第十四次实验： 数据持久化

学号：22920212204392 姓名：黄勖

一、 实验目的

- 掌握 Unity3D 游戏数据保存和读取的相关功能

二、 实验条件

- 系统环境：Windows 10 21H2
- 软件环境：Unity 3D 2021.3.14f1c1

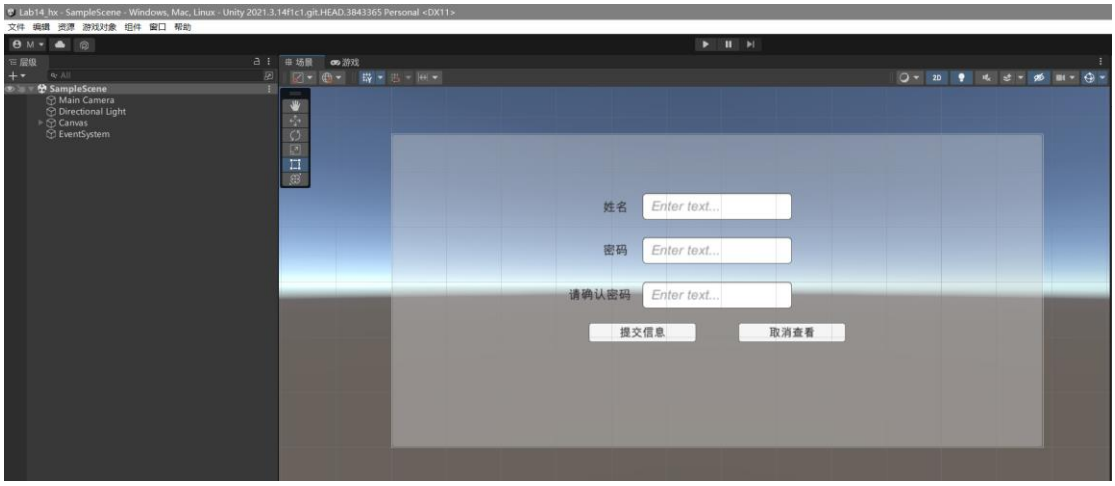
三、 实验内容

- 注册界面
  - 注册信息输入完成后，可点击“提交数据”将数据持久化保存，保存信息将显示在屏幕下方，点击“取消查看”即可关闭显示信息，并清空所有持久化存储信息
- 读取笑话&
  - 在本地保存 5 个包含笑话的文本文件，需为 UTF-16 格式，否则会出现乱码
  - 不同笑话之间用&做分隔符号，在程序中使用流将这些文本笑话读取并且分割显示在屏幕中

