开发篇

后缀说明

System	系统
Domain	领域，位于系统层内部，无状态
Context	存储 Entity 和系统状态的地方
Repo	位于 Context 内部，专门存储特定 Entity
Entity	实体，位于系统层内部，有状态，存储数据
Component	用于概括 Entity 里的字段，避免平铺字段太多；也是状态的概括
Model	和 Component 类似，但是更独立。比 Component 更轻量，可以位于 Component 内部。
App	应用，位于系统下层，囊括了所有表现层（UI、音效、后效等）
Infra	比 App 更底层的基础设施
XXXDomain	如 UIXXXDomain，说明是 UIApp 的下层 Domain，位于具体 App 的内部
Panel	UI 特有的后缀
Element	Panel 的组件
DTO	Data Transfer Object，Entity 和 存档数据之间的中介
Service	比较轻量的自带状态的服务，类似于 Entity，但是服务性质大于独立的实体性质。比如 RandomService、IDService

代码入口

ClientMain

核心 System 介绍

LoginSystem

控制登录界面

此状态与游戏状态互斥，进此则退彼

SettingSystem

控制设置界面

通过 Login 和 Game 都可以跳转到它

和 Login 互斥，进此则退彼

和 Game 不互斥，但进 Game 退 Setting、进 Setting 暂停 Game

PauseSystem

在 Game 中可以转到 Pause

进 Pause 则暂停 Game、进 Game 则退 Pause

Pause 可以跳转到 Setting，两者互斥，进此则退彼

GameSystem

控制点“NewGame”进入“游戏”后的一切。

- 核心类：
  - GameDomain：处理进游戏、退游戏、重玩游戏
  - StageDomain：处理进关卡、退关卡、进下一关
  - StageFSMDomain：核心状态机，负责几个 Stage 状态间过渡
  - EdgeGodFSMDomain：核心状态机，负责从 Edge 任务队列中取出待触发的 Edge，然后执行对应逻辑。具体见 Edge 的控制流设计

System 的通信

基于事件，具体见 ClientMain 类的 Binding 部分。

事件的静态阅读方法见 事件的静态阅读方法

逻辑层和表现层的通信

XXXSystem、XXXDomain 都位于逻辑层，

UIXXXDomain 位于表现层

逻辑层可以通过 UIApp 的 API，直接访问、控制表现层

表现层不能直接访问、控制逻辑层，但可以通过上层代码订阅的事件，实现控制反转

具体可以见 ClientMain 的 UI_Binding 部分

每一层的调用风格

- ClientMain
  - 为了更好的可读性，以及更好地处理控制流程，在 Main 的 Tick、Binding 中尽量不要跨级出现 Domain，尽量只操作 System。
- System 层
  - 为了更好的可读性，尽量只调用 Domain 里的方法，而不跨级去调 App 里的方法。而且调用 Domain 的时候，Domain 里往往不仅会调用 App，还会做一些额外的数据加工。
  - System 和 System 不支持互相调用，需要通讯只能基于事件
- Domain 层
  - 支持 Domain 互调。
  - 可以直接调用 App 层，可以访问 Entity、Context。
- Entity 层
  - 状态容器，只能被调用，不能调用别人。
- App 层
  - 开放 API 给上层调用。

原则上，ClientMain 和 System 里的代码越简洁越好，Domain 里可以写得具体一些，部分方法也可以封装到 Entity 里，这样每一层可展示的语义就会尽量简明易懂。

XXApp 类提供的是 API，原则上不允许包含逻辑，只能作为中间层，嵌套下层方法。

一些特殊的 Domain

为了确保上面的书写原则，会涉及到「多加几层包装」的情况。

比如，在 GameSystem 里想要调用 UIApp，就不会直接调用，而是调用 UIDomain。

这里的 UIDomain 本质上是 GameUIDomain 的简写，它位于逻辑层而非 UI 层。

这里的依赖关系是：

System → UIDomain → UIApp → UIXXXDomain

不要嫌麻烦，UIDomain 大部分时候是对 UIApp 再包装一层，但也有例外，比如 GameInfo_Open、Setting_Open 这些方法里，可以看到除了调用 UIApp，它们还额外做了很多事。

原则上 UIDomain 支持查询逻辑层的数据，因为它自己就位于逻辑层。UIDomain 只处理纯表现层的业务，不去修改任何逻辑层数据。

UIXXXDomain

比如 UILoginDomain、UISettingDomain

这些 Domain 是 UI 层内部的 Domain，原则上每个 Domain 只负责自己所管理的 Panel ，不允许互相调用。它们不像逻辑层拥有那么多互操作、互访问的需求。