《RPG 游戏开发》课程设计报告

《黑暗之光》的设计与实现

完成日期 2020 年 6 月 22 日- 2020 年 7 月 5 日

目录

一、游戏概述

2.1 游戏名称： 黑暗之光

2.2 运行环境

Windows7 以及以上系统。

游戏故事情节

当整个世界被黑暗所笼罩，是时候拿起你的武器去战斗了，勇士！在这个全新的大陆中，你可以扮演成法师或者战士，然后开始你的成长之路，拯救你的村庄！ 在成长的途中，你会遇到未知的怪物，全新的装备，不同的敌人，强大的 BOSS。你将不断的变强，直至打败大 BOSS，拯救大家美好的家园。（加油，奥里给！）

2.3 游戏风格

3D 角色扮演 RPG 类

2.4 游戏定位

16-30 岁人群

二、玩家心理分析与游戏性

游戏，作为第九艺术，深受着许多人的喜爱，人们都喜欢在闲暇之余来一把游戏，因此游戏产业的诞生是必然的。游戏，被设计出来的目的就是为了让玩家开心，因为一个游戏是否成功，取决于这款游戏的设计，并且需对该游戏的受众群体有深入了解，我们需要知道怎样的游戏更受哪一种群体的玩家的喜爱，玩家又最不喜爱哪一类游戏。设计游戏，首先自己要熟悉游戏，不熟悉游戏的游戏设计师，就好比不会游泳的游泳教练。需要自身对玩游戏有足够的经验，对游戏有独特的见解，才能够设计出好的游戏，深受玩家喜爱的游戏。

像是这款黑暗之光的游戏，受众群体应锁定在青少年身上，这款游戏具有较为卡通的画风，较为简单的玩法，比较适合比较有耐心的青少年们，因为这款游戏的玩法也比较单调，仅仅具备了 RPG 游戏的基础部分，接受任务，刷怪，升级，完成任务，购买武器，使用道具。

年龄稍微大一些的玩家可能经常接触此类游戏，已经对此类游戏并无太多感觉，因此我们应该将此款游戏的受众群体锁定于青少年玩家，并尽可能的丰富游戏玩法，使在长期进行游戏后不会感到过于单调甚至厌烦。

设计一款 Q 版 RPG 游戏，首先画风需要统一，体现于场景的风

格，人物的画风，游戏特效的统一风格以及配乐音效的风格，当然也缺少不了风格一致的 UI 界面，如果一个游戏的风格不统一且杂乱无章的话，会让玩家觉得某些部分特别突兀，会影响游戏的感官。

在风格确定好以后就应该确定游戏的玩法，是一款怎样的 RPG 游戏，因为已经提前确定了是一款 RPG 游戏，但是 RPG 游戏往下还分为许多类，分别是传统角色扮演类游戏（RPG），代表作有口袋妖怪系列，最终幻想系列，策略角色扮演类（SRPG）,代表作有火焰纹章系列，战场的女武神,动作扮演类游戏（ARPG），代表作有暗黑破坏神系列，王国之心系列，桌面角色扮演类游戏（TRPG）,此类游戏接触较少，据百科所讲此类型的理念由《龙与地下城（DND）》首创,大型多人在线角色扮演游戏（MMORPG），这类游戏玩家们接触的就比较多了，例如我们熟知的魔兽世界，梦幻西游，诸如此类。

这款游戏我们确定下来的游戏类型是 ARPG 类，根据提供的素材，我们认定这款游戏与 ARPG 较为符合。确认好游戏类型就要对游戏玩法进行设计了，首先 RPG 游戏应该需不止一种职业选择，根据素材包判定，职业分为法师和剑士。这款游戏应该还具有装

备系统，购买系统，以及简易的任务系统。

三、游戏玩法

根据玩家扮演游戏角色，通过鼠标操作与键盘互动的形式，完成 NPC 布置的任务进行打怪升级，探险游戏副本等多项活动来提升个人属性与获得新的装备，学习更厉害的技能，最终获得神器，完成任务并打败 BOSS。

操作：

相关操作	说明
鼠标左键	角色移动
鼠标右键	取消操作
鼠标中键	角色缩放
TAB	背包快捷栏
Z	装备快捷栏
X	角色属性栏
C	技能快捷栏
ESE	设置
1	技能 1
2	技能 2
3	技能 3
4	技能 4
5	技能 5
6	技能 6
M	小地图放大/缩小
R	进行复活
双击鼠标左键	加速奔跑
鼠标左键	攻击小狼
Q/W/E	使用药品