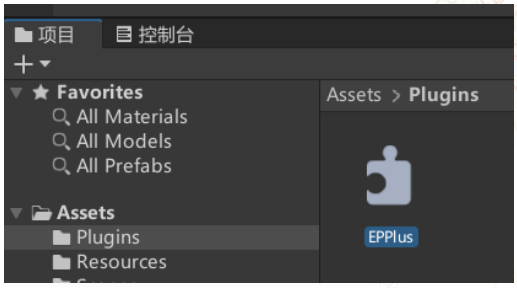
四、 实验项目步骤：

1．注册界面

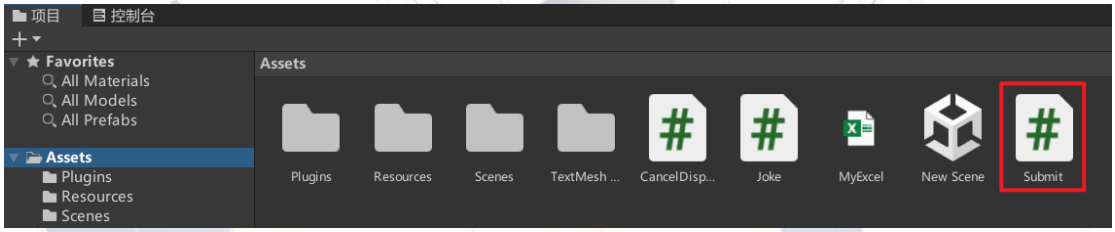
（1） 利用 Text，InputField 和 button 布置场景



（2） 下载 EPPlus 库并 Project 视图下新建的 Plugins 文件夹



（3） 新建一个“Submit”脚本，并进行编写，实现点击“提交数据”后信息显示在屏幕下方



· 声明需要用到的变量

Submit.cs

Assets > Submit.cs > Submit > CreateExcel()

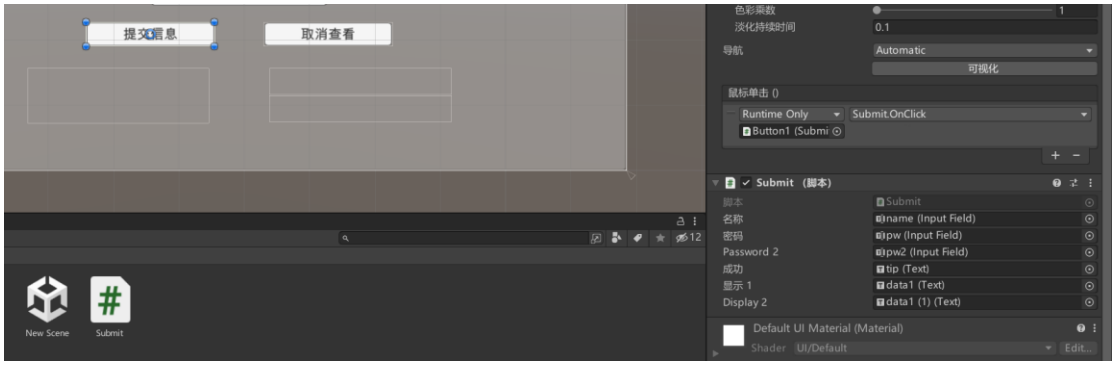
```
using System.Collections;
using System.Collections.Generic;
using System.Xml.Linq;
using UnityEngine;
using UnityEngine.UI;
using OfficeOpenXml;
using System.IO;

0 references
public class Submit : MonoBehaviour
{
    3 references
    public InputField Name;
    4 references
    public InputField password;
    2 references
    public InputField password2;
    3 references
    public Text success;
    3 references
    public Text display1;
    3 references
    public Text display2;
```

· 定义一个 OnClick 函数，用 if 语句进行判断

若三个 InputField 均为空，则提示输入姓名和密码，显示的两个 text 为空；若两次密码不同，则提示确认密码，两个显示 text 为空；否则提示注册成功，并在两个显示 text 中分别显示姓名及密码的注册信息

```
public void OnClick()
{
    if (Name.text == "" || password.text == "" || password2.text == "")
    {
        success.text = "请输入姓名及密码";
        display1.text = "";//显示昵称
        display2.text = "";//显示密码
    }
    else if (password.text != password2.text)
    {
        success.text = "请确认密码";
        display1.text = "";
        display2.text = "";
    }
    else
    {
        success.text = "注册成功！";
        display1.text = "姓名：" + Name.text;
        display2.text = "密码：" + password.text;
        i++;
        LoadExcel(newFile);
    }
}
```

