

3.2.1 Run Sample Scene (be skipped)

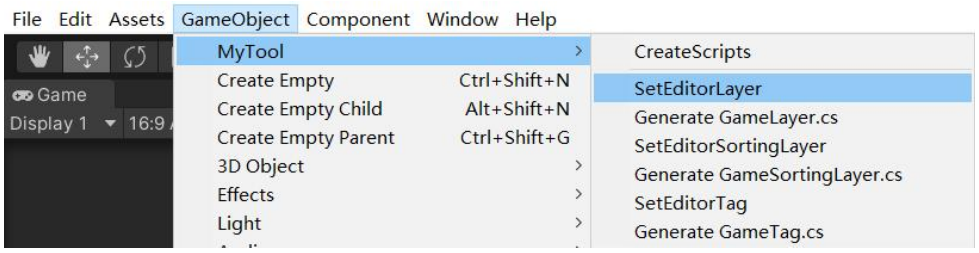
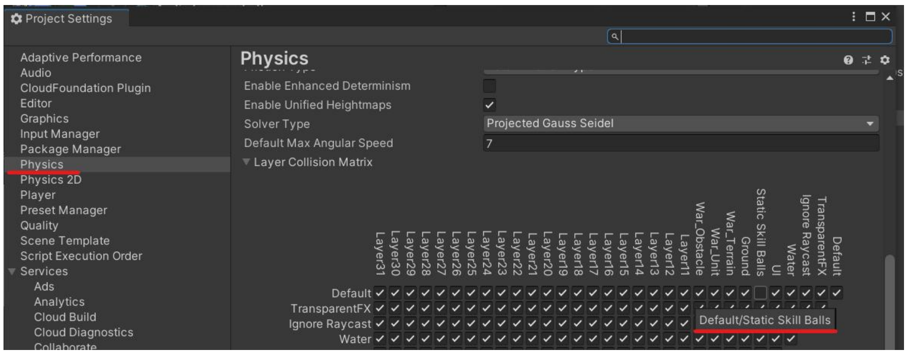


Figure 3.2 Setup for Layer

After the import, click GameObject-MyTool-SetEditorLayer.This step synchronizes the Layer information in the Demo scene to your project.Note that this step overrides your existing Layer information, so try it in a new project; if you do not need to try the Demo file, you can skip this step and start with 3.2.2.



Enable Enhanced Determinism
Enable Unified Heightmaps ✓
Solver Type Projected Gauss Seidel
Default Max Angular Speed 7

Figure 3.3 Setting the collision conditions

Next, open Edit-Project Settings, select the Physics tab, find Layer Collision Matrix, and uncheck Default / Static Skill Balls.It is in the sixth column of the first row, see figure above.

Then please open the Sample01 under the Scene folder. Check if the Layer of Stylized Astronaut in the scene has been marked as "Static Skill Balls".If so, now that we have completed the Sample Scene setup, we can start using it!

Click on the Run button.You will see the game running and the camera shaking slowly as the player breath.You can press the number key 1 or 2 above the keyboard to fire a weapon with fire or water attributes.When you press 2, the water weapon is fired, but the scene does not change; when you press 1, the fire weapon is fired, and the combustible materials such as trees in front of you are lit, with color changes and red smoke.

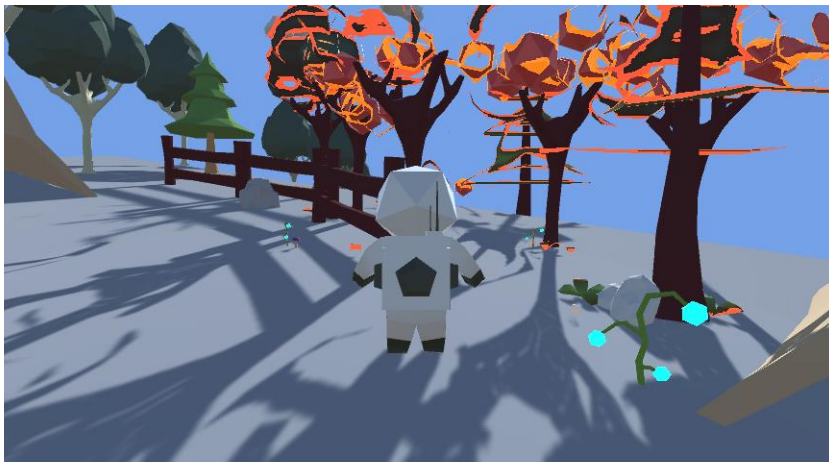


Figure 3.4, igniting the trees

When the tree is not finished burning, you can press 2 to fire the water attribute weapon.The burning trees in front of them will be extinguished, colored darker and emit dark blue smoke.

