**天车之辙** 项目汇报

一，需求分析

1. 用户类型：

2D RPG游戏爱好者、小说爱好者。

1. 项目背景：

随着当今阅读形式越来越多样化，互动式阅读取代了部分电子书阅读，越来越受到大众青睐。但是目前市面上的互动式阅读器可操作性不强，读者只能跟随着作者设定好的剧情单线发展下去，很难自己操控角色改变剧情走向，只能看着角色走向其固定化的结局。

为了弥补这一空缺，我们将整个阅读模式架构在游戏的基础上，读者在感受剧情的同时可以自行操作角色，通过自己的操作与选择来触发不同的剧情，自由度高，沉浸体验更佳。

1. 功能介绍：

本项目在互动式阅读的基础上，采用2D RPG游戏的形式复现小说剧情，实现互动式阅读器与游戏的结合，并加入解谜元素，完成游戏、阅读、解谜三合一，让玩家以第一视角沉浸式感受剧情的魅力。画风采取像素风，更显可爱风趣。

**二，人员分工**

组员-苏淇：基础游戏功能实现、阅读器互动、镜头跟随（50%）

组员-王子涵：游戏地图制作、角色运动与移动、UI设计（50%）

1. **开发计划**

###### 6月下旬：

1. 完成游戏地图的制作
2. 完成基本游戏功能
3. 动画制作

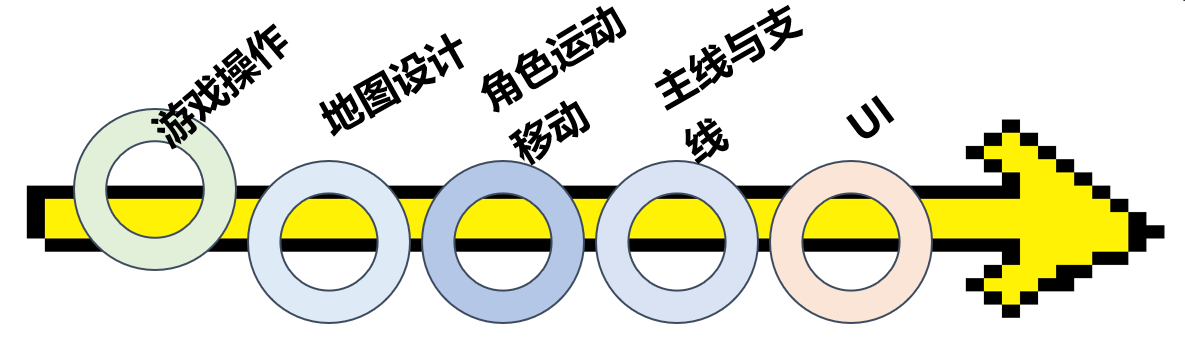
###### 7月上旬：

1. 核心功能全面实现
2. 基础UI设计

###### 7月上旬后期后：

1. 完成附加功能
2. 实现进程储存
3. 背景音乐、游戏动画等完善
4. 剧情跟随，增加新地图、新关卡
5. **项目总体架构**

实现项目需要游戏操作、对话、剧情跟随、动画等多种功能的网状结合，考虑到功能的复杂性以及人力有限，我们将项目任务拆分成线状的小任务串联起来，分工完成每个部分，最后结合在一起。



**Step1 Step2 Step3 Step4 Step5**

## 模块

##### 安装与进入游戏

安装后无需创建ID可直接进入开始游戏，进程储存功能可实现之前剧情的跳转。

##### 游戏功能

Part1：操作角色移动、与NPC对话、拾取掉落物品、使用背包内物品

Part2：多个不同地图的设计与绘制、地图间的传送、地图中线索物品的触发功能

Part3：角色的运动动画效果

Part4：触发不同机关产生不同剧情与对话

##### 剧情设计

通过对话框的形式展现小说剧情，实现角色属性查看、与人物对话等功能，并根据玩家操作顺序展开不同的剧情。

##### 动画及音乐添加

地图切换之间伴随音乐转换，增加人物对话时的动画。

**核心功能：**

1.控制人物在地图内移动，自由度高，可探索性高。

2.对话功能。剧情跟随玩家变化，根据玩家的操作不同可能触发不同的剧情及彩蛋。

## 辅助功能

1. 进程储存。玩家可以根据需要实时保存游戏进度。
2. 音效及动画。
3. **模块&函数细节、**

1.Basic Scripts

1.1:CameraMove:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class CamareMove : MonoBehaviour

