

# ET 框架学习笔记 - - 自己需要这样一个总结文档来帮助 总结与急速重构自己的游戏

deepwaterooo

May 14, 2023

## Contents

<b>1 客户端场景组件：客户端大致的起始过程</b>	<b>1</b>
1.1 Entry.cs: 指定的起始类，会触发三类回调，公用组件类的加载，和其它	1
1.2 EntryEvent1_InitShare: 第一类，公用组件类的加载，公用的几大组件	1
1.2.1 CurrentScenesComponent: 可以用来管理多个客户端场景，比如大世界会加载多块场景 (是说，大地图可以分 10 块 8 块小地图吗?)	1
1.2.2 CurrentScenesComponentSystem: CurrentScene() 方法，返回当前场景	2
1.2.3 ObjectWait: 也有生成系	2
1.2.4 PlayerComponent:	2
1.2.5 PlayerComponentSystem: 生成系，到处都要用它	2
1.3 AfterCreateCurrentScene_AddComponent:【UIComponent】【ResourcesLoaderComponent】	2
1.3.1 UIComponent: 管理 Scene 上的 UI	2
1.3.2 UIComponentSystem: 管理 Scene 上的 UI: 这个是组件生成管理系统，负责添加与删除。【UIEventComponent】是 UI 上的 UI 事件组件系统	3
1.3.3 ResourcesLoaderComponent: 相关的资源加载，这个文件里有生成系	3
1.4 EntryEvent2_InitServer: 前面 1 里，两端公用组件准备好了，现在就起始服务器？服务端的几大组件：	3
1.4.1 ActorMessageSenderComponent: 发送普通 actor 消息	4
1.4.2 ActorLocationSenderComponent: 发送 location actor 消息	4
1.4.3 LocationProxyComponent: 访问 location server 的组件	4
1.4.4 ActorMessageDispatcherComponent: Actor 消息分发组件	4
1.4.5 ServerSceneManagerComponent: 可以去对比，两端的管理者组件，有什么不同？	4
1.5 EntryEvent3_InitClient: 客户端	4
1.5.1 ResourcesComponent: 热更新资源包等的处理	5
1.5.2 GlobalComponent: 不知道是干什么的，Unity 里好像是 Root 根节点下的一个节点，组件？	5
1.6 前面三件 (【公用组件】，【服务器】，【客户端】的应用程序启动完成) 触发 UI 变更: 这个 UI 订阅说，一被通知，就创建注册登录界面	5
<b>2 ClientComponent ClientScene 等客户端相关：有点儿理不清</b>	<b>5</b>
2.1 ClientSceneManagerComponent: 是否，相当于，它是 SceneType 的管理者，就是先前各种服，注册登录服，网关服、匹配服等的管理者，大概主要还是地址传送	5