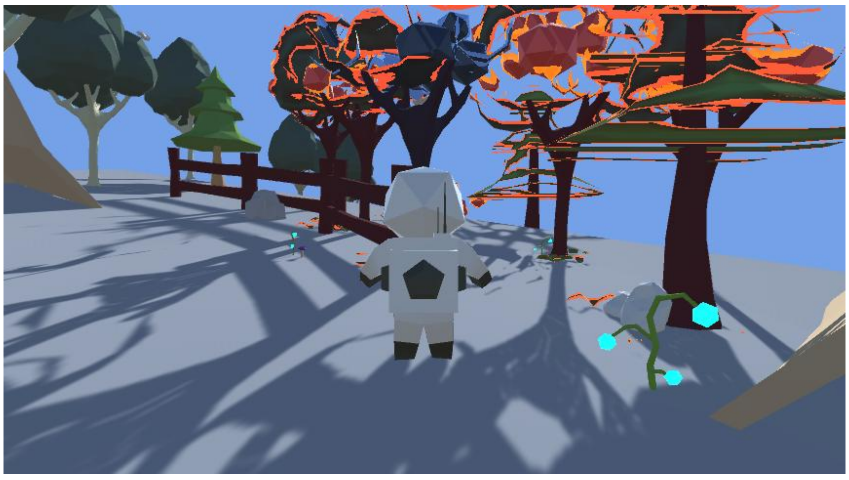


Figure 3.5 Put Out the burning trees

You can use WASD to roam the scene and try to ignite and extinguish other combustible objects.

Next, we open up the Sample Scene02. You'll see empty scenes, but don't worry.Click the Run button and the game will generate 50 * 50 simple wood.You can press the M button to observe the ignition of the wood, red represents the wood is ignited.If you see the red area spreading, the configuration is normal.

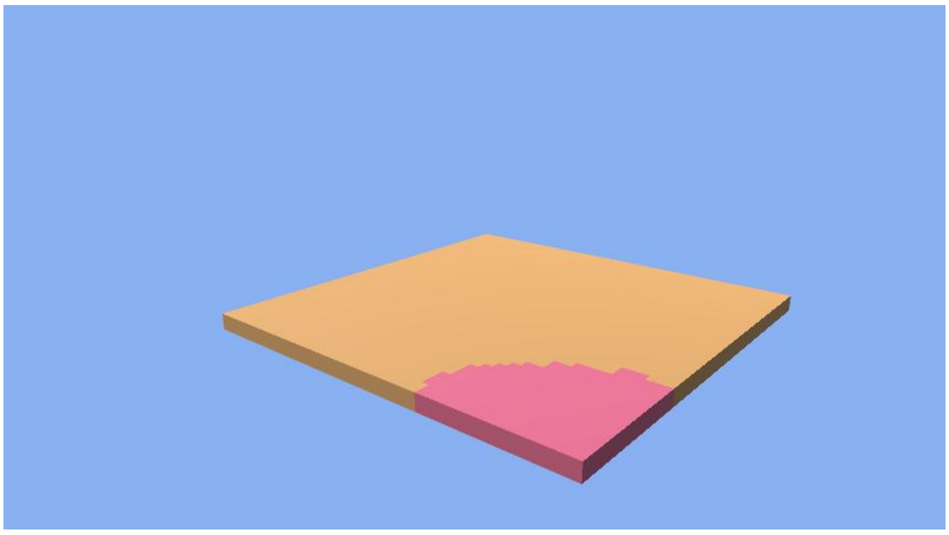


Figure 3.6 Transmission of fire elements

If there is a problem in these steps, reimport the kit, or save your Unity version, run screenshot, and the Physics section screenshot in Project Settings, and contact the developer.