{

public bool cinameSmoothMove;

private Transform target;

[SerializeField] private float smoothSpeed;

private void Start()

{

target = PlayerController.instance.transform;

}

private void LateUpdate()

{

if (cinameSmoothMove == false)

{

transform.position = new Vector3(target.position.x, target.position.y, transform.position.z);//完全跟随

}

transform.position = Vector3.Lerp(transform.position, new Vector3(target.position.x, target.position.y, transform.position.z), smoothSpeed \* Time.deltaTime);//缓慢跟随

}

}

1.2:Destory:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Destroy : MonoBehaviour

{

private float timer;//计时器

private void Update()

{

timer -= Time.deltaTime;

if (timer == 0)

{

Destroy(gameObject);

}

}

}

1.3:Entrance:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Entrance : MonoBehaviour

{

public string entrancePassword;

private void Start()

{

if (PlayerController.instance.scenePassword == entrancePassword)

{

PlayerController.instance.transform.position = transform.position;

}

}

}

1.4:ESC:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class ESC : MonoBehaviour

{

void Update()

{

if (Input.GetKeyDown(KeyCode.Escape))

{

Application.Quit();

}

}

}

1.5:Exit:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class Exit : MonoBehaviour

{

public string sceneName;

[SerializeField] private string newScenePassword;

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Player"))

{

PlayerController.instance.scenePassword = newScenePassword;

SceneManager.LoadScene(sceneName);

}

}

}

1.6:ExitDialogeTrigger:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class ExitDialogeTrigger : MonoBehaviour

{

public string sceneName;

[SerializeField] private string newScenePassword;

public GameObject player;

public int number;

private void Update()

{

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

if (dialogeNumber.dialogeNumber == number )

{

PlayerController.instance.scenePassword = newScenePassword;

SceneManager.LoadScene(sceneName);

}

}

}

1.7:Leave:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class Leave : MonoBehaviour

{

private void Start()

{

this.GetComponent<Button>().onClick.AddListener(OnClick);

}

private void OnClick()

{

Application.Quit();

}

}

2.Dialoge Scripts

2.1:DialogeButton:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class DialogeButton : MonoBehaviour

{

public GameObject Button;

private void OnTriggerEnter2D(Collider2D other)

{

if (other.tag == "Player")

{

Button.SetActive(true);

}

}

private void OnTriggerStay2D(Collider2D other)

{

PlayerController player = other.GetComponent<PlayerController>();

if (Button.activeSelf && Input.GetKeyDown(KeyCode.E))

{

player.dialogeJudge = true;

}

}

private void OnTriggerExit2D(Collider2D other)

{

Button.SetActive(false);

}

}

2.2:DialogeNumber:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class DialogeNumber : MonoBehaviour

{

public GameObject player;

public GameObject background;

public GameObject dialoge0;

public GameObject dialoge1;

public GameObject dialoge2;

public GameObject dialoge3;

public GameObject dialoge4;

public GameObject dialoge5;

public GameObject attribute;

public int number;

private bool dialogeJudge;

void Update()

{

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

number = dialogeNumber.dialogeNumber;

dialogeJudge = dialogeNumber.dialogeJudge;

if (number == 0)

{

dialoge0.SetActive(true);

background.SetActive(true);

}

if (number == 1)

{

dialoge1.SetActive(true);

}

if (number == 2)

{

attribute.SetActive(true);

dialoge2.SetActive(true);

}

if (number == 3 && dialogeJudge)

{

dialoge3.SetActive(true);

}

if (number == 4 && dialogeJudge)

{

dialoge4.SetActive(true);

}

if (number == 5 && dialogeJudge)

{

dialoge5.SetActive(true);

}

}

}

2.3:DialogeNumber1:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class DialogeNumber1 : MonoBehaviour

{

public GameObject player;

public GameObject background;

public GameObject dialoge0;

public GameObject dialoge1;

public GameObject dialoge2;

public GameObject dialoge3;

public GameObject dialoge4;

public GameObject dialoge5;

public GameObject dialoge6;

public int number;

private bool dialogeJudge;

void Update()

