# 说明

**当前内容都可以删除，主程序可重新分配结构和说明以规范文档的结构和书写**

为了查阅方便，个人建议：

1. 对文档进行分节处理，以便在world左下部分->导航->标题就可以清楚的查阅文档结构
2. 当前分节为示例，一切架构主程分配即可
3. 有特殊个人内容不包含公共架构（如个人编辑器等），可新建同层级的标题并以`主题-自己的名字`作为上层标题，并在下方建立层级/依照喜好说明
4. 不对文档过多分层，建议最多三层

# 基本架构

当前本节为示例说明，可删除

## UI架构

1. UI基类

原型

public abstract class BasePanel<T>

继承方法

子类名称:BasePanel<子类名称>

功能函数

-Update():在基类中提供基本的显示隐藏渐变效果或其他功能

-Awake():初始化Update中拥有功能的组件

-Start():调用Init,子类不必再写Start,而是使用父类

函数接口

+Show():显示当前面板

+Hide():隐藏当前面板

-Init():abstract的,其中书写初始化逻辑,建议为面板非public字段初始值,各个UI组件的事件注册,面板的初始化事件注册等

## 数据存储

### 1.CSV表

存储所有供策划配置的游戏预制信息

新增表格流程：

①和策划确认表头；表格功能及注释，查阅策划的表格需求文档：

https://docs.qq.com/sheet/DVVVtcEJsSlRkZ0dP?u=2ec52556d5d54a189d339217ec9f6025&tab=p3jq83

②新增csv表格放置到对应路径：Asset/Resources/Table

③新增表格类型，定义表格字段，重写处理表格方法。

表格结构基类：

VecTable<TV> : TableBase

MapTable<TK, TV>: TableBase

JsonTable<TV>: TableBase

④在tableManager的AddTables新增表格处理。

⑤参考：

技能表-AbilityTable.cs-AbilityData.csv；

角色表-BattleUnitTable.cs-BattleUnitData.csv。

读表：

TableManager.LoadTables():在创建游戏实例时自动读取并处理所有表格，存储数据在对应DataList。后续读取DataList里的数据即可

|  |
| --- |
| //更多是通过数据对应类型的工厂里的创建方法，来读取数据+创建实例，比如AbilityFactory.CreateAbility(int abiID).    public static TV Find(TK key)  {  if (DataList.TryGetValue(key, out var rst))  return rst;  return default;  } |

配置相关：

数组分割符号：|

ID结构：（待和策划定

## 重要类间继承

## 重要的接口和调用关系说明