· 将脚本挂载到 Button1 并对 public 变量进行赋值、更新 OnClick 列表



· 实现结果

（4） 实现点击“提交信息”后将数据持久化保存，保存信息将显示在屏幕下方

· 在 Submit 脚本中，自定义 CreateExcel 函数

定义 excel 文件的路径和名称后，实例化 EileInfo 类型并传入 outPutDir 参数来创建一个文件对象 newFile；

使用 using 语句创建一个 ExcelPackage 对象，表示一个 Excel 文档并提供各种操作方法；

excel.Workbook.Worksheets.Add("我的 Excel") 创建了名为“我的 Excel”工作表并返回其对象；

将 Excel 文档中的第一行第 1、2、3 列单元格分别设置了名称；

最后调用 excel.Save() 方法将结果保存到新的 Excel 文件中。

```
48      private FileInfo newFile;
        1 reference
49      void CreateExcel()
50      {
51          string outPutDir = Application.dataPath + "\\" + "MyExcel.xlsx";
52          newFile = new FileInfo(outPutDir);
53          if (newFile.Exists)
54          {
55              //创建一个新的Excel文件
56              newFile.Delete();
57              newFile = new FileInfo(outPutDir);
58          }
59          using (ExcelPackage excel = new ExcelPackage(newFile))
60          {
61              ExcelWorksheet worksheet = excel.Workbook.Worksheets.Add("我的Excel");
62              worksheet.Cells[1, 1].Value = "序号";
63              worksheet.Cells[1, 2].Value = "昵称";
64              worksheet.Cells[1, 3].Value = "密码";
65              excel.Save();
66          }
67      }
```

· 在 Submit 脚本中，自定义 LoadExcel 函数

传入 FileInfo 类型对象 newfile，指定要加载的 Excel 文件；

通过实例化 ExcelPackage 并将 newfile 传入来打开 Excel 文件；

访问 Excel 工作表的第一个工作表，向其中添加了一些新注册输入的数据；

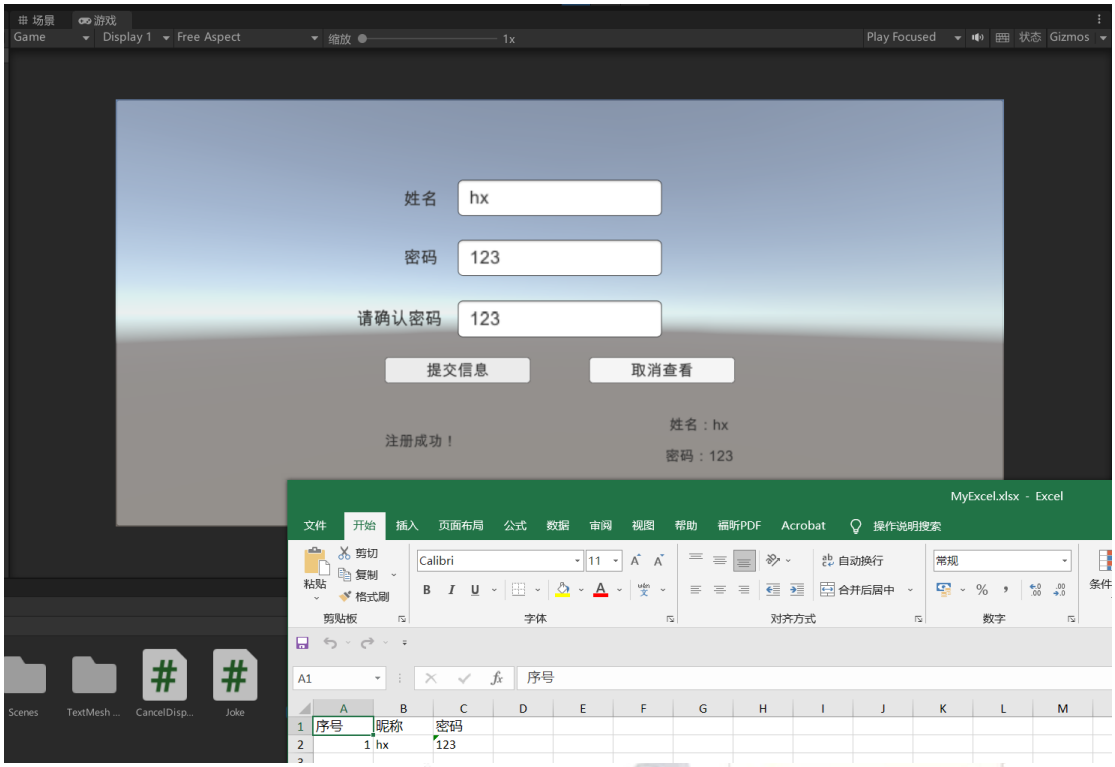
调用 excel.Save() 方法将更改后的 Excel 文件保存回磁盘

```
        1 reference
68      void LoadExcel(FileInfo newfile)
69      {
70          using (ExcelPackage excel = new ExcelPackage(newfile))
71          {
72              ExcelWorksheet worksheet = excel.Workbook.Worksheets[1];
73              worksheet.Cells[i, 1].Value = i - 1;
74              worksheet.Cells[i, 2].Value = Name.text;
75              worksheet.Cells[i, 3].Value = password.text;
76              excel.Save();
77          }
78      }
```