{

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

number = dialogeNumber.dialogeNumber;

dialogeJudge = dialogeNumber.dialogeJudge;

if (number == 0)

{

dialoge0.SetActive(true);

background.SetActive(true);

}

if (number == 1)

{

dialoge1.SetActive(true);

}

if (number == 2 && dialogeJudge)

{

dialoge2.SetActive(true);

}

if (number == 3 && dialogeJudge)

{

dialoge3.SetActive(true);

}

if (number == 4 && dialogeJudge)

{

dialoge4.SetActive(true);

}

if (number == 5 && dialogeJudge)

{

dialoge5.SetActive(true);

}

if (number == 6 && dialogeJudge)

{

dialoge6.SetActive(true);

}

}

}

2.4:DialogeOneAdd:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class DialogeOneAdd : MonoBehaviour

{

[Header("UI组件")]

public Text textLable;//要输出的文本

public Image faceImage;//要输出的头像

[Header("文本文件")]

public TextAsset textFile;//外部文件

[Header("头像")]

public Sprite face01;

public GameObject player;

public GameObject background;

public int index;

public float textSpeed;

bool textFinished;

bool cancerTyping;

List<string> textList = new List<string>();

void Awake()

{

GetTextFromFile(textFile);

}

private void OnEnable()

{

textFinished = true;

StartCoroutine(SetTextUI());

}

void Update()

{

if (Input.GetMouseButtonDown(0) && index == textList.Count)

{

gameObject.SetActive(false);

index = 0;

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

dialogeNumber.dialogeNumber++;

dialogeNumber.dialogeJudge = false;

background.SetActive(false);

return;

}

if (Input.GetMouseButtonDown(0))//如果按鼠标左键

{

if (textFinished && !cancerTyping)//如果完成，下一段

{

StartCoroutine(SetTextUI());

}

else if (!textFinished && !cancerTyping)//如果未完成，开启跳过编辑

{

cancerTyping = true;

}

}

}

void GetTextFromFile(TextAsset file)

{

textList.Clear();

index = 0;

var lineData = file.text.Split('\n');//对文档进行逐行切割，按行次序形成数组

foreach (var line in lineData)

{

textList.Add(line);

}

}

IEnumerator SetTextUI()

{

textFinished = false;

textLable.text = "";

switch (textList[index].Trim())//根据输入的字母判断应用哪个头像

{

case "A": faceImage.sprite = face01; index++; break;

}

int letter = 0;

while (!cancerTyping && letter < textList[index].Length - 1)//如果跳过编辑开启，结束进程，直接输出；如果未开启，逐字输出

{

textLable.text += textList[index][letter];

letter++;

yield return new WaitForSeconds(textSpeed);

}

textLable.text = textList[index];

cancerTyping = false;//取消跳过字幕编辑

textFinished = true;//完全输出完毕

index++;

}

}

2.5:DialogeThreeAdd:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class DialogeThreeAdd : MonoBehaviour

{

[Header("UI组件")]

public Text textLable;//要输出的文本

public Image faceImage;//要输出的头像

[Header("文本文件")]

public TextAsset textFile;//外部文件

[Header("头像")]

public Sprite face01, face02, face03;

public GameObject player;

public GameObject background;

public int index;

public float textSpeed;

bool textFinished;

bool cancerTyping;

List<string> textList = new List<string>();

void Awake()

{

GetTextFromFile(textFile);

}

private void OnEnable()

{

textFinished = true;

StartCoroutine(SetTextUI());

}

void Update()

{

if (Input.GetMouseButtonDown(0) && index == textList.Count)

{

gameObject.SetActive(false);

index = 0;

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

dialogeNumber.dialogeNumber++;

dialogeNumber.dialogeJudge = false;

background.SetActive(false);

return;

}

if (Input.GetMouseButtonDown(0))

{

if (textFinished && !cancerTyping)//如果完成，下一段

{

StartCoroutine(SetTextUI());

}

else if (!textFinished && !cancerTyping)//如果未完成，开启跳过编辑

{

cancerTyping = true;

}

}

}

void GetTextFromFile(TextAsset file)

{

textList.Clear();

index = 0;

var lineData = file.text.Split('\n');//对文档进行逐行切割，按行次序形成数组

foreach (var line in lineData)

