

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 12, 2023

Contents

1 UI 上的事件驱动系统:

1.1 EventType

```
namespace EventType {
    public struct SceneChangeStart {
    }
    public struct SceneChangeFinish {
    }

    public struct AfterCreateClientScene {
    }
    public struct AfterCreateCurrentScene {
    }

    public struct AppStartInitFinish {
    }
    public struct LoginFinish {
    }
    // public struct EnterMapFinish {
    public struct EnterRoomFinish {
    }
    public struct AfterUnitCreate {
        public Unit Unit;
    }
}
```

1.2 由 AppStartInitFinish 事件所触发的 CreateLoginUI

```
[Event(SceneType.Client)] // ET 事件系统的工具, 标签系
public class AppStartInitFinish_CreateLoginUI: AEvent<EventType.AppStartInitFinish> {
```

1.3 由 LoginFinish 事件所触发的 CreateLobbyUI

```
[Event(SceneType.Client)]
public class LoginFinish_CreateLobbyUI: AEvent<EventType.LoginFinish> {
```

- 这些是原示范框架都已经完成了的, 我只需要添加剩余的逻辑。

1.4 现游戏项目 ET7 里的主要适配: 由原 UI 上的组件系统, 手动组件添加移除, 更改为 UI 上的事件驱动系统的事件定义, 注册, 与工厂生成系?

- 就是说, 接下来游戏里的主要逻辑, 也将变成如此一个个不同的自定义事件, 来驱动 UI 上的各种回调。【爱表哥, 爱生活!!! 活宝妹就是一定要嫁给亲爱的表哥!!!】
- UI 界面上的按钮点击要如何处理呢, 新框架里, 找个例子看看, 点【进入地图】看看

2 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个的帮助类呢?

2.1 LoginHelper.cs

```
public static class LoginHelper {
    public static async ETTask Login(Scene clientScene, string account, string password) {
        try {
            // 创建一个 ETModel 层的 Session
            clientScene.RemoveComponent<RouterAddressComponent>();
            // 获取路由跟 realmDispatcher 地址
            RouterAddressComponent routerAddressComponent = clientScene.GetComponent<RouterAddressComponent>();
            if (routerAddressComponent == null) {
                routerAddressComponent = clientScene.AddComponent<RouterAddressComponent, string, int>(ConstValue.RouterHttpHost, ConstValue.RouterHttpPort);
                await routerAddressComponent.Init();
            }
            clientScene.AddComponent<NetClientComponent, AddressFamily>(routerAddressComponent.RouterManagerIPAddress.AddressFamily, routerAddressComponent.RouterManagerIPAddress.IPEndPoint, routerAddressComponent.RouterManagerIPAddress.IPEndPoint);
            IPEndPoint realmAddress = routerAddressComponent.GetRealmAddress(account);

            R2C_Login r2CLogin;
            using (Session session = await RouterHelper.CreateRouterSession(clientScene, realmAddress)) {
                r2CLogin = (R2C_Login) await session.Call(new C2R_Login() { Account = account, Password = password });
            }
            // 创建一个 gate Session, 并且保存到 SessionComponent 中: 与网关服的会话框。主要负责用户下线后会话框的自动移除销毁
            Session gateSession = await RouterHelper.CreateRouterSession(clientScene, NetworkHelper.ToIPEndPoint(r2CLogin.Address));
            clientScene.AddComponent<SessionComponent>().Session = gateSession;

            G2C_LoginGate g2CLoginGate = (G2C_LoginGate)await gateSession.Call(
                new C2G_LoginGate() { Key = r2CLogin.Key, GateId = r2CLogin.GateId });
            Log.Debug(" 登陆 gate 成功!");
            await EventSystem.Instance.PublishAsync(clientScene, new EventType.LoginFinish());
        }
        catch (Exception e) {
            Log.Error(e);
        }
    }
}
```