· 在 Start 函数中使用 CreateExcel 函数

```
18      private void Start()
19      {
20          CreateExcel();
21      }
```

· 实现结果



（5） 实现点击“取消查看”即可关闭显示信息，并清空所有持久化存储信息

· 在 Project 视图中，新建一个名为“CancelDisplay”脚本

声明几个需要用到的变量；

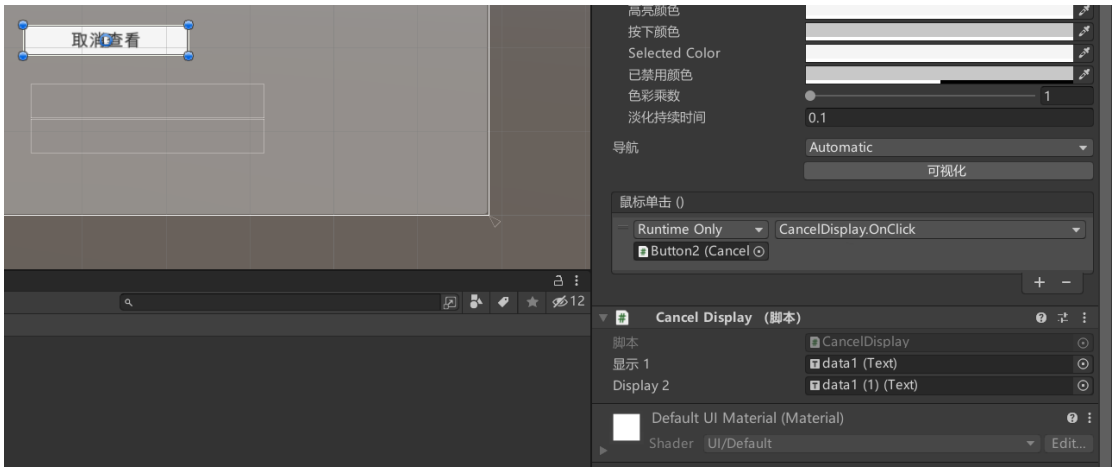
自定义 OnClick 函数，将两个显示 text 清空，调用在 Submit 脚本内编写好的 CreatExcel 函数，并输出清除成功的提示。