使用到的计算公式：

最终伤害=初始伤害+自身攻击力

减伤=所受伤害-自身防御力*5%

攻速=初始攻速 - (自身攻速)*0.2%

动画速度=初始速度+（自身攻速）*0.2%

四、完成主要内容

5.1 角色信息

5.11 显示头像

5.12 血条和蓝条

5.13 属性状态（技能对属性的影响）

5.14 角色控制（行走和奔跑）

5.15 角色穿戴效果（穿戴后对角色属性的影响）

5.2 地图

5.21 场景搭建

5.22 小地图的显示（放大、缩小、标识）

5.23 添加环境碰撞器 Collider

5.3 商店系统：

5.31 药品商店

（1）药类物品的数值信息

（2）添加药类物品的快捷键和快捷栏

（3）药类物品的使用（恢复相应的 hp 或 mp）

（4）添加药品商人 NPC

（5）物品的购买和出售

5.32 武器商店

（1）添加武器商人 NPC

（2）设计武器商店 UI

（3）创建装备列表和装备栏

（4）物品的购买和出售

5.4 敌人开发：

5.41 环境添加敌人

5.42 敌人的随机移动巡逻

5.43 打中敌人攻击效果（Miss 和伤害数值显示）

5.44 敌人自动攻击、分析属性和自动跟踪

5.45 敌人自动攻击

5.46 敌人的孵化器

5.47 敌人死亡后经验获取和任务奖励

5.5 战斗系统：

5.51 控制主角朝向敌人移动和动画播放

5.52 控制主角对敌人的攻击动画实现

5.53 添加攻击的特效和掉血的效果

5.53 主角受到攻击的效果（掉血、显示红色警告）

5.54 添加技能额外信息的存储

5.6 技能系统：

5.61 技能背包设计

5.62 技能栏设计

5.63 技能数值信息

5.64 技能状态（解锁、未解锁、等级限制）

5.65 技能释放（使用增益、增强技能、单个目标技能、群体技能）

额外功能实现：

使用 Dotween 插件制作部分 UI 动画

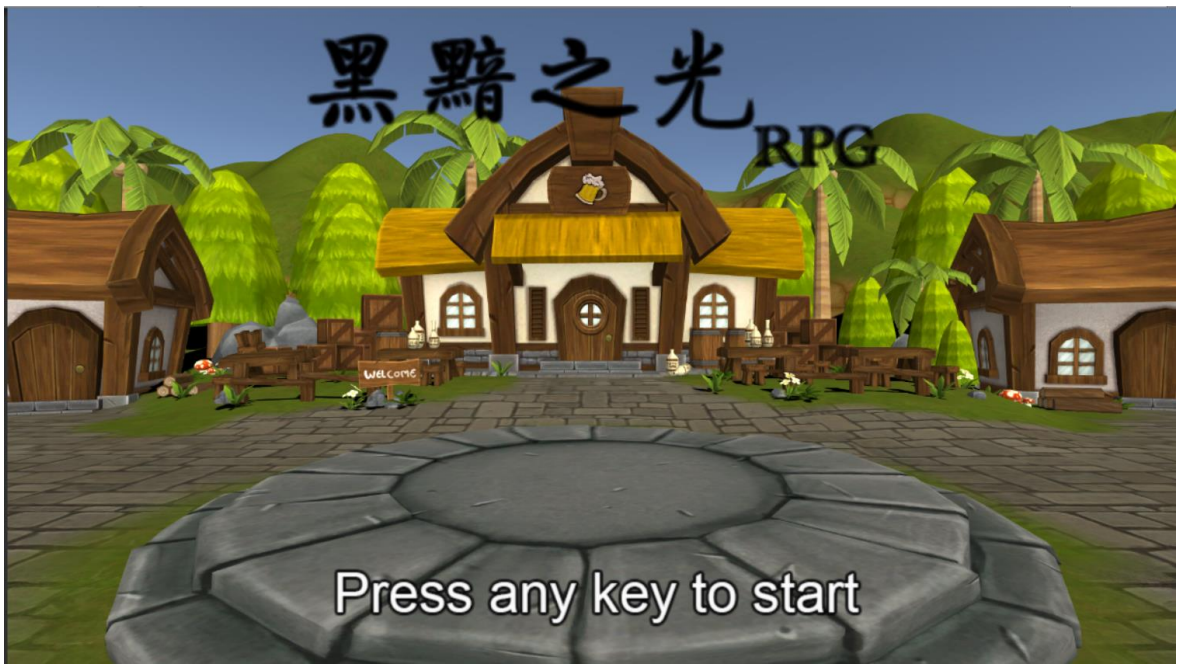
音效、音乐开启与关闭

小地图的 UI 放大与缩小（不是相机的视野）

场景异步加载进度条

五、技术重点

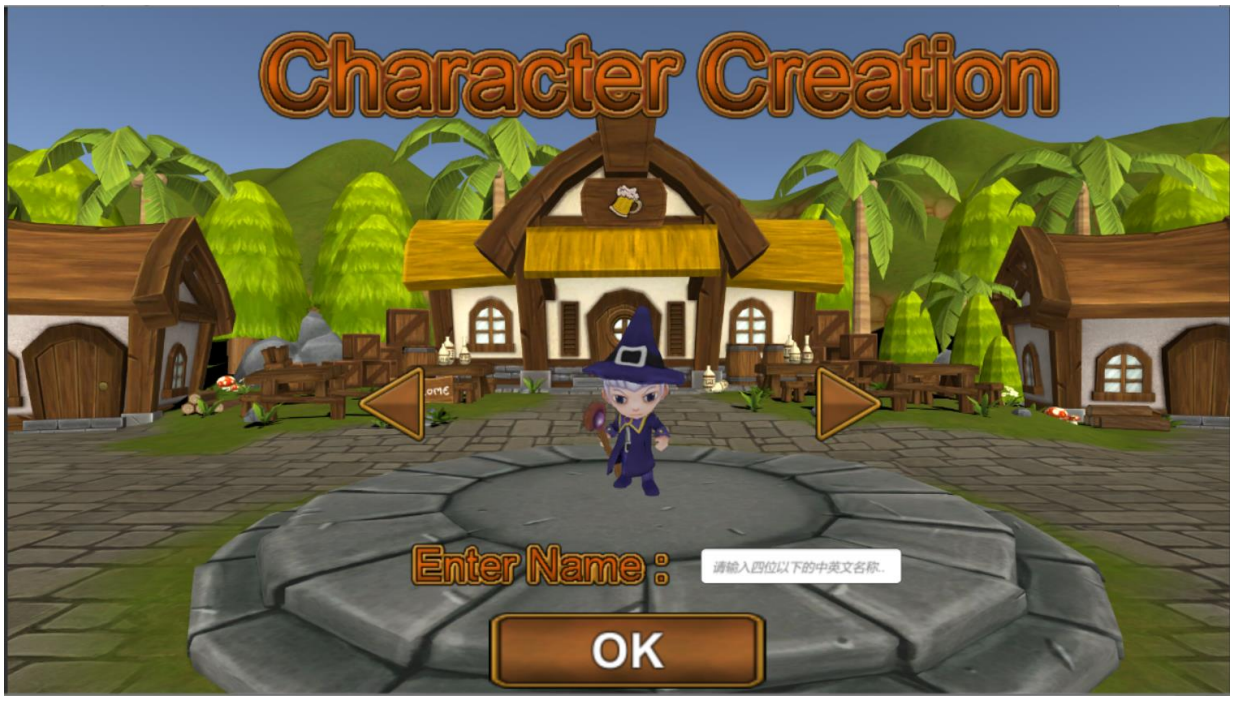
主界面:



主界面的镜头由远及近，最后停留在石墩的界面上。和一个按钮组成，按钮显示按任意键开始。以下是界面代码：

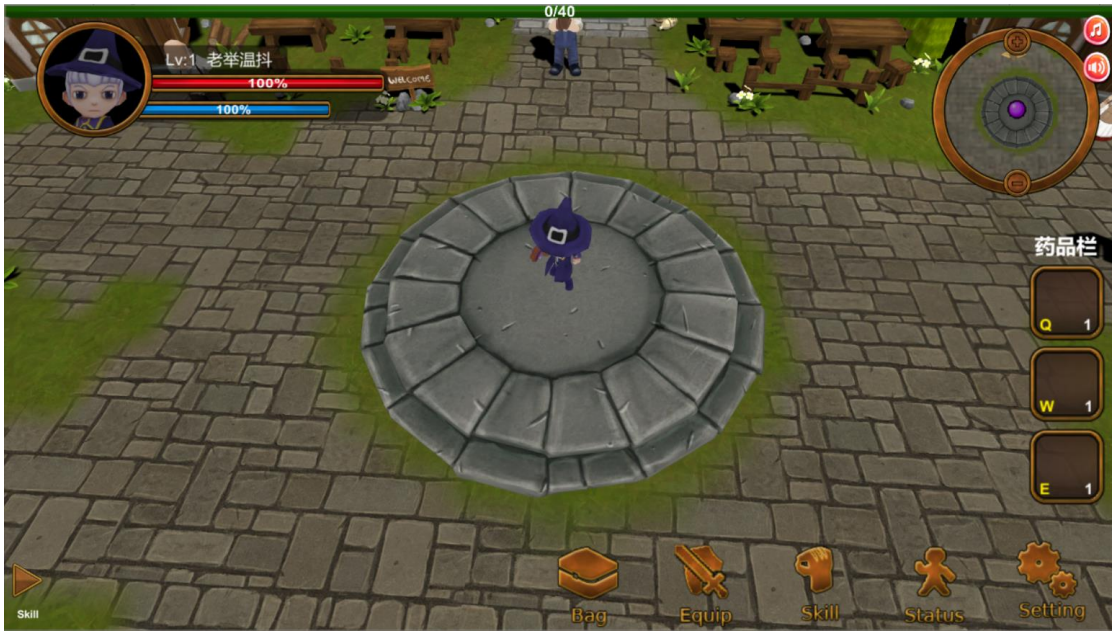
```
Void Update () {
        if(Input.anyKeyDown)
        {
            transform.GetComponent<Image>().enabled = false;
            Btn.SetActive(true);
        }
    }
```

点击 Name game 按钮进入游戏场景：



游戏主场景中在人物旁边有两个按钮，可以选择自己喜欢的主角，并且输入名字点击 ok，弹出进度条，加载完毕即可进入游戏。

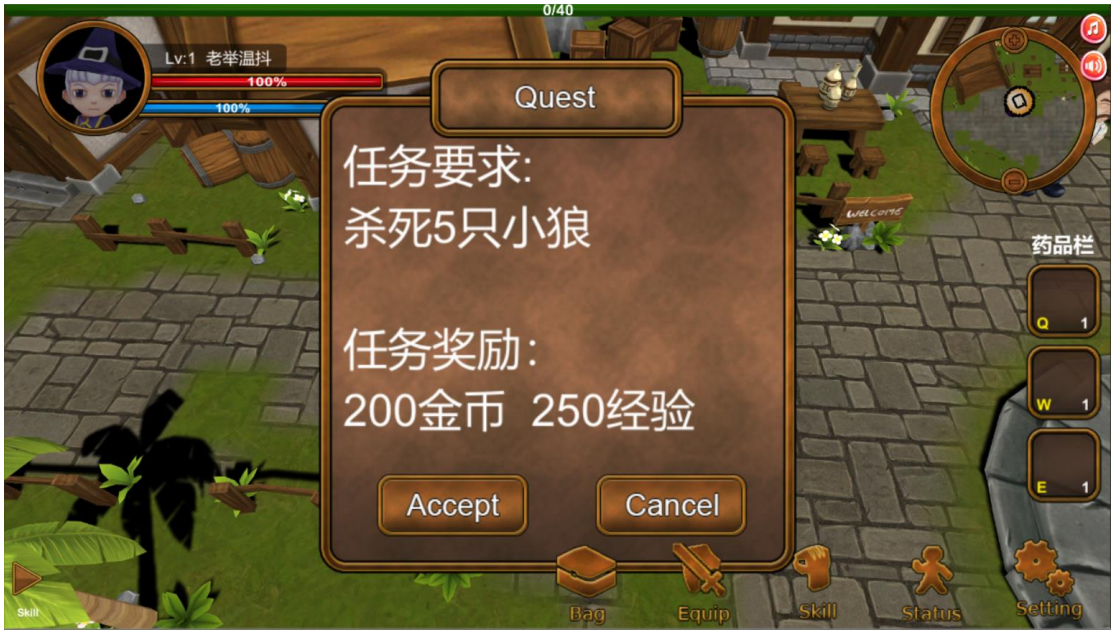
游戏界面：



进入游戏后，界面上左上角是人物的小地图，可以更清楚的看清人物走动的过程，以及人物的血条显示。右上角是场景的小地图，跟随着人物的走

动可以看清楚人物周围的场景，并且上面有“+”“-”按钮，用来放大和缩小小地图。右下角是各个技能的小图标，点击后会弹出相对应的窗口。

任务界面：



点击任务 NPC，会弹出任务窗口，需要完成制定的任务，才可以获得金币与经验奖励，既可以购买装备。

代码如下：

```
public class QuestNpc : NPC
{
    public static QuestNpc Instances;
    private void Awake()
    {
        Instances = this;
        if(PlayerPrefs.GetInt("任务次数")<=0)
        {
            DoNum = 1;
        }
        else
        {
            DoNum = PlayerPrefs.GetInt("任务次数");
        }

    }
    public Animator CanvasAnim;
```

