郑鸿棣代码部分：

1.EasyJoystick

using System;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.EventSystems;

[RequireComponent(typeof(RectTransform))]

public class EasyJoystick : MonoBehaviour, IPointerDownHandler, IDragHandler, IPointerUpHandler

{

public RectTransform stick; //stick image;

public float returnRate = 15.0F; //default position returning speed;

public float dragRadius = 65.0f; //drag radius;

public AlphaControll colorAlpha;

public event Action<EasyJoystick, Vector2> OnStartJoystickMovement;

public event Action<EasyJoystick, Vector2> OnJoystickMovement;

public event Action<EasyJoystick> OnEndJoystickMovement;

private bool \_returnHandle, pressed, isEnabled = true;

private RectTransform \_canvas;

private Vector3 globalStickPos;

private Vector2 stickOffset;

private CanvasGroup canvasGroup;

Vector2 Coordinates

{

get

{

if (stick.anchoredPosition.magnitude < dragRadius)

return stick.anchoredPosition / dragRadius;

return stick.anchoredPosition.normalized;

}

}

void IPointerDownHandler.OnPointerDown(PointerEventData eventData)

{

pressed = true;

\_returnHandle = false;

stickOffset = GetJoystickOffset(eventData);

stick.anchoredPosition = stickOffset;

if (OnStartJoystickMovement != null)

OnStartJoystickMovement(this, Coordinates);

}

void IDragHandler.OnDrag(PointerEventData eventData)

{

stickOffset = GetJoystickOffset(eventData);

stick.anchoredPosition = stickOffset;

if (OnJoystickMovement != null)

OnJoystickMovement(this, Coordinates);

}

void IPointerUpHandler.OnPointerUp(PointerEventData eventData)

{

pressed = false;

\_returnHandle = true;

if (OnEndJoystickMovement != null)

OnEndJoystickMovement(this);

}

private Vector2 GetJoystickOffset(PointerEventData eventData)

{

if (RectTransformUtility.ScreenPointToWorldPointInRectangle(\_canvas, eventData.position, eventData.pressEventCamera, out globalStickPos))

stick.position = globalStickPos;

var handleOffset = stick.anchoredPosition;

if (handleOffset.magnitude > dragRadius)

{

handleOffset = handleOffset.normalized \* dragRadius;

stick.anchoredPosition = handleOffset;

}

return handleOffset;

}

private void Start()

{

canvasGroup = GetComponent ("CanvasGroup") as CanvasGroup;

\_returnHandle = true;

var touchZone = GetComponent("RectTransform") as RectTransform;

touchZone.pivot = Vector2.one \* 0.5F;

stick.transform.SetParent(transform);

var curTransform = transform;

do

{

if (curTransform.GetComponent<Canvas>() != null)

{

\_canvas = curTransform.GetComponent("RectTransform") as RectTransform;;

break;

}

curTransform = transform.parent;

}

while (curTransform != null);

}

private void FixedUpdate()

{

if (\_returnHandle)

if (stick.anchoredPosition.magnitude > Mathf.Epsilon)

stick.anchoredPosition -= new Vector2(stick.anchoredPosition.x \* returnRate,

stick.anchoredPosition.y \* returnRate) \* Time.deltaTime;

else

\_returnHandle = false;

switch(isEnabled)

{

case true:

canvasGroup.alpha = pressed ? colorAlpha.pressedAlpha : colorAlpha.idleAlpha;

canvasGroup.interactable = canvasGroup.blocksRaycasts = true;

break;

case false:

canvasGroup.alpha = 0;

canvasGroup.interactable = canvasGroup.blocksRaycasts = false;

break;

}

}

public Vector3 MoveInput()

{

return new Vector3 (Coordinates.x, 0, Coordinates.y);

}

public void Rotate(Transform transformToRotate, float speed)

{

if(Coordinates != Vector2.zero)

transformToRotate.rotation = Quaternion.Slerp (transformToRotate.rotation,

Quaternion.LookRotation (new Vector3 (Coordinates.x, 0, Coordinates.y)),

speed \* Time.deltaTime);