3.2.1 Configure your own chemical engine

Open the ChemistryEngineScripts folder in File Explorer or Visit and open Scripts-StreamingAssets-Config and you will see two.The json file, used for configuring your own chemical engine.The developer has given examples in which you can just follow the steps to learn the configuration without changing it yourself.

```
[
{
EleType s: Fire
}
]
```

In ElementManager, you can use the EleType keyword to add your own element types that will automatically generate an ID starting from 0 without manual configuration.Types are displayed in the Element Settings panel to distinguish between different elements.In the tutorial, fire with ID=0, ice with ID=1, and water with ID=2 are already configured.

```
[
{
"FuncName": "OnFire",
"PrevElementID": "-1",
"NextElementID": "0"
}
]
```

In RuleManager, you can define your reaction rules using the above format: when the receiver changes from the state of PrevElementID to NextElementID, the engine calls the function with the name FuncName.The ID of-1 indicates that the receiver is not currently responding, and the OnFire function is called when it is affected by an element of ID=0.The OnFire function specifies the performance of the reaction and you need to write yourself and bind this script to the receiver.In the tutorial, the following rules have been created:

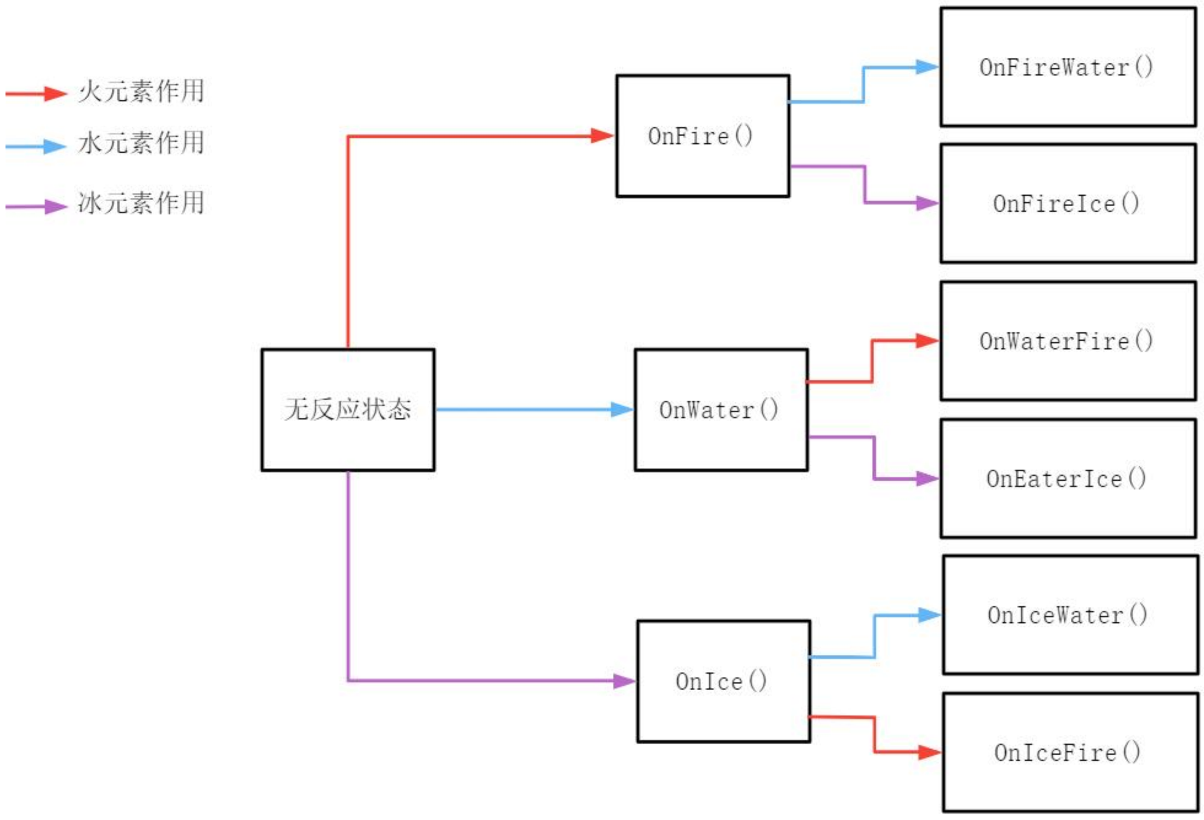


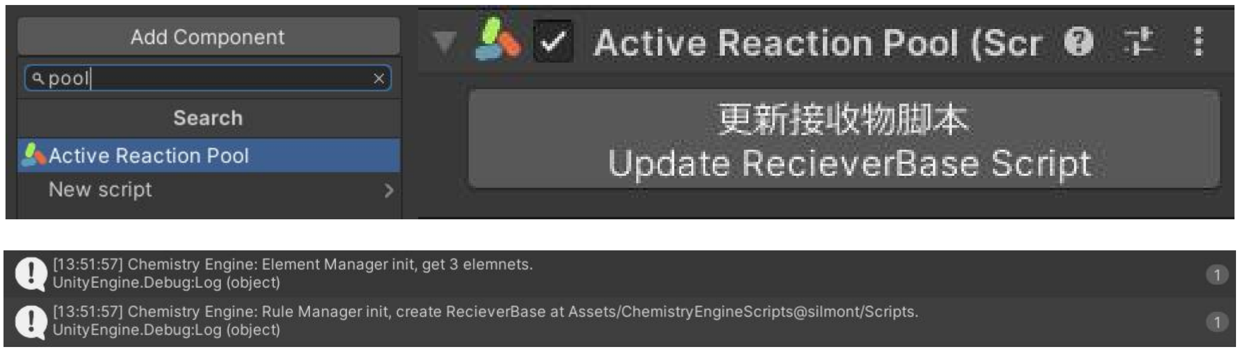
Figure 3.7 Response rules set up in the tutorial

3.2.2 Serve your scenario with a chemical engine

First, create a new scene, create an empty object, and name it CE.

Add the components to the CE

Click Add Component, search for "pool," add Active Reation Pool components, and click the Update Receipt Script button, and if successful, will print the following Log.



```
[13:51:57] Chemistry Engine: Element Manager init, get 3 elemnets.
UnityEngine.Debug:Log (object)
[13:51:57] Chemistry Engine: Rule Manager init, create RecieverBase at Assets/ChemistryEngineScripts@silmont/Scripts.
UnityEngine.Debug:Log (object)
```

Figure 3.8 Update the received script

Create the elements and the receivers

Create a cube as the receiver in the scene and create a sphere as the element.Named them "Reciever" and "Element".The Transform of the Reciever.position is set to (-0.5,0,0), the Transform of Element.The position is set to (0.5,0,0).Make sure that you can see two objects in the camera preview screen.