<b>3 客户端场景与客户端场景加工厂</b>	<b>6</b>
3.1 SceneChangeHelper: 场景切换协程	6
3.1.1 Unit: Unit 究竟是什么, 干什么的? 像是游戏的一个最小单位, 有位置与旋转参数	6
3.1.2 UnityEngine: 组件	7
3.1.3 UnityEngineSystem: 生成系. 感觉这个系统不太懂	7
3.1.4 UnitHelper: 帮助在不同使用情境下, 拿到 unit	7
3.2 SceneFactory: ClientScene: 添加三组件: <b>【CurrentScenesComponent】【PlayerComponent】【ObjectWait】</b> 。	7
3.2.1 UnitFactory: 为什么我抓出两个不一样的定义, 还没弄明白	8
<b>4 标签系: 标签系统重构了, 现分为几个类型</b>	<b>8</b>
4.1 ComponentOfAttribute : Attribute	8
4.2 ComponentView: MonoBehaviour	9
4.3 ComponentViewEditor: Editor	9
<b>5 UI 上的事件驱动系统:</b>	<b>9</b>
5.1 EventType	9
5.2 由 AppStartInitFinish 事件所触发的 CreateLoginUI	9
5.3 由 LoginFinish 事件所触发的 CreateLobbyUI	9
5.4 SceneChangeStart_AddComponent: 开始切换场景的时候, 就自动添加 <b>【OperaComponent】</b> 组件。现在对场景这块儿还不够熟悉	10
<b>6 Helper 类的总结: 【但凡点击回调方法, 就变成 Helper 类!】为什么就变成了这么一个的帮助类呢?</b>	<b>10</b>
6.1 LoginHelper.cs	10
6.2 EnterRoomHelper.cs	10
6.3 UIHelper.cs: 负责 UI 界面上的组件的, 添加与删除, 异步完成	11
6.4 SceneChangeHelper: 场景切换协程	12
<b>7 UI 控件的生产事件机制流程: 以前的专用工厂再包装为 UI 上的事件机制</b>	<b>12</b>
7.1 LoginHelper 发布 EventType.LoginFinish() 事件	12
7.2 LoginFinish_RemoveLoginUI: 一般对应两个事件, 旧视图的去除, 与新视图的添加	13
7.3 LoginFinish_CreateLobbyUI: 创建新视图	13
7.4 UIHelper: 帮助类, 来添加或是移除 UI 上的可装可拆的组件	13
7.5 UnityEngineSystem: 管理 Scene 上的 UI: 这个是组件生成管理系统, 负责添加与删除。 <b>【UIEventComponent】</b> 是 UI 上的 UI 事件组件系统	13
7.6 UIEventComponentSystem: 管理所有 UI GameObject 以及 UI 事件: 应该主要是 UI 控件相关的事件。 <b>【自顶向下】</b> 的组件系统	13
7.7 AUIEvent: 跟下面的 UIEventComponent 关系是?	14
7.8 UIEventComponent: 管理所有 UI GameObject	14
7.9 UIEventComponentSystem: 生成系, 管理所有 UI GameObject 以及 UI 事件: 应该主要是 UI 控件相关的事件。 <b>【自顶向下】</b> 的组件系统	14
7.10 UILoginEvent: 一个实体类的例子, 具体的工厂生产逻辑	15
7.11 UILobbyEvent: 再加一个实体类的例子	15
<b>8 Session 相关: 进行间通信</b>	<b>16</b>
8.1 SessionComponent	16
8.2 SessionComponentDestroySystem: <b>【销毁系】</b> : 只负责用户掉线, 或是下线后的自动移除会话框	16
8.3 OperaComponentSystem: 一个拿会话框必消息的使用场景	16

<b>9 Player: 玩家相关，添加的地方，以及使用【这里还是有点儿糊涂】</b>	<b>17</b>
9.1 Player: 玩家	17
9.2 PlayerComponent:	17
9.3 服务器端的 PlayerComponent	17
9.4 服务器端 PlayerComponentSystem	17
9.5 服务器端 SceneFactory-CreateServerScene 时【网关服】会添加【PlayerComponent】 玩家组件	17
9.6 SessionPlayerComponentSystem	18
9.7 SessionPlayerComponent: 会话框里，会保留客户端玩家 playerId	18
<b>10 Match: 匹配服，没有独立出来的匹配服</b>	<b>18</b>
10.1 服务器端 SceneFactory 的场景类型: SceneType-s	18
<b>11 ResourcesComponent 资源包管理器相关：有时候，拖拉机游戏里会需要拿它来加载图片</b>	<b>19</b>
11.1 ResourcesComponent: 同文件有其生成系	19
11.2 客户端 ConfigLoader 的 Invoke 标签下：在根控件 Root 下添加资源管理器组件	19
<b>12 整个框架：ET 7.2 + YooAssets + luban + FairGUI</b>	<b>20</b>
<b>13 写在最后：反而是自己每天查看一再更新的</b>	<b>20</b>
<b>14 现在的修改内容，记忆</b>	<b>21</b>
<b>15 TODO 今天晚上把几个消息抓全了，免得一堆的报错</b>	<b>22</b>

## 1 客户端场景组件：客户端大致的起始过程

### 1.1 Entry.cs: 指定的起始类，会触发三类回调，公用组件类的加载，和其它