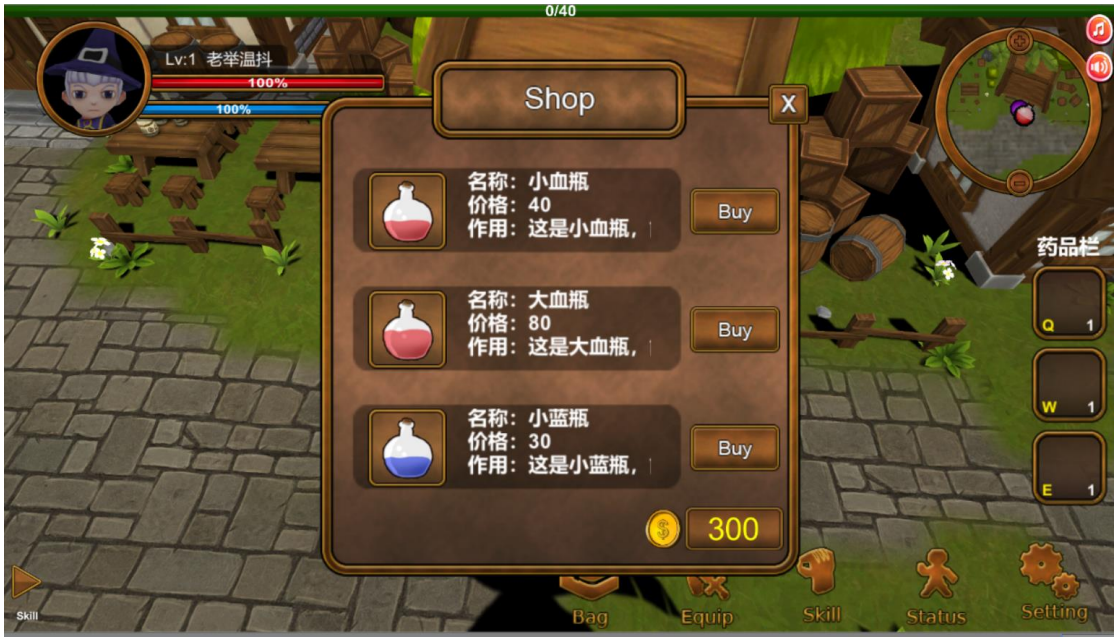
```
    public GameObject AcceptBtn;
    public GameObject FinishBtn;
    public Text TaskString;
    public Text NotFinish;
    public int KillAmount;
    public bool IsAcceptTask;
    public int DoNum=1;
    public EnemyType enemyType;
    //private void OnMouseOver()
    //{
    //
Cursor.SetCursor(Resources.Load<Texture2D>(TagsManager.TalkCursor),
Vector2.zero, CursorMode.Auto);
    //}
    public void AcceptBool()
    {
            if (IsAcceptTask)
            {
                IsAccept();
            }
            else
            {
                NotAccept();
            }
    }

    public void NotAccept()
    {
        if(DoNum<=8)
        {
            enemyType = EnemyType.Small;
            TaskString.text = "任务要求:\n 杀死" + ((DoNum - 1) * 5 + 5) + "只小
狼\n\n 任务奖励：\n" + (200 * DoNum) + "金币  " + (200 + 50 * DoNum) + "经验";
        }
        if (DoNum > 8)
        {
            enemyType = EnemyType.Middle;
            TaskString.text = "任务要求:\n 杀死" + ((DoNum - 1) * 5 + 5) + "只大
狼\n\n 任务奖励：\n" + (200 * DoNum) + "金币  " + (200 + 50 * DoNum) + "经验";
        }

        FinishBtn.SetActive(false);
        AcceptBtn.SetActive(true);
    }
```

```
    public void IsAccept()
    {
        if (DoNum <= 5)
        {
            TaskString.text = "任务进度:\n 已杀死" + KillAmount + "/" + ((DoNum -
1) * 5 + 5) + "只小狼\n\n 任务奖励：\n" + (200 * DoNum) + "金币  " + (200 + 50 *
DoNum) + "经验";
        }
        if (DoNum >= 10)
        {
            TaskString.text = "任务进度:\n 已杀死" + KillAmount + "/" + ((DoNum -
1) * 5 + 5) + "只大狼\n\n 任务奖励：\n" + (200 * DoNum) + "金币  " + (200 + 50 *
DoNum) + "经验";
        }
        AcceptBtn.SetActive(false);
        FinishBtn.SetActive(true);
    }

    public void OkBtn()
    {
        if(KillAmount>= ((DoNum - 1) * 5 + 5))
        {
            DoNum++;
            PlayerPrefs.SetInt("任务次数", DoNum);
            KillAmount = 0;
            IsAcceptTask = false;
            StartCoroutine(TextApear("获得奖励"+(200*DoNum)+"金币"+ (200 + 50 *
DoNum) + "经验"));
            PlayerStatusManager.Instances.AddCoins(200*DoNum);
            PlayerStatusManager.Instances.AddExp(200+ 50*DoNum);
            transform.GetComponent<AudioSource>().Play();
        }
        else
        {
            StartCoroutine(TextApear("任务未完成"));
        }
    }

    IEnumerator TextApear(string str)
    {
        NotFinish.text = str;
        NotFinish.gameObject.SetActive(true);
        yield return new WaitForSeconds(2f);
        NotFinish.gameObject.SetActive(false);
```

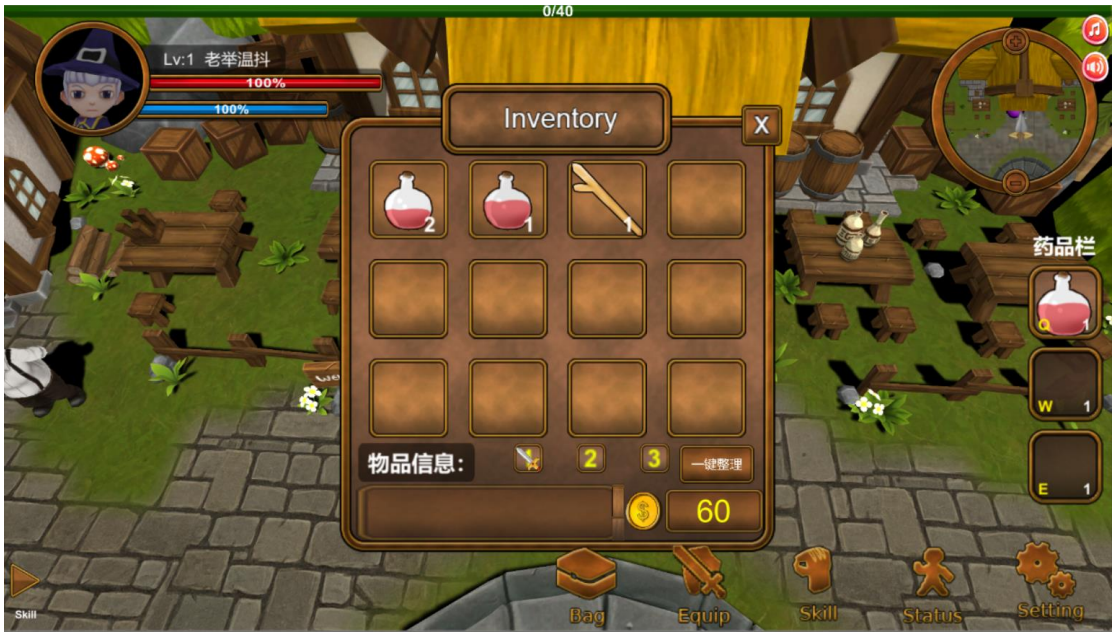
```
            Hidden();
        }

        //private void OnMouseExit()
        //{
        //    Cursor.SetCursor(Resources.Load<Texture2D>(TagsManager.NormalCursor),
Vector2.zero, CursorMode.Auto);
        //}
        public void Accept()
        {
            IsAcceptTask = true;
            AcceptBtn.SetActive(false);
            Hidden();
            FinishBtn.SetActive(true);
        }
}
```