}

public bool IsPressed()

{

return pressed;

}

public void Enable(bool enable)

{

isEnabled = enable;

}

}

[Serializable]

public class AlphaControll

{

public float idleAlpha = 0.5F, pressedAlpha = 1.0F;

}

测评人：徐余浩

1. 对象“returnRate”，” returnHandle”等符合我们的代码规范。
2. 类方法“FixedUpdate”，“IPointerDownHandler”实现了方法所要实现的内容，并且按照代码规范编写。

3. 测试时，对于虚拟摇杆的使用，包括在地图边界，一些不合理的位置，以及对于单位体积碰撞等问题都进行了人工的白盒（条件覆盖），黑盒（边界值），测定没有出现问题。

2.出现杆

using System.Collections.Generic;

using UnityEngine;

using System.Collections;

using UnityEngine.UI;

public class 出现杆 : MonoBehaviour {

public GameObject btnObj ;

public GameObject caidan1;

public GameObject caidan2;

public GameObject yugan;

public GameObject yu1;

public GameObject yuwenzi1;

public GameObject yu2;

public GameObject yuwenzi2;

public GameObject shibai;

public Sprite expan;

public Sprite back;

Button btn;

bool isshow=false ;

// Use this for initialization

void Start () {

caidan1.SetActive (isshow);

caidan2.SetActive (isshow);

yugan.SetActive (isshow);

btn = btnObj.GetComponent<Button>();

btn.onClick.AddListener(delegate ()

{

isshow=!isshow;

caidan1.SetActive (isshow);

caidan2.SetActive (isshow);

yu1.SetActive (false);

yu2.SetActive (false);

yuwenzi1.SetActive (false);

yuwenzi2.SetActive (false);

shibai.SetActive (false);

yugan.SetActive(isshow);

if (isshow)

{

btn.GetComponent<Image>().sprite=expan ;

}

else {

btn.GetComponent<Image>().sprite=back;

}

});

}

// Update is called once per frame

void Update () {

}

}

测评人：徐余浩

1.类中使用了中文字符“出现杆”，这个不符合我们的代码规范。

2.类方法“Start”实现了方法所要实现的内容。

3. 对象“caidan1”，“yugan”等都是中式翻译的对象，不符合我们的代码规范。

4.效果测试中，不同的场景地点实现了鱼竿的出现，像一些不能钓鱼的地方鱼竿就不出现，这个之前发现这个的bug消除，其他bug没有出现。

3.Trigger\_A

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Trigger\_A : MonoBehaviour {

public GameObject btnObj;

public GameObject btnObj1;

public GameObject btnObj2;

public GameObject yugan;

public GameObject yu1;

public GameObject yuwenzi1;

public GameObject yu2;

public GameObject yuwenzi2;

public GameObject shibai;

void OnTriggerEnter(Collider other)

{

Debug.Log(Time.time + ":进入该触发器的对象是：" + btnObj);

}

void OnTriggerStay(Collider other) //每帧调用一次OnTriggerStay()函数

{

btnObj.SetActive(true);

}

void OnTriggerExit(Collider other)

{

btnObj.SetActive(false);

yu1.SetActive (false);

yu2.SetActive (false);

yuwenzi1.SetActive (false);

yuwenzi2.SetActive (false);

shibai.SetActive (false);

yugan.SetActive(false);

btnObj1.SetActive (false);

btnObj2.SetActive (false);

}

}

测评人：徐余浩

1.对于“每帧调用一次OnTriggerStay()函数”使用了注释，方便了测试人员的测试。

2.这个是之前发现的一个严重bug“发现可以在陆地上钓鱼”而加的类，目的是判定甩杆的位置是不是可以钓鱼的地方，人工的白盒（条件覆盖），黑盒（边界值）测定没有出现问题。