Submit.cs CancelDisplay.cs

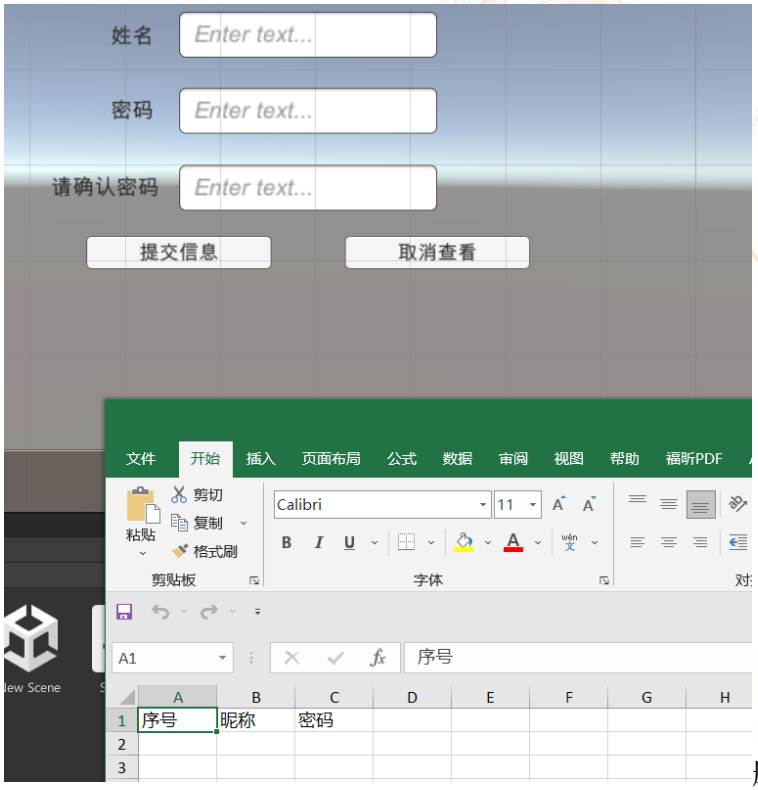
Assets > CancelDisplay.cs > CancelDisplay > OnClick()

```
using OfficeOpenXml;
using System.Collections;
using System.Collections.Generic;
using UnityEngine.UI;
using System.IO;
using UnityEngine;

0 references
public class CancelDisplay : MonoBehaviour
{
    1 reference
    public Text display1;
    1 reference
    public Text display2;
    7 references
    private FileInfo newFile;

    0 references
    public void OnClick()
    {
        display1.text = "";
        display2.text = "";
        //ClearExcel();
        CreateExcel();
        Debug.Log("清除成功");
    }

    //直接新建一个覆盖
    1 reference
    void CreateExcel()
    {
        string outPutDir = Application.dataPath + "\\" + "MyExcel.xlsx";
        newFile = new FileInfo(outPutDir);
        if (newFile.Exists)
        {
            //创建一个新的Excel文件
            newFile.Delete();
            newFile = new FileInfo(outPutDir);
        }
        using (ExcelPackage excel = new ExcelPackage(newFile))
        {
            ExcelWorksheet worksheet = excel.Workbook.Worksheets.Add("我的Excel");
            worksheet.Cells[1, 1].Value = "序号";
            worksheet.Cells[1, 2].Value = "昵称";
            worksheet.Cells[1, 3].Value = "密码";
            excel.Save();
        }
    }
```