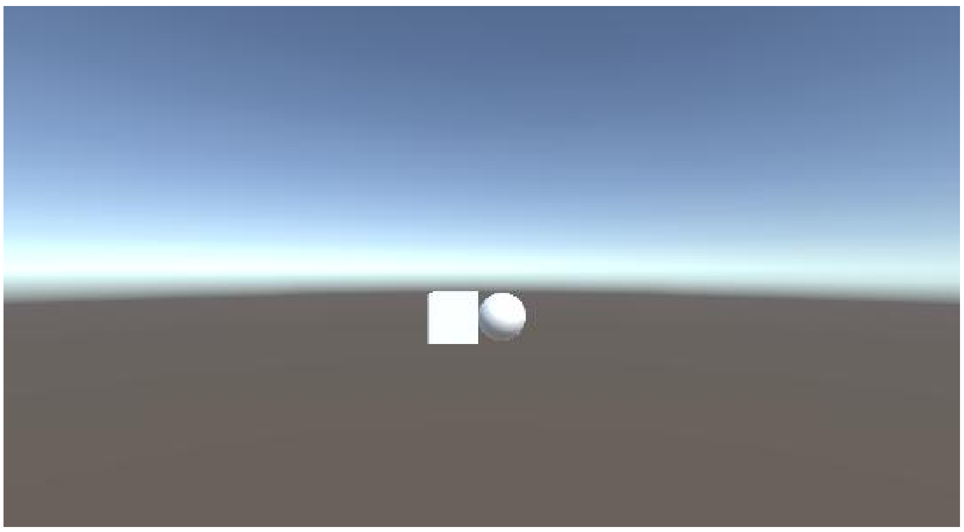
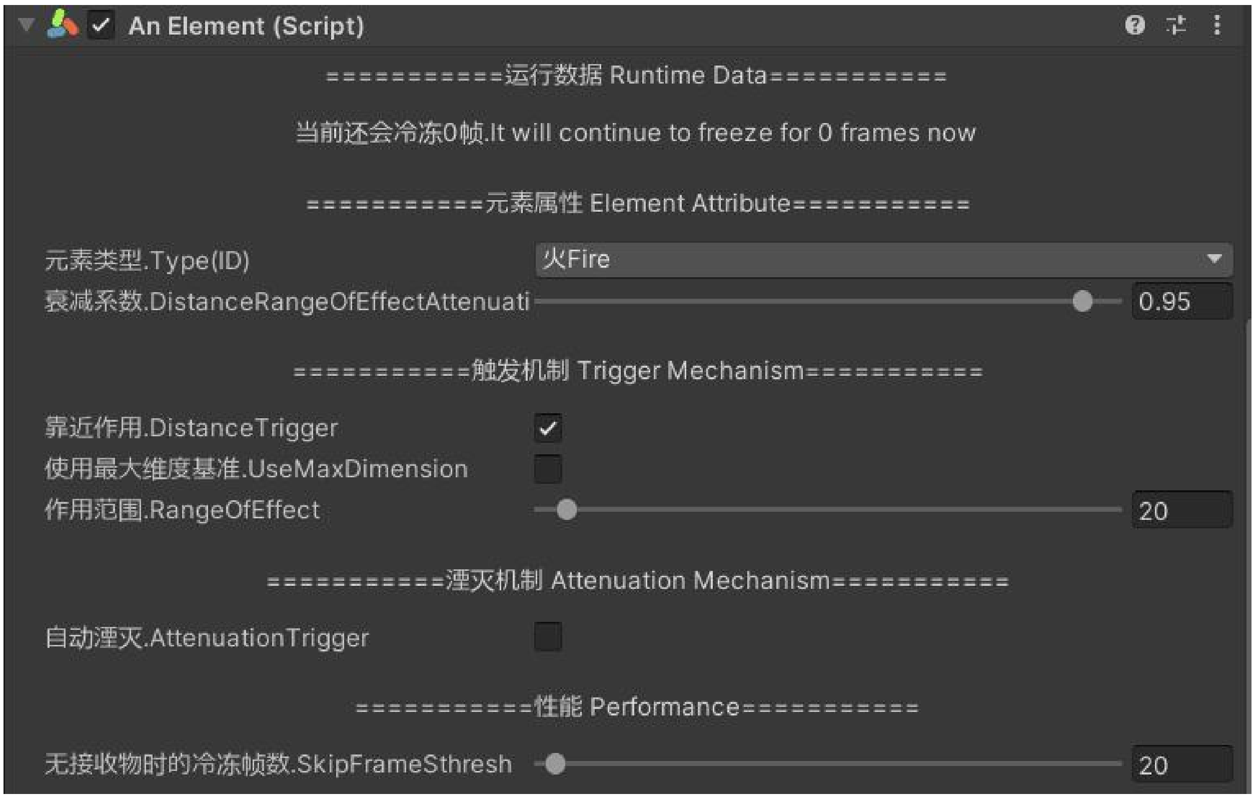


Figure 3.9 Creating Elements and Receivers

Set the element properties

Add an element component for the Element.Click Add Component, search "ele", add An Element component, normal interface is shown in the figure below.If you find a Unity error reported here, please confirm that you have clicked the Update Receive Script button.If you still report an error after clicking again, please contact the developer.


An Element (Script)

===========运行数据 Runtime Data===========

当前还会冷冻0帧.It will continue to freeze for 0 frames now

===========元素属性 Element Attribute===========

元素类型.Type(ID) 火Fire

衰减系数.DistanceRangeOfEffectAttenuati 0.95

===========触发机制 Trigger Mechanism===========

靠近作用.DistanceTrigger ✓

使用最大维度基准.UseMaxDimension

作用范围.RangeOfEffect 20

===========湮灭机制 Attenuation Mechanism===========

自动湮灭.AttenuationTrigger

===========性能 Performance===========

无接收物时的冷冻帧数.SkipFrameSthresh 20

Figure 3.10 The Element Settings Panel

First, we explain how the panel is used.The panel consists of four plates: running data, elemental attributes, trigger mechanism, annihilation mechanism, and performance.