2.2 EnterRoomHelper.cs

- 这里需要注意的是: 原项目里面还是保留了 C2G_EnterMap 消息的。分两块查看一下:
 - 可以先去查一下, 斗地主里是如何【开始匹配】的
 - ET 7 框架里, 服务器是如何处理消息的, 变成了路由系统, 如何由特定路由服务器来处理消息?
 - * 仍然是 **标签系的消息处理器**: 因为先前的不同服变成了现在的不同场景, 分场景(先前的不同服)来定义消息处理器, 以处理当前场景(特定功能逻辑服)下的消息, 如匹配服的消息。
 - 如果每个按钮的回调: 都单独一个类, 不成了海量回调类了?
 - 老版本: 斗地主里, 进入地图的参考【ET】里, 就要去找, 如何处理这些组件的?

```
// public static class EnterMapHelper {
public static class EnterRoomHelper {
    // 进拖拉拉机房: 异步过程, 需要与房间服交互的。【房间服】:
    // 【C2G_EnterRoom】: 消息也改下
    public static async ETTask EnterRoomAsync(Scene clientScene) {
        try {
            G2C_EnterMap g2CEnterMap = await clientScene.GetComponent<SessionComponent>().Session.Call(new C2G_EnterMap()) as G2C_EnterMap;
            clientScene.GetComponent<PlayerComponent>().MyId = g2CEnterMap.MyId;

            // 等待场景切换完成
            await clientScene.GetComponent<ObjectWait>().Wait<Wait_SceneChangeFinish>();
        }
    }
}
```

```

// EventSystem.Instance.Publish(clientScene, new EventType.EnterMapFinish());
EventSystem.Instance.Publish(clientScene, new EventType.EnterRoomFinish()); // 这个，再去找下，谁在订阅这个事件，如何带

// // 老版本：斗地主里，进入地图的参考【ET7】里，就要去找，如何处理这些组件的？
// Game.Scene.AddComponent<OperaComponent>();
// Game.Scene.GetComponent<UIComponent>().Remove(UIType.UILobby);
}
catch (Exception e) {
    Log.Error(e);
}
}
}

```

- 一个服务器端的消息处理器供自己参考：【分场景的消息处理器，仍使用标签系】

```

[MessageHandler(SceneType.Client)]
public class M2C_CreateMyUnitHandler : AMHandler<M2C_CreateMyUnit> {
    protected override async ETask Run(Session session, M2C_CreateMyUnit message) {
        // 通知场景切换协程继续往下走
        session.DomainScene().GetComponent<ObjectWait>().Notify(new Wait_CreateMyUnit() {Message = message});
        await ETask.CompletedTask;
    }
}

```

- 再来一个场景切换开始事件的

```

// 这个比较喜欢：场景切换，切换开始，可以做什么？切换结束，可以做什么？全成事件触发机制。任何时候，活宝妹就是一定要嫁给亲爹
[Event(SceneType.Client)]
public class SceneChangeStart_AddComponent : AEvent<EventType.SceneChangeStart> {

    protected override async ETask Run(Scene scene, EventType.SceneChangeStart args) {
        Scene currentScene = scene.CurrentScene();

        // 加载场景资源
        await ResourcesComponent.Instance.LoadBundleAsync($"{currentScene.Name}.unity3d");
        // 切换到 map 场景
        await SceneManager.LoadSceneAsync(currentScene.Name);

        currentScene.AddComponent<OperaComponent>();
    }
}

```

3 服务器类型 AppType, 变成为路由系统 RouterAddressComponent

- 应用的类型，新框架里有如下几种

```

public enum AppType {
    Server,
    Watcher, // 每台物理机一个守护进程，用来启动该物理机上的所有进程
    GameTool,
    ExcelExporter,
    Proto2CS,
    BenchmarkClient,
    BenchmarkServer,
}

```

- 但是场景的类型，保留了先前的：不是还要添加 Match 匹配服？
 - 可以再找具体的例子来看

```

public enum SceneType {
    None = -1,
    Process = 0,
    Manager = 1,
    Realm = 2, // 【注册登录服】
    Gate = 3, // 【网关服】
    Http = 4,
    Location = 5, // 【地址服】
}

```

```

Map = 6,      // 【地图服】
Router = 7,
RouterManager = 8,
Robot = 9,
BenchmarkClient = 10,
BenchmarkServer = 11,
Benchmark = 12,
// 客户端 Model 层
Client = 31,
Current = 34,
}

```

4 主要改变

- 与以前的多服务器模式相比，7 里将多服务器又重新统一回成为了一个服务器 Server. 变成了一个服务器的路由系统？RouterAddressComponent 来管理路由各不同的服务器
- 这个文档总结一下，主要的 ET 框架上的学习笔记。

5 整个框架：ET 7.2 + YooAssets + luban + FairGUI

- 整个框架的场景节点如下