4.甩杆

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using System;

public class 甩杆 : MonoBehaviour {

public GameObject btnObj;

public GameObject btnObj2;

public GameObject yugan;

public GameObject yu1;

public GameObject yuwenzi1;

public GameObject yu2;

public GameObject yuwenzi2;

public GameObject shibai;

public GameObject caidan;

public Sprite expan;

public Sprite back;

Button btn;

Button btn2;

// Use this for initialization

void Start () {

btn = btnObj.GetComponent<Button>();

btn.onClick.AddListener(delegate ()

{

yu1.SetActive (false);

yu2.SetActive (false);

yuwenzi1.SetActive (false);

yuwenzi2.SetActive (false);

shibai.SetActive (false);

StartCoroutine("TriggerSplash");

if (true)

{

btn.GetComponent<Image>().sprite=expan ;

yugan.transform.localEulerAngles = new Vector3 (65.0f, 0.0f, 0.0f);

yugan.transform.localPosition = new Vector3(3.0f,0.0f,-1f);

}

});

}

// Update is called once per frame

void Update () {

}

//

// IEnumerator TriggerSplash()

// {

// caidan.SetActive (false);

// float pr = UnityEngine.Random.Range (3, 5);

// btn2 = btnObj2.GetComponent<Button> ();

// if (caidan == false && btn2.IsInteractable) {

//

// btn2.onClick.AddListener (delegate () {

// yu1.SetActive (false);

// yu2.SetActive (false);

// yuwenzi1.SetActive (false);

// yuwenzi2.SetActive (false);

// shibai.SetActive (true);

// StopCoroutine ("TriggerSplash");

// });

// }

// yield return new WaitForSeconds (pr);

// caidan.SetActive (true);

// if (caidan == true) {

// btn2 = btnObj2.GetComponent<Button> ();

// btn2.onClick.AddListener (delegate () {

//

// yu1.SetActive (false);

// yu2.SetActive (false);

// yuwenzi1.SetActive (false);

// yuwenzi2.SetActive (false);

// shibai.SetActive (false);

// if (true) {

// int pr2 = UnityEngine.Random.Range (0, 101);

// if (pr2 < 5) {

// shibai.SetActive (true);

// } else if (pr2 < 30) {

// yu2.SetActive (true);

// yuwenzi2.SetActive (true);

// } else {

// yu1.SetActive (true);

// yuwenzi1.SetActive (true);

// }

// btn.GetComponent<Image> ().sprite = expan;

// yugan.transform.localEulerAngles = new Vector3 (0.0f, 0.0f, 0.0f);

// yugan.transform.localPosition = new Vector3 (3.3f, 0.0f, 5f);

// }

// });

// }

//

// yield return new WaitForSeconds (1f);

// caidan.SetActive (false);

// if (caidan == false) {

// btn2 = btnObj2.GetComponent<Button> ();

// btn2.onClick.AddListener (delegate () {

//

// yu1.SetActive (false);

// yu2.SetActive (false);

// yuwenzi1.SetActive (false);

// yuwenzi2.SetActive (false);

// shibai.SetActive (true);

// StopCoroutine ("TriggerSplash");

// });

// }

// }

IEnumerator TriggerSplash()

{

bool bl = false;

bool bl1 = true;

yu1.SetActive (false);

yu2.SetActive (false);

yuwenzi1.SetActive (false);

yuwenzi2.SetActive (false);

shibai.SetActive (false);

caidan.SetActive (false);

float pr = UnityEngine.Random.Range (3, 5);

btn2 = btnObj2.GetComponent<Button> ();

btn2.onClick.AddListener (delegate () {

bl = false;

bl1 = false;

btn.GetComponent<Image> ().sprite = expan;

yugan.transform.localEulerAngles = new Vector3 (0.0f, 0.0f, 0.0f);

yugan.transform.localPosition = new Vector3 (3.3f, 0.0f, 5f);

});

if (bl1 == true) {

yield return new WaitForSeconds (pr);

caidan.SetActive (true);