Running data: When the game enters Play Mode, the freezing and life cycle of the corresponding elements (if any).

Elements Properties:

Element type drop-down menu can select us in ElementManager.Type of element set in json.

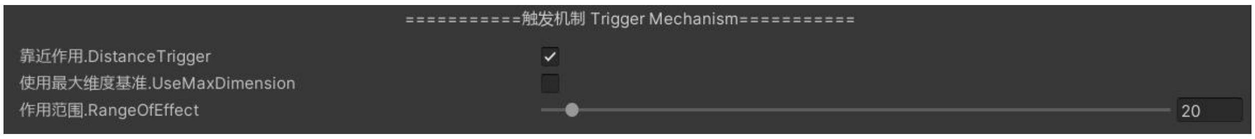
The propagation decay coefficient defines the degree to which the influence range of the element can decay at each propagation.For example, for elements with a propagation decay coefficient of 0.9 and an action range of 100, after a propagation, the element carried by the receiver changes to 0.9 * 100=90. Note that the propagation decay coefficient itself decay, so the range of propagation is squared.

Trigger mechanism: Define how the element acts.

If not checked, it means that the element can only act when the actual object and the receiver collide.

If the element checks the near action, you can further set the action range of the element.This range is the radius expanding outward relative to the spherical collider of an object.

For example, an object uses Box Collider, this Collider is x 20, y 30, z 50 respectively, set its action range to 40, then the engine will look for a legitimate receiver within the radius of (x 20 + 40=) 60.If you check Use Maximum Dimension Base, the radius of (z 50 + 40=) 90 is used to find it.The maximum dimension datum is whether the spherical collider radius of the object adopts the maximum dimension in the current collider xyz axis.


===========触发机制 Trigger Mechanism==========

靠近作用.DistanceTrigger ✓

使用最大维度基准.UseMaxDimension

作用范围.RangeOfEffect 20

Figure 3.11 Elements tick the interface near the action

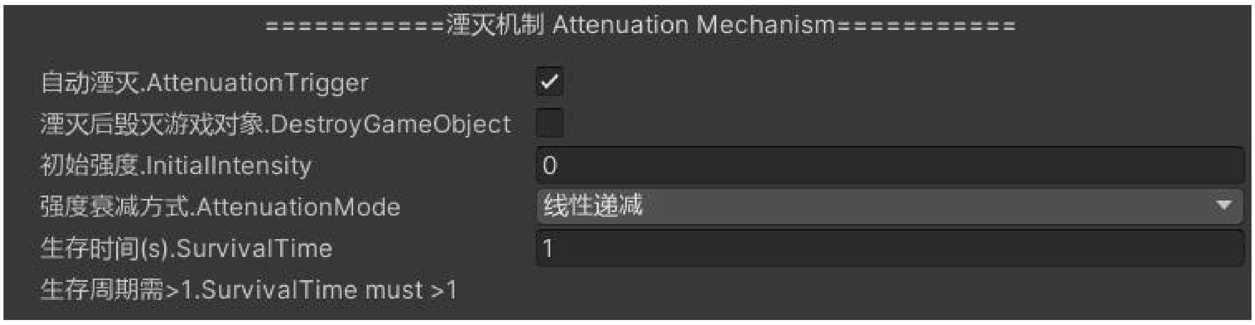
Oblihilation mechanism: For users to design simple elements for rapid verification of game mechanics.But because this feature confuses top-level user operations with middle-level computing, and can cause unexpected conflicts, it is not recommended in actual games except to verify whether the game mechanics are reasonable.

After checking the automatic annihilation, you can further set the life time of the element, and the life countdown opens after the element is activated.

Check "Do you destroy the game object after annihilation" will destroy the object that hangs the element while the element annihilate.