游戏界面右下角五个技能窗口以及三位 NPC 介绍如下。

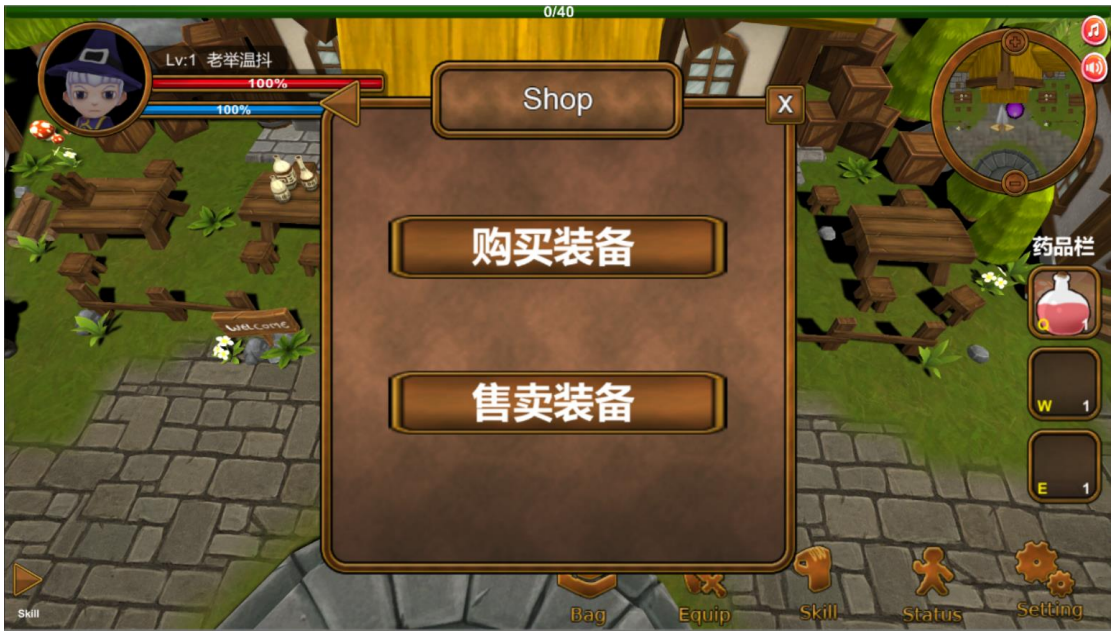
购买界面：



点击商店门前的 NPC，会弹出购买窗口，可根据自己的金币数量及需要，进行购买。成功购买后在我的背包界面就会显示已购买的物品。如下图所示：

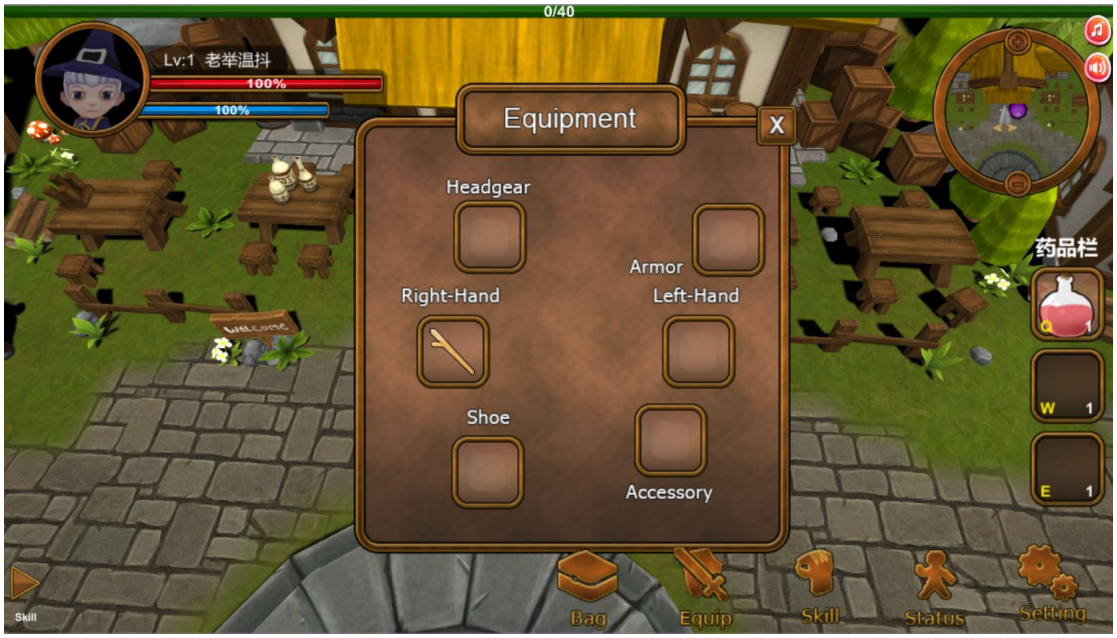


装备商店界面：

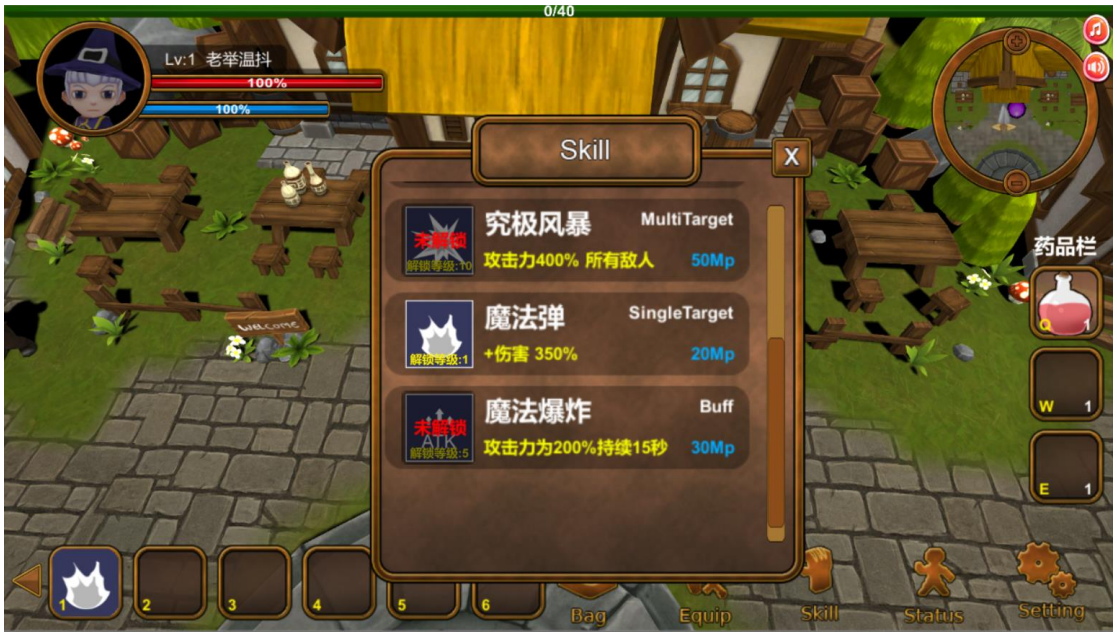


点击装备商店前的 NPC，可以购买或者售卖装备。