}

btn2 = btnObj2.GetComponent<Button> ();

btn2.onClick.AddListener (delegate () {

bl = true;

btn.GetComponent<Image> ().sprite = expan;

yugan.transform.localEulerAngles = new Vector3 (0.0f, 0.0f, 0.0f);

yugan.transform.localPosition = new Vector3 (3.3f, 0.0f, 5f);

});

if(bl1 == true)

yield return new WaitForSeconds (1.5f);

caidan.SetActive (false);

shibai.SetActive (false);

if (bl == true) {

int pr2 = UnityEngine.Random.Range (0, 101);

if (pr2 < 5) {

shibai.SetActive (true);

} else if (pr2 < 40) {

yu2.SetActive (true);

yuwenzi2.SetActive (true);

} else {

yu1.SetActive (true);

yuwenzi1.SetActive (true);

}

} else {

yu1.SetActive (false);

yu2.SetActive (false);

yuwenzi1.SetActive (false);

yuwenzi2.SetActive (false);

shibai.SetActive (true);

btn.GetComponent<Image> ().sprite = expan;

yugan.transform.localEulerAngles = new Vector3 (0.0f, 0.0f, 0.0f);

yugan.transform.localPosition = new Vector3 (3.3f, 0.0f, 5f);

}

}

}

徐余浩代码部分：

1.equip

using UnityEngine;

using System.Collections;

using UnityEngine.UI;

using Game\_Manager;

public class equip : MonoBehaviour {

private Game\_Scene\_Manager gsm;

private Image equip\_image;

public int mouse\_type;

public Sprite weapon; // 为此背包中相应的物品

public Sprite UISprite;

public Color weapon\_color;

public Color UISprite\_color;

void Awake()

{

gsm = Game\_Scene\_Manager.GetInstance();

equip\_image = GetComponent<Image>();

}

public void On\_equip\_Button() {

Debug.Log("equip click");

int MouseType = gsm.GetMouse().GetMouseType(); // 得到鼠标上的mousetype

if (equip\_image.sprite == weapon && MouseType == 0) // 取走背包中的物品，当携带区含有物品并且mousetype=0鼠标上没有物品

{

equip\_image.sprite = UISprite;

equip\_image.color = UISprite\_color;

gsm.GetMouse().SetMouseType(mouse\_type);

}

else

{

// 只有相同各类型的物品能够放到相应的携带区，并且不能够重复携带

if (mouse\_type == MouseType && equip\_image.sprite != weapon) {

// 将物品佩戴到携带区中

equip\_image.sprite = weapon;

equip\_image.color = weapon\_color;

mouse\_type = MouseType;

gsm.GetMouse().SetMouseType(0);

}

}

}

// Use this for initialization

void Start () {

}

// Update is called once per frame

void Update () {

// 防止重复物品

if (mouse\_type == 1 && gsm.GetHair() == 1)

{

Debug.Log("mouse\_type");

gsm.SetHair(0); // 已携带物品不能再携带

equip\_image.sprite = weapon;

equip\_image.color = weapon\_color;

} else if (mouse\_type == 2 && gsm.GetWeapon() == 1)

{

gsm.SetWeapon(0); // 已装备鱼竿不能再携带

equip\_image.sprite = weapon;

equip\_image.color = weapon\_color;

} else if (mouse\_type == 3 && gsm.GetFoot() == 1)

{

gsm.SetFoot(0); // 已携带鱼线

equip\_image.sprite = weapon;

equip\_image.color = weapon\_color;

}

}

}

using UnityEngine;

using System.Collections;

using Game\_Manager;

using UnityEngine.UI;

public class Mouse\_Image : MonoBehaviour {

private Game\_Scene\_Manager gsm;

private Image mouse\_image;

private int mouse\_type = 0;

public Sprite none;

public Sprite hair; // 保存相应的物品

public Sprite weapon; // 保存相应的物品

public Sprite foot; // 保存相应的物品

public Color None;

public Color NotNone;

public Camera cam;

void Awake() {

gsm = Game\_Scene\_Manager.GetInstance();

gsm.SetMouse(this);

mouse\_image = GetComponent<Image>();