{

textList.Add(line);

}

}

IEnumerator SetTextUI()

{

textFinished = false;

textLable.text = "";

switch (textList[index].Trim())//根据输入的字母判断应用哪个头像

{

case "A": faceImage.sprite = face01; index++; break;

case "B": faceImage.sprite = face02; index++; break;

case "C": faceImage.sprite = face03; index++; break;

}

int letter = 0;

while (!cancerTyping && letter < textList[index].Length - 1)//如果跳过编辑开启，结束进程，直接输出；如果未开启，逐字输出

{

textLable.text += textList[index][letter];

letter++;

yield return new WaitForSeconds(textSpeed);

}

textLable.text = textList[index];

cancerTyping = false;//取消跳过字幕编辑

textFinished = true;//完全输出完毕

index++;

}

}

2.6:DialogeTrigger:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class DialogeTrigger : MonoBehaviour

{

private void OnTriggerEnter2D(Collider2D collision)

{

PlayerController player = collision.GetComponent<PlayerController>();

if (player != null)//如果玩家的判定不为空，防止敌人接触后报错

{

player.dialogeJudge = true;

Destroy(gameObject);

}

}

}

2.7:DialogeTrigger1:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class DialogeTrigger1 : MonoBehaviour

{

private void OnTriggerEnter2D(Collider2D collision)

{

PlayerController player = collision.GetComponent<PlayerController>();

if (player != null)//如果玩家的判定不为空，防止敌人接触后报错

{

player.dialogeJudge = true;

}

}

}

2.8:DialogeTwoAdd:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class DialogeTwoAdd : MonoBehaviour

{

[Header("UI组件")]

public Text textLable;//要输出的文本

public Image faceImage;//要输出的头像

[Header("文本文件")]

public TextAsset textFile;//外部文件

[Header("头像")]

public Sprite face01, face02;

public GameObject player;

public GameObject background;

public int index;

public float textSpeed;

bool textFinished;

bool cancerTyping;

List<string> textList = new List<string>();

void Awake()

{

GetTextFromFile(textFile);

}

private void OnEnable()

{

textFinished = true;

StartCoroutine(SetTextUI());

}

void Update()

{

if (Input.GetMouseButtonDown(0) && index == textList.Count)

{

gameObject.SetActive(false);

index = 0;

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

dialogeNumber.dialogeNumber++;

dialogeNumber.dialogeJudge = false;

background.SetActive(false);

return;

}

if (Input.GetMouseButtonDown(0))

{

if (textFinished && !cancerTyping)//如果完成，下一段

{

StartCoroutine(SetTextUI());

}

else if (!textFinished && !cancerTyping)//如果未完成，开启跳过编辑

{

cancerTyping = true;

}

}

}

void GetTextFromFile(TextAsset file)

{

textList.Clear();

index = 0;

var lineData = file.text.Split('\n');//对文档进行逐行切割，按行次序形成数组

foreach (var line in lineData)

{

textList.Add(line);

}

}

IEnumerator SetTextUI()

{

textFinished = false;

textLable.text = "";

switch (textList[index].Trim())//根据输入的字母判断应用哪个头像

{

case "A": faceImage.sprite = face01; index++; break;

case "B": faceImage.sprite = face02; index++; break;

}

int letter = 0;

while (!cancerTyping && letter < textList[index].Length - 1)//如果跳过编辑开启，结束进程，直接输出；如果未开启，逐字输出

{

textLable.text += textList[index][letter];

letter++;

yield return new WaitForSeconds(textSpeed);

}

textLable.text = textList[index];

cancerTyping = false;//取消跳过字幕编辑

textFinished = true;//完全输出完毕

index++;

}

}

3.Event

3.1:Event00:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Event00 : MonoBehaviour

{

void Update()

{

if (gameObject.activeSelf && Input.GetKeyDown(KeyCode.V))

{

Destroy(gameObject);

}

}

}

3.2:Event10:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Event10 : MonoBehaviour

{

public GameObject player;

public GameObject Tanghuang;

private void Update()

{

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

if (dialogeNumber.dialogeNumber == 2)

{

Tanghuang.SetActive(true);

dialogeNumber.dialogeJudge = true;