装备界面：

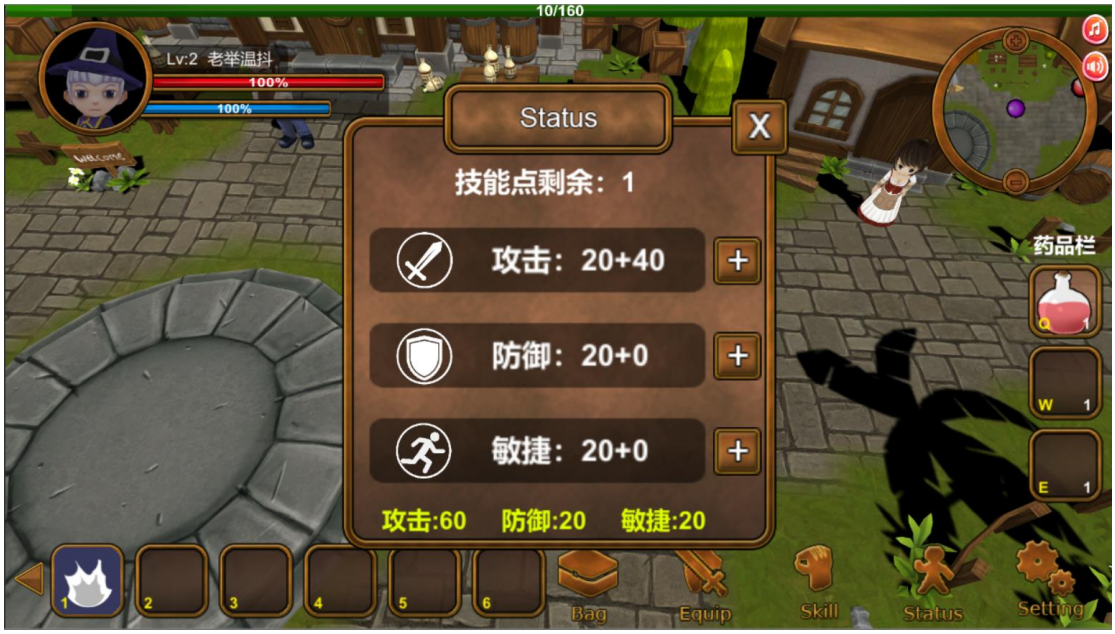


技能界面：

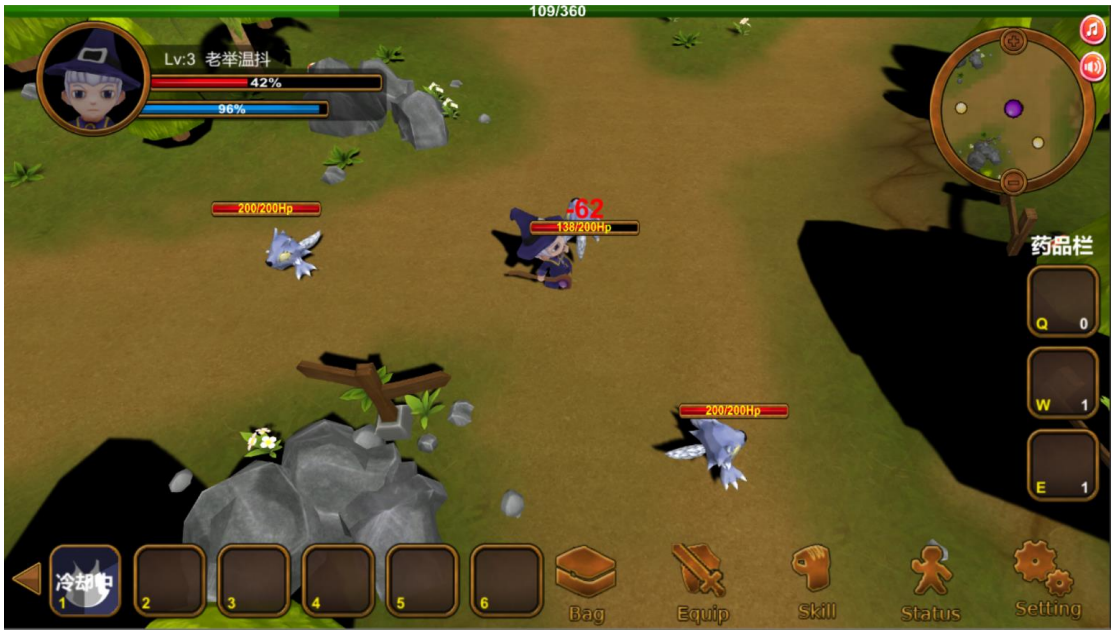


在左下角有一个 skill，点击之后如图所示，会弹出技能的快捷方式，可以将技能背包中已解锁的技能拖到快捷窗口里。

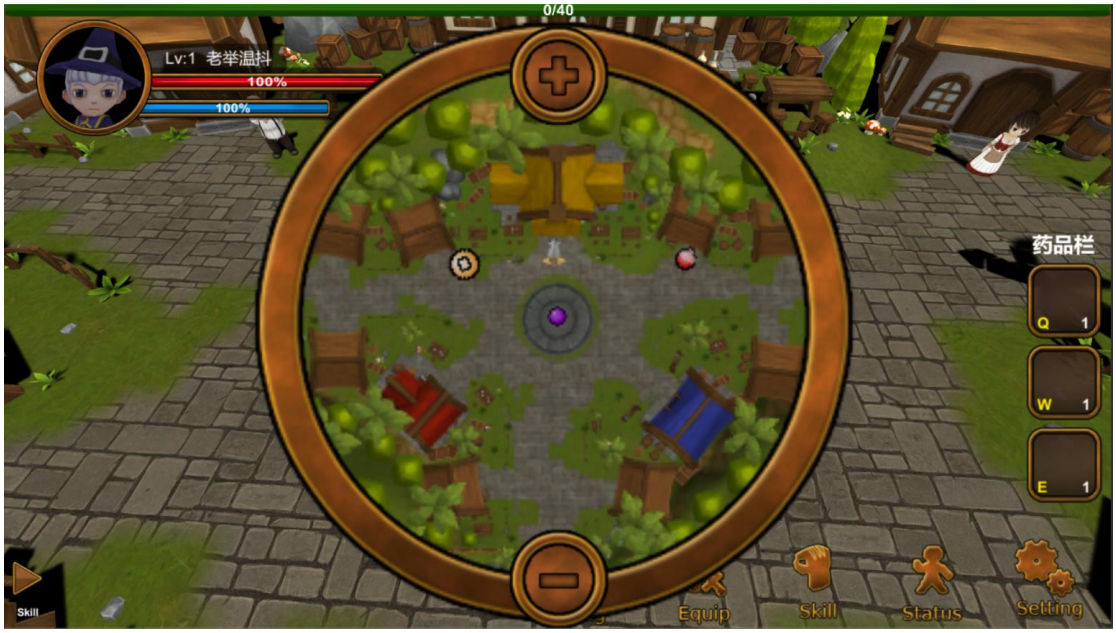
技能点窗口：


10/160
Lv:2 老举温抖
100%
100%
Status
X
技能点剩余：1
攻击：20+40