}

public int GetMouseType() {

return mouse\_type;

}

public void SetMouseType(int Mouse\_type) {

mouse\_type = Mouse\_type;

}

void Update () {

if (mouse\_type == 0) // 每一帧进行更新，检查鼠标上是否需要有物品，根据mousetype更新，如果mousetype为0则物品无

{

mouse\_image.sprite = none;

mouse\_image.color = None;

}

else

{ // mousetype不为零，根据mousetype加上相应物品

//Debug.Log("I am mouse image");

//Debug.Log(mouse\_type);

mouse\_image.color = new Color(1F, 1F, 1F, 1F);

//Debug.Log(mouse\_image.color);

if (mouse\_type == 1) mouse\_image.sprite = hair;

else if (mouse\_type == 2) mouse\_image.sprite = weapon;

else if (mouse\_type == 3) mouse\_image.sprite = foot;

}

transform.position = new Vector3 (Input.mousePosition.x-600, Input.mousePosition.y-230, 0);

}

}

2. Game\_Scene

using UnityEngine;

using System.Collections;

using Game\_Manager;

namespace Game\_Manager {

public class Game\_Scene\_Manager : System.Object {

private static Game\_Scene\_Manager \_instance;

private static Mouse\_Image \_Mouse;

private int IsHair = 0;

private int IsWeapon = 0;

private int IsFoot = 0;

public static Game\_Scene\_Manager GetInstance() {

if (\_instance == null) {

\_instance = new Game\_Scene\_Manager();

}

return \_instance;

}

public void SetMouse(Mouse\_Image \_mouse) {

if (\_Mouse == null) {

\_Mouse = \_mouse;

}

}

public Mouse\_Image GetMouse() {

return \_Mouse;

}

public void GenAll() {

IsFoot = 1;

IsHair = 1;

IsWeapon = 1;

}

public int GetHair() { return IsHair; }

public int GetWeapon() { return IsWeapon; }

public int GetFoot() { return IsFoot; }

public void SetHair(int a) { IsHair = a; }

public void SetWeapon(int a) { IsWeapon = a; }

public void SetFoot(int a) { IsFoot = a; }

}

}

public class Mouse : MonoBehaviour {

// Use this for initialization

void Start () {

}

// Update is called once per frame

void Update () {

}

}

3. MyBag

using UnityEngine;

using System.Collections;

using UnityEngine.UI;

using Game\_Manager;

public class MyBag : MonoBehaviour {

private Game\_Scene\_Manager gsm;

private Image bag\_image;

public int mouse\_type = 0;

public Sprite hair;

public Sprite weapon;

public Sprite foot;

public Sprite UISprite;

public Color weapon\_color;

public Color UISprite\_color;

void Awake()

{

gsm = Game\_Scene\_Manager.GetInstance();

bag\_image = GetComponent<Image>();

}

public void On\_equip\_Button()

{

Debug.Log("my bag click!");

int MouseType = gsm.GetMouse().GetMouseType(); // 得到鼠标目前的mousetype

if (bag\_image.sprite != UISprite && MouseType == 0) // 若鼠标没有图片在上面，并且bag的image不为空有物品，则取走bag\_image的物品

{

Debug.Log(mouse\_type);

bag\_image.sprite = UISprite;

bag\_image.color = UISprite\_color;

gsm.GetMouse().SetMouseType(mouse\_type); // 将当前物品的type给鼠标

mouse\_type = 0; // 此背包的mousetype变为0，则当前背包啥都没有

}

else

{ // 若鼠标上有物品，则改背包的image sprite改变，根据type变为不同物品图片

Debug.Log("my bag equipped!");

if (MouseType == 1) bag\_image.sprite = hair;

else if (MouseType == 2) bag\_image.sprite = weapon;

else if (MouseType == 3) bag\_image.sprite = foot;

