# 加载配置文件

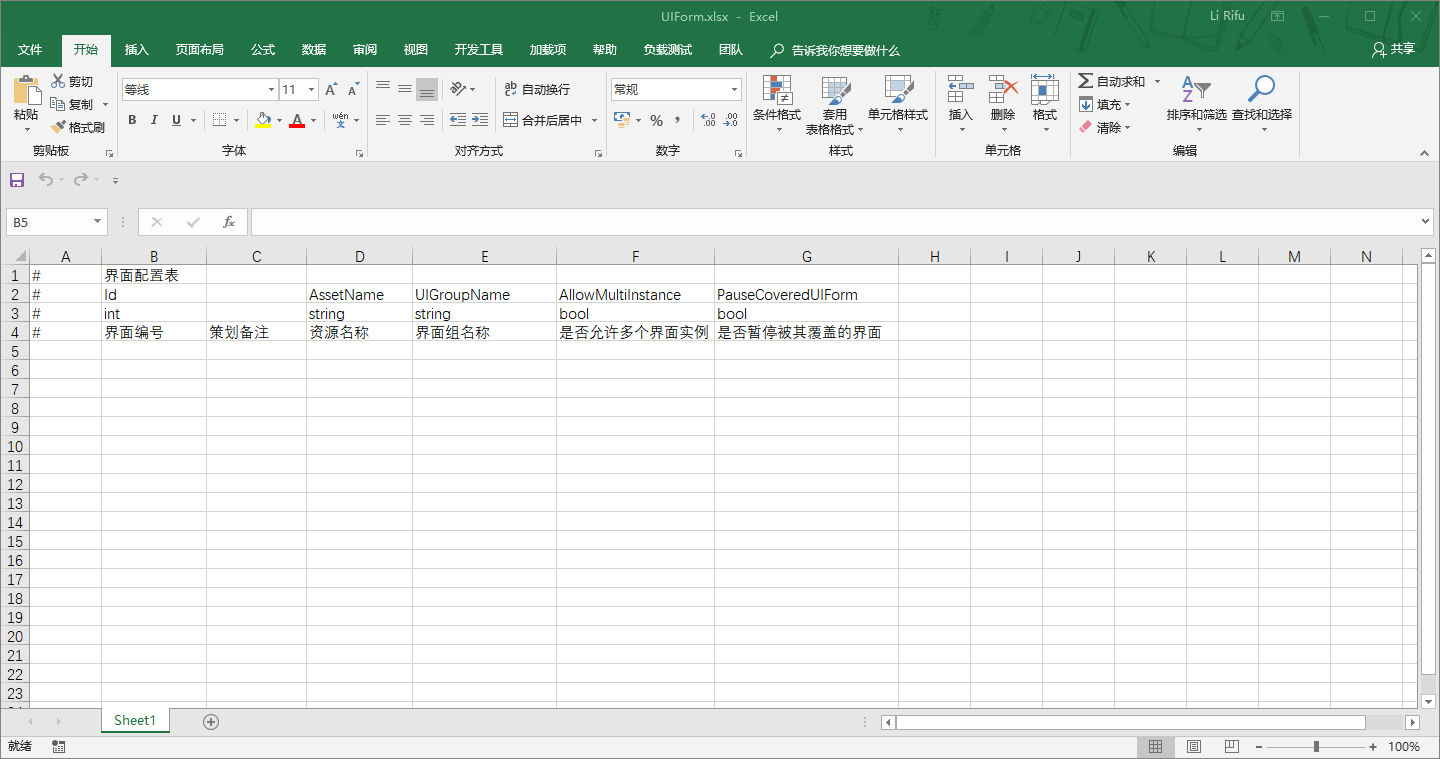
## 配置文件格式

官方默认的配置文件格式是”制表符分隔”文本，即用tab作为分隔符的文件。

我们项目中的配置文件我到放到了配表文件夹下：



打开文件夹内的UIForm文件



前四行带有#符号的行为注释行，正式数据前不带#符号。第一行表格名称，第二行为数据名称，第三行为数据类型，第四行为文字备注。

## 导出配置文件

编写好配置文件，直接选择另存为将文件保存到Assets\DataTables文件夹下，文件类型为.txt格式，编码格式为utf-8。

## 编写加载配置文件的类

在Assets\Scripts\DataTable路径下新建一个脚本，命名规则为DR+配表文件名。以UIForm文件为例，内容如下：

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using GameFramework.DataTable;

using System;

namespace RoyalWar

{

/// <summary>

/// 界面配置表

/// </summary>

public class DRUIForm : IDataRow

{

/// <summary>

/// 界面编号

/// </summary>

public int Id

{

get;

protected set;

}

/// <summary>

/// 资源名称

/// </summary>

public string AssetName

{

get;

protected set;

}

/// <summary>

/// 界面组名称

/// </summary>

public string UIGroupName

{

get;

protected set;

}

/// <summary>

/// 是否允许多个界面实例

/// </summary>

public bool AllowMultiInstance

{

get;

protected set;

}

/// <summary>

/// 是否暂停被其覆盖的界面

/// </summary>

public bool PauseCoveredUIForm

{

get;

protected set;

}

public void ParseDataRow(string dataRowText)

{

string[] text = DataTableExtension.SplitDataRow(dataRowText);

int index = 0;

index++;

Id = int.Parse(text[index++]);

index++;

AssetName = text[index++];

UIGroupName = text[index++];

AllowMultiInstance = bool.Parse(text[index++]);

PauseCoveredUIForm = bool.Parse(text[index++]);

}

private void AvoidJIT()

{

new Dictionary<int, DRUIForm>();

}

}

}

配置文件中包含了Id、AssetName、UIGroupName、AllowMultiInstance、PauseCoveredUIForm 5个字段，所有DRUIForm类中有对应的字段。

加载配置文件的类需要实现IDataRow接口，并实现ParseDataRow(string dataRowText)方法。