防御：20+0
敏捷：20+0
攻击:60 防御:20 敏捷:20
药品栏
Q 1
W 1
E 1
1
2
3
4
5
6
Bag
Equip
Skill
Status
Setting

攻击小狼界面：

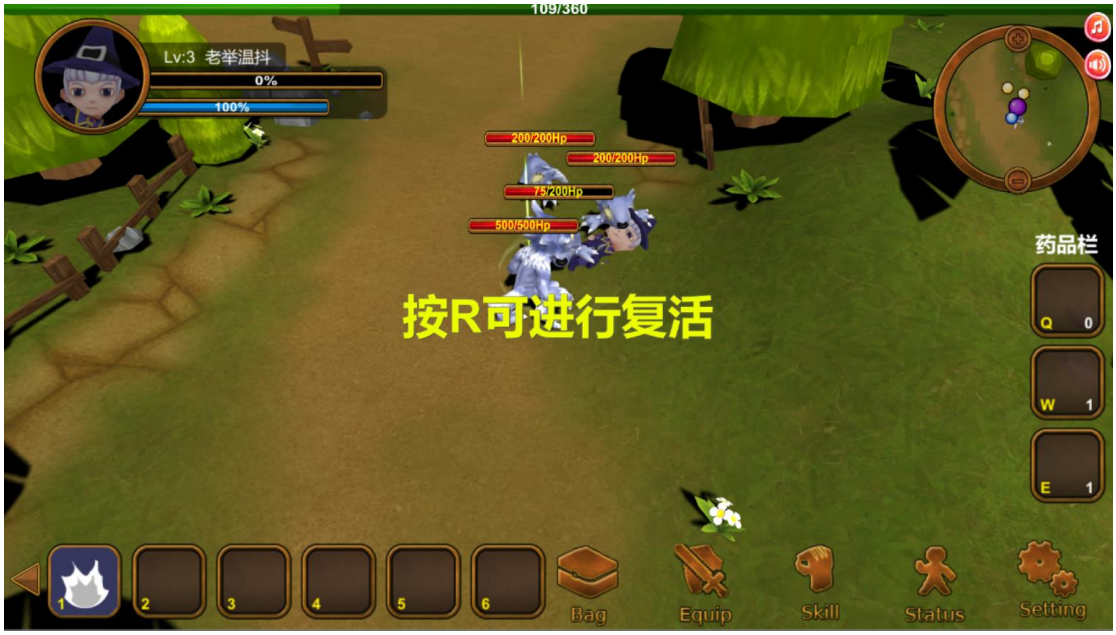

109/360
Lv:3 老举温抖
42%
96%
200/200Hp
-62
138/200Hp
200/200Hp
药品栏
Q 0
W 1
E 1
冷却中
1
2
3
4
5
6
Bag
Equip
Skill
Status
Setting