}

}

}

3.3:Event11:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Event11 : MonoBehaviour

{

public GameObject player;

public GameObject Claus;

private Rigidbody2D rigidbody2d;

public float speed;

private int direction = 1;

public float timer;

void Start()

{

rigidbody2d = Claus.GetComponent<Rigidbody2D>();

}

void Update()

{

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

if (dialogeNumber.dialogeNumber == 3)

{

if (timer > 0)

{

timer -= Time.deltaTime;

}

if (timer < 0)

{

direction = 0;

dialogeNumber.dialogeJudge = true;

timer = 0;

}

Vector2 position = Claus.transform.position;

position.x += speed \* Time.deltaTime \* direction;

rigidbody2d.MovePosition(position);

}

}

}

3.4:Event12:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Event12 : MonoBehaviour

{

public GameObject player;

public GameObject Claus;

private Rigidbody2D rigidbody2d;

public float speed;

private int direction = 1;

public float timer;

void Start()

{

rigidbody2d = Claus.GetComponent<Rigidbody2D>();

}

void Update()

{

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

if (dialogeNumber.dialogeNumber == 4)

{

if (timer > 0)

{

timer -= Time.deltaTime;

}

if (timer < 0)

{

direction = 0;

Claus.SetActive(false);

dialogeNumber.dialogeJudge = true;

timer = 0;

}

Vector2 position = Claus.transform.position;

position.x += speed \* Time.deltaTime \* direction;

rigidbody2d.MovePosition(position);

}

}

}

3.5:Event13:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Event13 : MonoBehaviour

{

public GameObject player;

public GameObject tangHuang;

private Rigidbody2D rigidbody2d;

public float speed;

private int direction = 1;

public float timer;

void Start()

{

rigidbody2d = tangHuang.GetComponent<Rigidbody2D>();

}

void Update()

{

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

if (dialogeNumber.dialogeNumber == 5)

{

if (timer > 0)

{

timer -= Time.deltaTime;

}

if (timer < 0)

{

direction = 0;

tangHuang.SetActive(false);

dialogeNumber.dialogeJudge = true;

timer = 0;

}

Vector2 position = tangHuang.transform.position;

position.x += speed \* Time.deltaTime \* direction;

rigidbody2d.MovePosition(position);

}

}

}

3.6:Event14:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Event14 : MonoBehaviour

{

public GameObject player;

public GameObject mission;

private void Update()

{

PlayerController dialogeNumber = player.GetComponent<PlayerController>();

if (dialogeNumber.dialogeNumber == 6)

{

mission.SetActive(true);

dialogeNumber.dialogeJudge = true;

}

}

}

3.7:LeaveHere:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class LeaveHere : MonoBehaviour

{

public GameObject button;

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Player"))

{

button.SetActive(true);

}

}

}

4.Fighting Scripts

4.1:EnemyHealthAdd:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class EnemyHealthAdd : MonoBehaviour

{

public bool destroy;

private void OnTriggerEnter2D(Collider2D collision)

{

EnemyHealthControler enemy = collision.GetComponent<EnemyHealthControler>();

if (enemy != null)//如果敌人的判定不为空，防止玩家接触后报错

{

if (enemy.Health < enemy.maxHealth)

{

enemy.ChangeHealth(1);

if (destroy)

{

Destroy(gameObject);

}

}

}

}

}

4.2:EnemyHealthControler:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class EnemyHealthControler : MonoBehaviour

{

public int maxHealth;//最大生命值

public int currentHealth;//当前生命值

public int Health { get { return currentHealth; } }

public float timeInvincible;//无敌时间常量

private bool isInvincible;//布尔类型：用作逻辑判断

private float invincibleTimer;//计时器

private void Update()

{

if (isInvincible)

{

invincibleTimer -= Time.deltaTime;

if (invincibleTimer <= 0)

{

isInvincible = false;//无敌帧结束

}

}

}

public void ChangeHealth(int amount)

{

if (amount < 0)

{

if (isInvincible)

{

return;

}

isInvincible = true;//受伤触发无敌

invincibleTimer = timeInvincible;

}

currentHealth = Mathf.Clamp(currentHealth + amount, 0, maxHealth);//生命值的最大值与最小值

}

}

4.3:EnemyHealthReduce:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class EnemyHealthReduce : MonoBehaviour