The initial intensity is the intensity before the countdown, which decreases over time until below 1, and the element "annihilate".

There are two ways to decrease the strength, one is linear decline, decrease the fixed intensity per second, the user can directly set the element after annihilation; the second is nonlinear decline, the strength of the element will drop faster and faster, so the user can set the life time of the element indirectly by setting the decay factor.


===========湮灭机制 Attenuation Mechanism==========

自动湮灭.AttenuationTrigger ✓

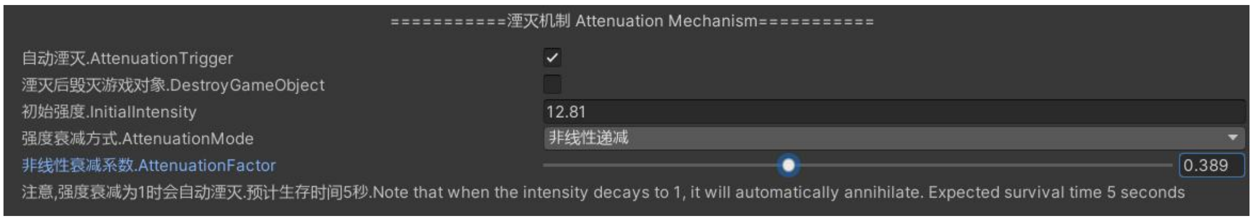
湮灭后毁灭游戏对象.DestroyGameObject

初始强度.InitialIntensity 0

强度衰减方式.AttenuationMode 线性递减

生存时间(s).SurvivalTime 1

生存周期需>1.SurvivalTime must >1



===========湮灭机制 Attenuation Mechanism==========

自动湮灭.AttenuationTrigger	✓
湮灭后毁灭游戏对象.DestroyGameObject	
初始强度.InitialIntensity	12.81
强度衰减方式.AttenuationMode	非线性递减
非线性衰减系数.AttenuationFactor	0.389

注意,强度衰减为1时会自动湮灭.预计生存时间5秒.Note that when the intensity decays to 1, it will automatically annihilate. Expected survival time 5 seconds

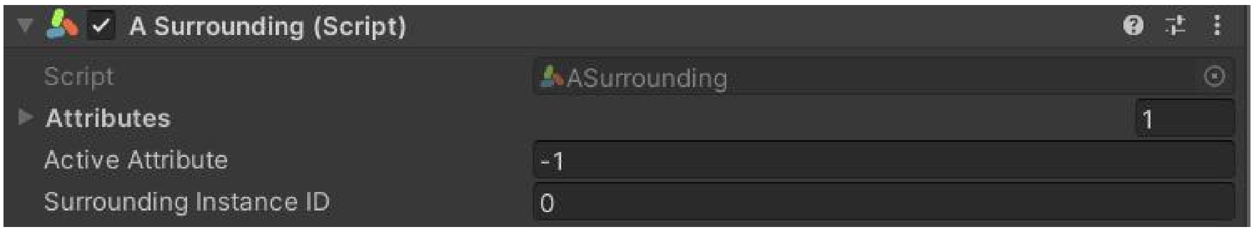
Figure 3.12 Elements tick the interface for automatic annihilation

Performance: This engine uses slow start methods in computer networks to gradually slow monitoring of active elements.The performance panel mainly controls the freezing threshold at the beginning of the element.When the threshold is 0, each element frame is monitored, consuming performance but monitored timely and accurately; when the threshold is not zero, the freezing time of the element is dynamically changed in the game to maximize performance.It is recommended to set it to 20. A high threshold will delay the element action.

In the tutorial, set the element properties as shown in Figure 3.10.

Set the receiver properties

Add the chemical components to the Reciever.Click Add Component, search for "surr", add the A Surrounding component, the normal interface is shown in the figure below.If you find a Unity error reported here, please confirm that you have clicked the Update Receive Script button.If you still report an error after clicking again, please contact the developer.