小地图放大界面：

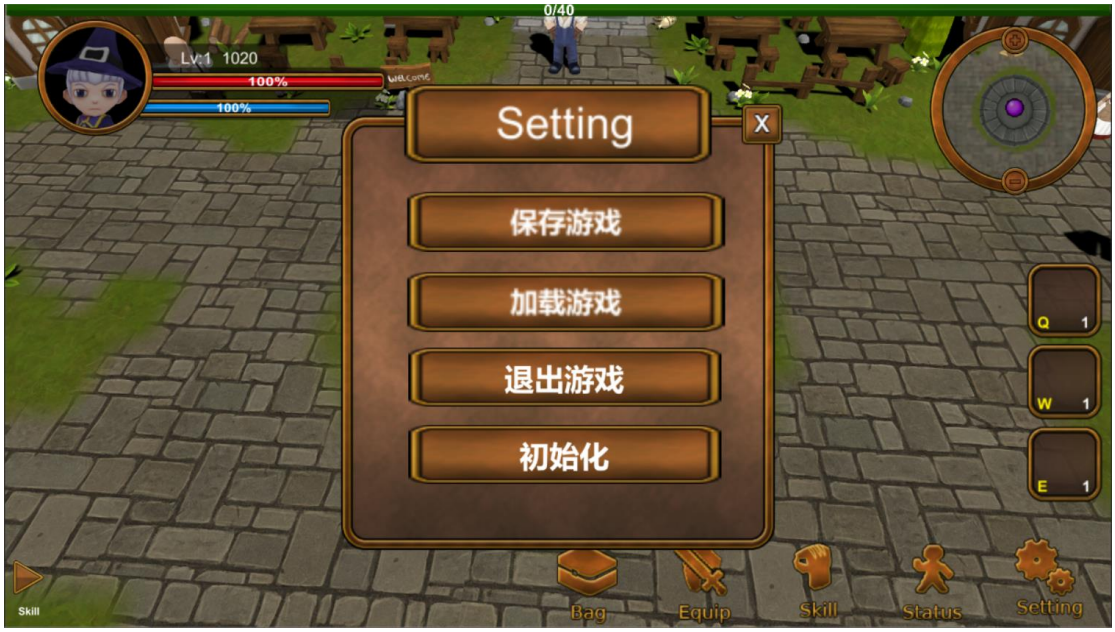


设置了小狼随机巡逻，在攻击小狼的过程中，小狼的血条会减少，如果成功击毙小狼，可获取相应的经验值。如若被小狼击中，主角血条减少，直至死亡。若杯小狼击毙，可以按 R 键，返回角色出生地。

并且在主角与小狼打斗中添加了特效，主角死亡按 R 键复活也使用了特效。



退出游戏界面：



六、场景制作方面：

1.镜头移动

代码如下：

```
public class CameraFollow : MonoBehaviour {
    public Player player;

    public Vector3 offset;
    public float distance;

    public float FarAwayTime;
    public bool IsFar, IsNear;
    public float ScrollSpeed;
    public float RotateSpeed;
    void Start()
    {
        player = GameManager.player;
        offset = transform.position - player.transform.position;
    }
    void Update()
    {
        TeacherCamFollow();
        CamFarOrAway();
        if(Input.GetMouseButton(1))
```

```
    {
        CamRotate();
    }

}
void TeacherCamFollow()
{
    transform.position = player.transform.position + offset;
}
void ScrollView()
{
    distance = offset.magnitude;
    distance -= Input.GetAxis(TagsManager.MouseScroll) * ScrollSpeed;
    distance = Mathf.Clamp(distance, 5f, 20f);
    offset = distance * (offset).normalized;
}

void CamFarOrAway()
{
    Camera Cam = transform.GetComponent<Camera>();
    if (Input.GetAxis("Mouse ScrollWheel") > 0)
    {
        FarAwayTime = 0;
        IsFar = true;
        IsNear = false;
    }
    else if (Input.GetAxis("Mouse ScrollWheel") < 0)
    {
        FarAwayTime = 0;
        IsNear = true;
        IsFar = false;
    }
    if (IsNear)
    {
        FarAwayTime += Time.deltaTime;
        Cam.fieldOfView += Time.deltaTime * ScrollSpeed;
        if (Cam.fieldOfView >= 90)
        {
            Cam.fieldOfView = 90;
        }
        if (FarAwayTime >= 0.3f)
        {
            IsNear = false;
        }
```

```
        }
        if (IsFar)
        {
            FarAwayTime += Time.deltaTime;
            Cam.fieldOfView -= Time.deltaTime * ScrollSpeed;
            if (Cam.fieldOfView <= 40)
            {
                Cam.fieldOfView = 40;
            }
            if (FarAwayTime >= 0.3f)
            {
                IsFar = false;
            }
        }
    }

    void CamRotate()
    {
        transform.RotateAround(player.transform.position, player.transform.up,
Time.deltaTime * Input.GetAxis(TagsManager.MouseX) * RotateSpeed);

        Vector3 pos = transform.position;
        Quaternion rot = transform.rotation;

        transform.RotateAround(player.transform.position, transform.right,
Time.deltaTime * Input.GetAxis(TagsManager.MouseY) * RotateSpeed);

        float x = transform.eulerAngles.x;
        if(x<20||x>80)
        {
            transform.position = pos;
            transform.rotation = rot;
        }
        offset = transform.position - player.transform.position;
    }
}
```

2.游戏逻辑

游戏的设计是需要逻辑依据的，在设置游戏玩法时，只有将逻辑理清楚，才能把一个游戏做完整。比如在黑暗之光的游戏中：从一开始进入的场景界面，点击开始游戏确定人物以及

设计名字，到正式进入游戏，玩家通过 NPC 的指引，击毙小狼，获取金币及经验，才可购买物品以及技能，是一个非常复杂的过程。所以游戏的逻辑显得非常的重要，在设置人物以及 NPC 的提示指引时，我们可以将具体思路写下来，以便更好的执行。

难点：

背包系统的复杂逻辑，各个系统之间的逻辑联系（例如背包与装备系统），敌人的 AI 的自然性。

七、参考资料：

1. Unity 4 3D 开发实战详解》人民邮电出版社

2.《Unity 3D 游戏开发》人民邮电出版社

3.《Unity 3D 游戏开发技术详解与典型案例》人民邮电出版社

4.《Unity 游戏开发实战》机械工业出版社

5.《Beginning 3D Game Development with Unity 4》 Sue Blackman