· 将脚本挂载到 Button2 并对 public 变量进行赋值、更新 OnClick 列表



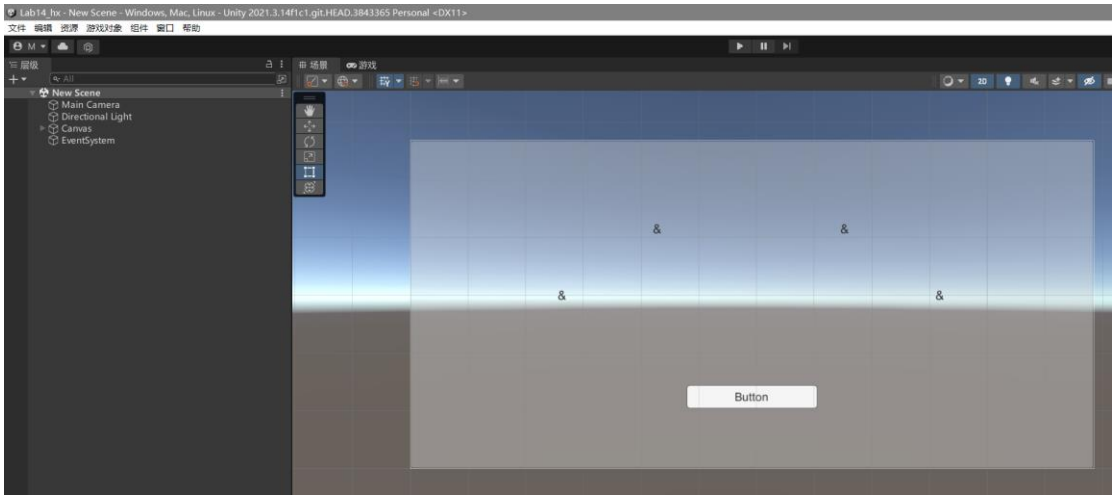
· 实现结果



删除完毕。

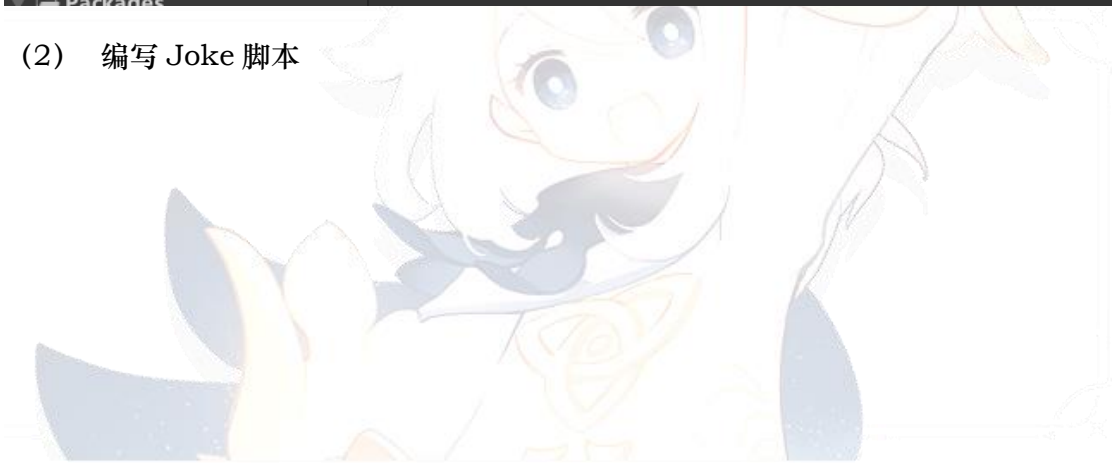
2. 读取笑话&——使用五个文本框进行分割，含有&的文本框进行连接

（1） 创建场景和保存笑话


New Scene
Main Camera
Directional Light
Canvas
EventSystem
Button


项目
控制台
Favorites
All Materials
All Models
All Prefabs
Assets
Plugins
Resources
Scenes
TextMesh Pro
Assets > Resources
笑话1
笑话2
笑话3
笑话4
笑话5

