## UI框架详述

以下内容按照类图从上到下的顺序叙述UI框架。

### UIWindow：

所有UI都继承自这个类，类中持有UI的预制体，定义了整个UI的生命周期，子类UI根据不同的需求去重写生命周期方法。

持有当前UI的状态、当前UI的弹出类型、当前UI的父级UI。

当弹出类型是Show时，如果当前有展示的UI，这个UI会直接覆盖在上一个UI上层；

当弹出类型是Pop时，如果当前有展示的UI，会先关闭当前的UI，再弹出这个UI；

当弹出类型是PopSub时，如果当前有展示的UI，会先关闭当前的UI，将其保存成当前UI的父级，然后弹出这个UI。并且当关闭这个UI时，重新弹出父级UI。

封装了点击事件方法，在UIShow、Hide阶段或手动屏蔽点击事件后，屏蔽所有点击事件，并且每个按钮有0.3秒的连点CD。

### UIWindowManager：

UI管理类。

首先统一了所有UI可能用到的底层黑遮罩，不论有多少个UI都只展示一个黑遮罩。

缓存UIWindow预制体，在GetUIWindow时，若List中有缓存，则取缓存，跳过OnCreate，直接进入OnPreShow，否则实例化新Prefab，并且执行UIWindow的OnCreate。通过手动Destory一个Window或者游戏切换状态时在UnloadUnUseAsset之前释放掉所有Window。

管理所有UIWindow的生命周期，当Get一个UIWindow时，会执行上述取缓存的流程，之后如果调用Show逻辑，会先执行OnPreShow中的逻辑，之后播放UI的Show动画，播放完成后，执行OnShowOver，至此UI已经展示出来了，之后如果调用Hide逻辑，会先执行OnPreHide，在Hide动画播放结束后，执行OnHideOver逻辑，如果想手动释放这个预制体，就Destory这个Window，至此UI生命周期结束。

管理了一个待弹出的UI栈，用于自动、连续的弹出UI，比如每天首次登陆游戏时，连弹几个活动UI，在首次进入MapState时或关闭一个UI时会从栈顶弹出一个UI。

### MessageCenter：

消息中心。

采用观察者模式，首先AddListener注册一个Key，这个Key可以对应多个不同的event，当对这个Key执行SendMessage后，触发所有对应的event。通过RemoveListener，可以移除该Key下注册的所有事件。

### LocalizationManager：

本地化工具。

UILocalization，继承MonoBehaviour，与TextMeshPro成对出现。

持有多语言Key，通过Key去匹配不同语言的文本；

持有预设的字体名称，预设的字体材质；

当SendMessag(OnLanguageChange)时，根据持有的信息刷新字体的Text、字体、材质。