1. 介绍

项目主要用Unity开发,目前主要包含一下几个模块:

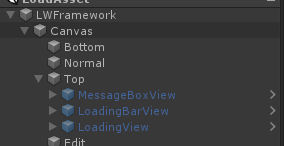
LWFramework

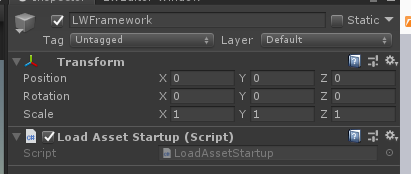
LWFrameworkAsset

LWFrameworkWebRequest

LWFrameworkStepGraph

另外框架依赖UniTask

1. 使用方法
   1. 在场景中放入内置的LWFramewok预制体,在Top层级下放入消息框\进度条\旋转进度等内置控件
   2. 编写启动脚本,挂载在LWFramework对象上



* 1. 编写Startup脚本



* 1. 状态机管理器-流程

使用状态机默认的Procedure控制整个项目的流程运转,即按需求功能自己划分成多个Procedure,不同模块中通过当前的Procedure进行函数调用

ManagerUtility.FSMMgr.GetFSMProcedure().GetState<流程状态>().函数()

* 1. UI管理器

UI模块包含的内容比较多,主要控制方式都是通过

ManagerUtility.UIMgr.OpenViewAsync<XXXView>(); // 异步打开

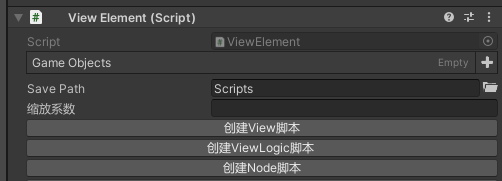
ManagerUtility.UIMgr.OpenView<XXXView>();//同步打开

ManagerUtility.UIMgr.Close<XXXView>();//关闭

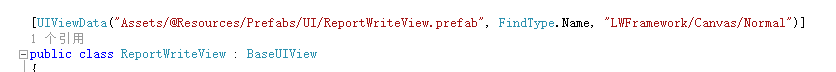
ManagerUtility.UIMgr.Clear<XXXView>();//清理

具体看查看IUIManager的接口注释

---------View

创建View可以使用挂载在View对象上自动创建, 此处建议,简单的View,只使用View.具有一定逻辑操作的创建ViewLogic,这样可以将逻辑与View拆分.Node脚本会适用于类似包裹格子等相同的小view使用,可以使用对象池的方式进行处理,详细查看Example

------------View特性

创建好的View类行会自带一个UIViewData特性,特性中包含了UI资源的路径(会用到资源管理器加载),FindType-寻找父节点的方式,字符串参数,用于FindType使用的参数)

UIElement-通过transform.Find查找的参数,自动生成

DataBinding()-将UI与数据进行绑定,通过Key将控件绑定到IUIManager.ViewData的数据中.

备注:Logic中内置了个OnViewDataChange同步可以监听ViewData中的数据状态,复杂逻辑建议使用OnViewDataChange

* 1. 资源管理器

ManagerUtility.AssetsMgr.Load<T>(path) //同步加载资源

ManagerUtility.AssetsMgr.LoadAsync<T>(path)//异步加载资源

ManagerUtility.AssetsMgr.LoadSceneAsync<T>(path)//异步加载场景,如果场景是一个分包,切没下载,会自动下载

备注:详细可以查看IAssetsManager 接口注释



资源管理器 内置了AB及Resources,在Startup中设置.上图可以开启AB功能.