{

public bool destroy;

private void OnTriggerStay2D(Collider2D collision)

{

EnemyHealthControler enemy = collision.GetComponent<EnemyHealthControler>();

if (enemy != null)//如果敌人的判定不为空，防止玩家接触后报错

{

enemy.ChangeHealth(-1);

if (destroy)

{

Destroy(gameObject);

}

}

}

}

4.4:PlayerHealthAdd:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerHealthAdd : MonoBehaviour

{

public bool destroy;

private void OnTriggerEnter2D(Collider2D collision)

{

PlayerHealthControler player = collision.GetComponent<PlayerHealthControler>();

if (player != null)//如果玩家的判定不为空，防止敌人接触后报错

{

if (player.Health < player.maxHealth)

{

player.ChangeHealth(1);

if (destroy)

{

Destroy(gameObject);

}

}

}

}

}

4.5:PlayerHealthControler:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerHealthControler : MonoBehaviour

{

public int maxHealth;//最大生命值

public int currentHealth;//当前生命值

public int Health { get { return currentHealth; } }

public float timeInvincible;//无敌时间常量

private bool isInvincible;//布尔类型：用作逻辑判断

private float invincibleTimer;//计时器

private void Update()

{

if (isInvincible)

{

invincibleTimer -= Time.deltaTime;

if (invincibleTimer <= 0)

{

isInvincible = false;//无敌帧结束

}

}

}

public void ChangeHealth(int amount)

{

if (amount < 0)

{

if (isInvincible)

{

return;

}

isInvincible = true;//受伤触发无敌

invincibleTimer = timeInvincible;

}

currentHealth = Mathf.Clamp(currentHealth + amount, 0, maxHealth);//生命值的最大值与最小值

}

}

4.6:PlayerHealthReduce:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerHealthReduce : MonoBehaviour

{

public bool destroy;

private void OnTriggerStay2D(Collider2D collision)

{

PlayerHealthControler player = collision.GetComponent<PlayerHealthControler>();

if (player != null)//如果玩家的判定不为空，防止敌人接触后报错

{

player.ChangeHealth(-1);

if (destroy)

{

Destroy(gameObject);

}

}

}

}

4.7:PlayerController:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerController : MonoBehaviour

{

public static PlayerController instance;

private new Rigidbody2D rigidbody2D;

public float speed;

private void Awake()

{

if (instance == null)

{

instance = this;

}

else

{

if (instance != this)

{

Destroy(gameObject);

}

}

DontDestroyOnLoad(gameObject);

}

void Start()

{

Application.targetFrameRate = 60;

rigidbody2D = GetComponent<Rigidbody2D>();

}

public PlayerController(Rigidbody2D rigidbody2D)

{

this.rigidbody2D = rigidbody2D;

}

void Update()

{

float horizontal = Input.GetAxis("Horizontal");

float vertical = Input.GetAxis("Vertical");

Vector2 position = transform.position;

position.x += speed \* horizontal \* Time.deltaTime;

position.y += speed \* vertical \* Time.deltaTime;

rigidbody2D.MovePosition(position);

}

}

5.Game Control Scripts

5.1:OpenBag:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class OpenBag : MonoBehaviour

{

public GameObject myBag;

bool isopen;

private void Update()

{

OpenMyBag();

if (myBag.activeSelf)

{

Time.timeScale = 0.0f;

}

else

{

Time.timeScale = 1.0f;

}

}

void OpenMyBag()

{

isopen = myBag.activeSelf;

if (Input.GetKeyDown(KeyCode.B))

{

isopen = !isopen;

myBag.SetActive(isopen);

}

}

}

5.2:OpenAttribute:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class OpenAttribute : MonoBehaviour

{

public GameObject myAttribute;

bool isopen = true;

private void Update()

{

OpenMyAttribute();

if (myAttribute.activeSelf)

{

Time.timeScale = 0.0f;

}

else

{

Time.timeScale = 1.0f;

}

}

void OpenMyAttribute()

{

isopen = myAttribute.activeSelf;

if (Input.GetKeyDown(KeyCode.V))

{

isopen = !isopen;

myAttribute.SetActive(isopen);

}

}

}

5.3:PlayerController:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class PlayerController : MonoBehaviour