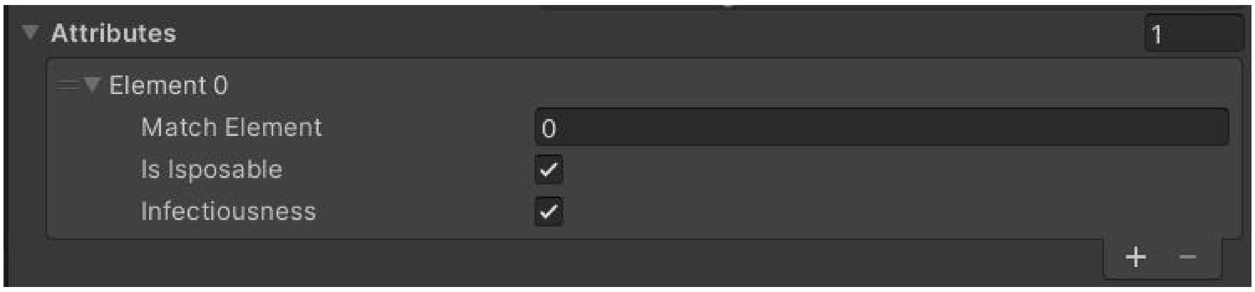


A Surrounding (Script)

Script	ASurrounding
Attributes	1
Active Attribute	-1
Surrounding Instance ID	0

Figure 3.13 Receiving object property panel

For the receiver, we just need to set up the Attributes.The Attributes marks what element the receiver can react with.Click the plus sign to add and set the following attributes.



Attributes	1
Element 0	
Match Element	0
Is Isposable	✓
Infectiousness	✓

Figure 3.14 Receiving object property panel

Match Element marks the ID of the element, and Is Isposable marks whether the property is used at once, if checked, which means it can only be activated once.For example, lighting of wood is disposable and melting of ice is not disposable.The Infectiousness marks whether the attribute can copy its corresponding element at the same time.For example, wood being lit can carry new fire elements, but being wet with wood is merely getting wet and can not become a new water element.

Write a receiving script

In RuleManager, we have specified that the object is executed by the OnFire function when affected by the fire element.Create a new script, named the Cube.cs。Next, you will learn how to make the chemical engine communicate with your code.

First, let the Cube class inherit the RecieverBase class.

Then we write a OnFire function for Cube to print the "OnFire" when it is lit.Note to use the override keyword and contain a shape parameter of type AnElement.

```
public  override  void  OnFire (AnElement e)
{
Debug.Log("OnFire");
}
```

Then, we bind this function in the Start ():

```
void  Start ()
{
Bound();
}
```

Now you get a full receiver script for mounting it to the Reciever.

Write an element script

So when will an element start working?We also need to write the element script.Create a new script and name it a Sphere.cs。Declaration a member variable of type AnElement and initialize it in the Start function.

```
AnElement  _t his E lement;
void  Start()
{
_t his E lement  = gameObject.GetComponent<AnElement>();
}
```

We want the element to be activated when the player presses the M key:

```
if  (Input.GetKeyDown(KeyCode.M))
{
```

```
this_element.ElementIfActive  = true ;
}
```

Now you get a full element script to mount it to the Element.

Run your game

Click the Run button, and press the M button, and the Console window displays the word "OnFire".congratulations!You have mastered the basic way to use this tool!

3.2.3 Advanced 1: Use the weather system

We know that many of the intensity of reactions is dependent to weather conditions.For this purpose, the developer provides an environment manager with the default humidity (wet), temperature (temprature) and wind (wind), the default value is 1.0f.

First, in Cube.Declaration of the manager variable in the cs.Then, in the OnFire function, get the humidity value and print it out.

```
public  class  Cube  : RecieverBase
{
Environment _environment = Environment.EnvironmentInstance;
void  Start ()
{
Bound();
}

public  override  void  OnFire (AnElement e)
{
Debug.Log(_environment.GetEnvironment("wet"));
Debug.Log("OnFire");
}
}
```

Run the game, activate the elements, and print out the "1" in the Console window, indicating that we have successfully obtained the humidity values!Other weather system API, please refer to the documentation: https://silmont2000.github.io/ChemistryEngineDoc/

3.2.4 Advanced 2: Rewriting on the existing basis

This project is the graduation design of the author during his undergraduate period. The sample size is not large enough, and there are many shortcomings. Correction and rewriting are welcome.Refer to the documentation for specific logic:

https://silmont2000.github.io/ChemistryEngineDoc/