robomaster game 软件架构

- 参数配置
  - 开发者参数配置
    - Assets/my_project/project_settings中使用static类进行参数配置
    - 改进 — 如果使用scriptable object 可以在editor运行中修改参数
  - 用户参数配置 — 使用yaml文件进行配置的保存和读取
- 游戏流程
  - 进入游戏
  - PreGameConfiger 进行参数文件选择，选择作为host或client进入游戏
  - NetworkManager 生成Player Prefab(Robot)加入网络
  - Robot中的NetSpawn进行位置初始化并从PreGameConfiger中取出参数存进StateStore — 改进
    - 改为RobotTransform初始化位置更符合面向对象
    - 改为StateStore取出参数更符合面向对象
  - Robot组件运行相应功能
    - RobotTransform 在服务器端修改Player Prefab位置
    - InputMange 在客户端读取输入并传出
    - CameraManage 在客户端读取相机位置并传出
    - LigntManage 接收灯条颜色信息并传出
    - StateStore 存储各种信息
    - NetSyncManage 处理网络的发送接收
    - RobotGhost 实例化 Robot Entity，并对其进行处理
- 数据流向
  - PreGameConfiger(InGameConfig) — -> StateStore(InGameConfig)
  - CameraMange() — StateStore(camera_rotate_y, state.vision_mode)
  - InputManage(InputNetworkData) — -> NetSyncManage() — --Server
    - InputNetworkData -> NetSyncManage() — --client — InputSyncData -> ShootControl()
    - InputNetworkData -> RobotTransform()
  - NetSyncManage(PlayerNetworkState) — --client — NetSyncManage(PlayerNetworkState)
  - NetSyncManage() — --client — --server — --client — StateStore(InGameConfig)
- 事件触发API
  - UniRx
    - 触发 — Subject.OnNext
    - 订阅 — Subject.Subribe
  - IRobotComponent接口在对象树中共用一个_subject，使得对象树中实现这个接口的组件都能共享
  - Referee — 暂定使用Singleton，存储所有玩家，并进行后台的逻辑处理
- ECS 实体系统
  - 子弹发射与物理计算 — 由ShootControl()组件 处理实体
  - 与装甲板碰撞系统 — 使用Collison Event检测碰撞

StateStore(InGameConfig)
- IdentityId -> ShootControl()
- -> LightManage() — LightData -> LigntControl()