（2） 编写 Joke 脚本

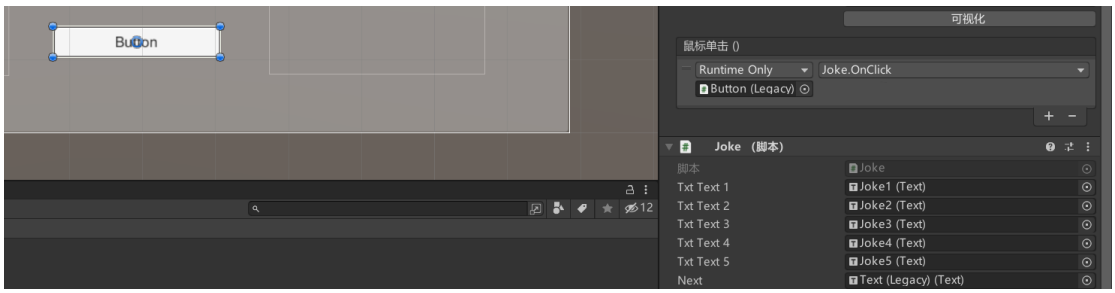


Joke.cs CancelDisplay.cs

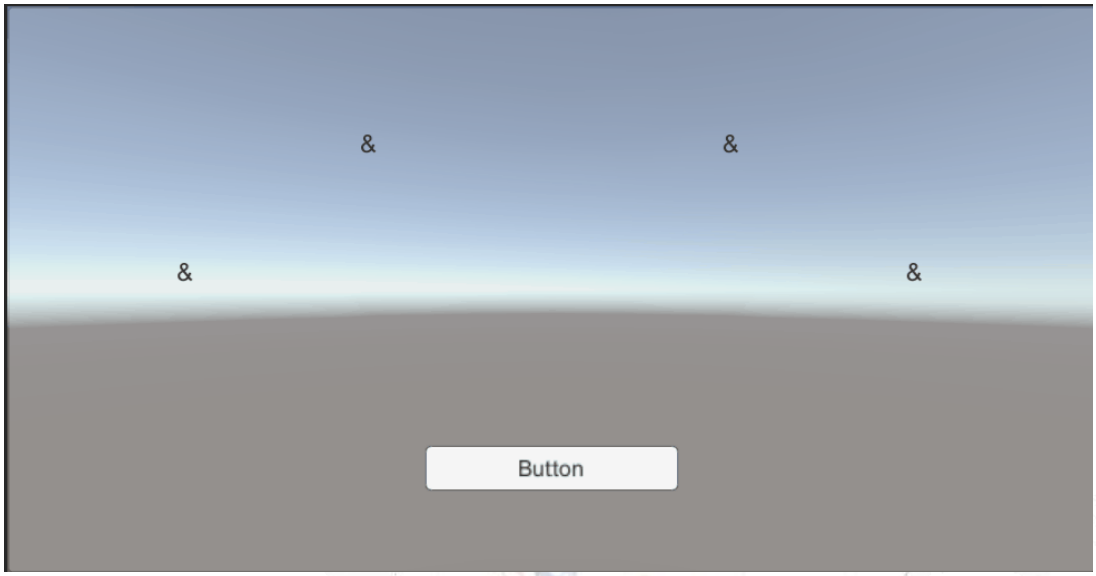
Assets > Joke.cs > Joke > OnClick()

```
7   public class Joke : MonoBehaviour
8   {
        6 references
9       private string joke;//存储当前笑话
        2 references
10      private string path;
        2 references
11      private StreamReader sr;
12
        7 references
13      int i = 1;//用于记录笑话的个数
14
        1 reference
15      public Text txtText1;//显示笑话
        1 reference
16      public Text txtText2;//显示笑话
        1 reference
17      public Text txtText3;//显示笑话
        1 reference
18      public Text txtText4;//显示笑话
        1 reference
19      public Text txtText5;//显示笑话
        1 reference
20      public Text next;//按钮文字
21
        0 references
22      public void OnClick()
23      {
24          path = Application.dataPath + "\\resources\\" + "笑话" + i + ".txt";
25
26          //读取内容
27          sr = new StreamReader(path);
28          joke = sr.ReadToEnd();
29
30          if (i == 1) txtText1.text = joke;
31          else if (i == 2) txtText2.text = joke;
32          else if (i == 3) txtText3.text = joke;
33          else if (i == 4) txtText4.text = joke;
34          else if (i == 5)//最后一个笑话后改变按钮文字
35          {
36              next.text = "没了哦";
37              txtText5.text = joke;
38          }
39          i++;
40      }
```

（3） 修改脚本的赋值



（4） 实现结果



最终效果详见视频演示

五、 实验心得总结：

通过本次实验我掌握了 Unity3D 游戏数据保存和读取的相关功能，基本了解了 Unity3D 实现数据持久化的过程，对 Unity 有了进一步的认识。