{

public static PlayerController instance;

private new Rigidbody2D rigidbody2D;

public float speed;

public int dialogeNumber;

public bool dialogeJudge = false;

private Vector2 lookDirection = new Vector2(-1, 0);

private Animator animator;

public string scenePassword;

private void Awake()

{

if (instance == null)

{

instance = this;

}

else

{

if (instance != this)

{

Destroy(gameObject);

}

}

}

void Start()

{

Application.targetFrameRate = 60;

rigidbody2D = GetComponent<Rigidbody2D>();

animator = GetComponent<Animator>();

}

public PlayerController(Rigidbody2D rigidbody2D)

{

this.rigidbody2D = rigidbody2D;

}

void Update()

{

float horizontal = Input.GetAxis("Horizontal");

float vertical = Input.GetAxis("Vertical");

Vector2 move = new Vector2(horizontal, vertical);

if (move != Vector2.zero)

{

animator.SetFloat("MoveX", move.x);

animator.SetFloat("MoveY", move.y);

}

animator.SetFloat("Speed", move.magnitude);

Vector2 position = transform.position;

position.x += speed \* horizontal \* Time.deltaTime;

position.y += speed \* vertical \* Time.deltaTime;

rigidbody2D.MovePosition(position);

}

}

6.Inventory Scripts

6.1:Inventory:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[CreateAssetMenu(fileName = "New Inventory", menuName = "Inventory/New Inventory")]

public class Inventory : ScriptableObject

{

public List<Item> itemList = new List<Item>();

}

6.2:InventoryManager:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class InventoryManager : MonoBehaviour

{

static InventoryManager instance;

public Inventory mybag;

public GameObject emptySlot;

public Text itemInformation;

public GameObject slotGrid;

public List<GameObject> slots = new List<GameObject>();

void Awake()

{

if (instance != null)

Destroy(this);

instance = this;

}

private void OnEnable()

{

RefreshItem();

instance.itemInformation.text = "";

}

public static void UpdateItemInformation(string itemDescription)

{

instance.itemInformation.text = itemDescription;

}

//全删刷新图标

public static void RefreshItem()

{

//循环删除slotgrid下的物品

for (int i = 0; i < instance.slotGrid.transform.childCount; i++)

{

if (instance.slotGrid.transform.childCount == 0)

break;

Destroy(instance.slotGrid.transform.GetChild(i).gameObject);

instance.slots.Clear();

}

//重新生成物品slot

for (int i = 0; i < instance.mybag.itemList.Count; i++)

{

instance.slots.Add(Instantiate(instance.emptySlot));

instance.slots[i].transform.SetParent(instance.slotGrid.transform);

instance.slots[i].GetComponent<Slot>().slotID = i;

instance.slots[i].GetComponent<Slot>().SetUpSlot(instance.mybag.itemList[i]);

}

}

}

6.3:Item:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

[CreateAssetMenu(fileName = "New Item", menuName = "Inventory/New Item")]

public class Item : ScriptableObject

{

public string itemName;

public Sprite itemImage;

public int itemNumber = 1;

[TextArea]

public string itemInformation;

public bool equipment;

public bool medicine;

}

6.4:ItemForSearch:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class ItemForSearch : MonoBehaviour

{

public Item thisItem;

public Inventory playerInventory;

public GameObject button;

private void OnTriggerEnter2D(Collider2D other)

{

if (other.CompareTag("Player"))

{

button.SetActive(true);

}

}

private void OnTriggerExit2D(Collider2D other)

{

button.SetActive(false);

}

private void Update()

{

if (button.activeSelf && Input.GetKeyDown(KeyCode.E))

{

AddNewItem();

Destroy(gameObject);

}

}

public void AddNewItem()

{

if (!playerInventory.itemList.Contains(thisItem))

{

for (int i = 0; i < playerInventory.itemList.Count; i++)

{

if (playerInventory.itemList[i] == null)

{

playerInventory.itemList[i] = thisItem;

break;

}

}

}

else

{

thisItem.itemNumber += 1;

}

InventoryManager.RefreshItem();

}

}

6.5:ItemOnDrag:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.EventSystems;

public class ItemOnDrag : MonoBehaviour, IBeginDragHandler, IDragHandler, IEndDragHandler