mouse\_type = MouseType; // mousetype变为鼠标的mousetype

bag\_image.color = weapon\_color; // 有物品

gsm.GetMouse().SetMouseType(0); // 鼠标物品消失

}

}

}

测试人：陈瑜安

我测试的是组员徐余浩负责的部分，他负责的部分是背包设计和钓鱼百科。

在我测试的过程中发现，进入游戏后再使用背包，拖动里面的物品，会导致出现不明白色方框，已反馈给他。但是由于技术问题，这个bug不知道是哪里出了问题，所以没有解决。钓鱼百科做的比较详细，做的也不错，就是可能文本太多略微有些卡顿。总的来说，完成的还是不错的。

代码上来看，标识了文字，不会让人看不懂，并且代码本身逻辑较为清晰，没有过于繁杂的判断或者循环，减少了程序出现问题的可行性。易于修改，符合低耦合，高内聚。部分函数通过简单修改可以直接用于其他相似环境，可重用性较好。

陈瑜安代码部分：

1.消失

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class 消失 : MonoBehaviour {

public GameObject caidan1;

bool isshow=false ;

// Use this for initialization

void Start () {

caidan1.SetActive (isshow);

}

// Update is called once per frame

void Update () {

}

}

2. 渔具按钮

using UnityEngine;

using System.Collections;

using UnityEngine.UI;//注意这个不能少

//using UnityEditor.Sprites ;

public class 渔具按钮 : MonoBehaviour {

//public GameObject Gmenue;

public GameObject btnObj ;

public GameObject caidan;

public Sprite expan;

public Sprite back;

Button btn;

bool isshow=false ;

// Use this for initialization

void Start () {

caidan.SetActive (isshow);

btn = btnObj.GetComponent<Button>();

btn.onClick.AddListener(delegate ()

{

isshow=!isshow;

caidan.SetActive (isshow);

if (isshow)

{

btn.GetComponent<Image>().sprite=expan ;

}

else {

btn.GetComponent<Image>().sprite=back;

}

});

}

// Update is called once per frame

void Update () {

}

}

3. 转跳场景选择

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class 转跳场景选择 : MonoBehaviour {

public Button mButton;

// Use this for initialization

void Start () {

//Gets ButtonMount

Button btnMount = mButton.GetComponent<Button>();

//add a listener to ButtonMount, executing TaskOnClick() when click ButtonMount

btnMount.onClick.AddListener(TaskOnClick);

}

void TaskOnClick()

{

//Loading Scene1

UnityEngine.SceneManagement.SceneManager.LoadScene(1);

}

}

4. 转跳钓鱼小知识

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class 转跳钓鱼小知识 : MonoBehaviour {

public Button mButton;

// Use this for initialization

void Start () {

//Gets ButtonMount

Button btnMount = mButton.GetComponent<Button>();

//add a listener to ButtonMount, executing TaskOnClick() when click ButtonMount

btnMount.onClick.AddListener(TaskOnClick);

}

void TaskOnClick()

{

//Loading Scene1

UnityEngine.SceneManagement.SceneManager.LoadScene(5);

}

}

5. 转跳渔具商店

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class 转跳渔具商店 : MonoBehaviour {

public Button mButton;

// Use this for initialization

void Start () {

//Gets ButtonMount

Button btnMount = mButton.GetComponent<Button>();

//add a listener to ButtonMount, executing TaskOnClick() when click ButtonMount

btnMount.onClick.AddListener(TaskOnClick);

}

void TaskOnClick()

{

//Loading Scene1

UnityEngine.SceneManagement.SceneManager.LoadScene(3);

}

}

测试人：郑鸿棣

1.这些类名称如“转跳渔具商店”等都没有采用代码规范来写。

2.“mButton”等对象使用了代码规范来写。

3.代码测试方面，这几个代码实现的是每个页面的跳转，所以代码的相似度较高，部分函数通过简单修改可以直接用于其他相似环境，可重用性较好。检测比较方便。点击了每个按钮，这写代码都能准确的跳转到相应的界面中去，不存在bug。