{

public Transform originalParent;

public Inventory myBag;

private int currentItemID;//当前物品ID

public void OnBeginDrag(PointerEventData eventData)

{

originalParent = transform.parent;

currentItemID = originalParent.GetComponent<Slot>().slotID;

transform.SetParent(transform.parent.parent);

transform.position = eventData.position;//跟随鼠标位置

GetComponent<CanvasGroup>().blocksRaycasts = false;//关闭该物体射线碰撞

}

public void OnDrag(PointerEventData eventData)

{

transform.position = eventData.position;

}

void IEndDragHandler.OnEndDrag(PointerEventData eventData)

{

//无效检测，拖拽物体归位

if (eventData.pointerCurrentRaycast.gameObject == null || !eventData.pointerCurrentRaycast.gameObject.CompareTag("Slot"))

{

transform.position = originalParent.position;

transform.SetParent(originalParent);

GetComponent<CanvasGroup>().blocksRaycasts = true;

return;

}

//位置互换

if (eventData.pointerCurrentRaycast.gameObject.name == "Item Image")

{

transform.SetParent(eventData.pointerCurrentRaycast.gameObject.transform.parent.parent);

transform.position = eventData.pointerCurrentRaycast.gameObject.transform.parent.parent.position;

//ID改变，既数据中换位

var temp = myBag.itemList[currentItemID];

myBag.itemList[currentItemID] = myBag.itemList[eventData.pointerCurrentRaycast.gameObject.GetComponentInParent<Slot>().slotID];

myBag.itemList[eventData.pointerCurrentRaycast.gameObject.GetComponentInParent<Slot>().slotID] = temp;

eventData.pointerCurrentRaycast.gameObject.transform.parent.position = originalParent.position;

eventData.pointerCurrentRaycast.gameObject.transform.parent.SetParent(originalParent);

GetComponent<CanvasGroup>().blocksRaycasts = true;

return;

}

//挂在检测到的slot下面

transform.SetParent(eventData.pointerCurrentRaycast.gameObject.transform);

transform.position = eventData.pointerCurrentRaycast.gameObject.transform.position;

myBag.itemList[eventData.pointerCurrentRaycast.gameObject.GetComponentInParent<Slot>().slotID] = myBag.itemList[currentItemID];

//防止不换位后删除物品

if (eventData.pointerCurrentRaycast.gameObject.GetComponent<Slot>().slotID != currentItemID)

{

myBag.itemList[currentItemID] = null;

}

GetComponent<CanvasGroup>().blocksRaycasts = true;

}

}

6.6:ItemOnWorld;

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class ItemOnWorld : MonoBehaviour

{

public Item thisItem;

public Inventory playerInventory;

public void OnTriggerEnter2D(Collider2D other)

{

if (other.gameObject.CompareTag("Player"))

{

AddNewItem();

Destroy(gameObject);

}

}

public void AddNewItem()

{

if (!playerInventory.itemList.Contains(thisItem))

{

for (int i = 0; i < playerInventory.itemList.Count; i++)

{

if (playerInventory.itemList[i] == null)

{

playerInventory.itemList[i] = thisItem;

break;

}

}

}

else

{

thisItem.itemNumber += 1;

}

InventoryManager.RefreshItem();

}

}

6.7:Slot:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class Slot : MonoBehaviour

{

public int slotID;

public Image slotImage;

public Text slotNumber;

public string slotInformation;

public GameObject useBotton;

public bool slotEquipment;

public GameObject itemInSlot;

public GameObject itemIsUsing;

bool itemUsing = false;

public void ItemOnClicked()

{

InventoryManager.UpdateItemInformation(slotInformation);

UseJudge();

}

public void SetUpSlot(Item item)

{

if (item == null)

{

itemInSlot.SetActive(false);

return;

}

slotImage.sprite = item.itemImage;

slotNumber.text = item.itemNumber.ToString();

slotInformation = item.itemInformation;

slotEquipment = item.equipment;

}

private void UseJudge()

{

if (slotEquipment)

{

useBotton.SetActive(true);

}

else

{

useBotton.SetActive(false);

}

}

public void UseBottonOnClicked()

{

if (slotEquipment)

{

itemUsing = !itemUsing;

itemIsUsing.SetActive(itemUsing);

}

}

}