程序设计文档

1. **基本信息**

项目名称：简单卡牌游戏

开发环境：coocs2dx 3.17.2

开发工具：visual studio 2022

版本号：0.1

作者：林炜昊

日期：2025.5.31-2025.6.3

1. **概述**

设计目标：实现游戏系统中简单的点击匹配以及回退功能

功能描述：点击匹配、回退

设计原则：采用MVC架构，视图、逻辑、数据分离，高内聚低耦合

1. **系统结构设计**

模块划分与说明：

Classes/

├── configs/ # 所有静态配置相关类

├── models/ # 运行时动态数据模型

├── views/ # 视图层，包含所有的UI展示组件

├── controllers/ # 控制器层，协调模型和视图

├── managers/ # 管理器层，提供全局性的服务和功能

├── services/ # 服务层,提供无状态的服务，处理业务逻辑，不管理数据生命周期；

└── utils/ # 工具类，提供通用功能

1. **类与接口设计**
2. 核心类设计

GameController类：使用单例模式，同时只会运行一场游戏。处理游戏初始化、视图加载等操作。

CardController类：控制卡牌视图与模型的交互，进行卡牌点击回调函数的初始化。

CardStack类：包装了vector进行卡牌堆的存放。

UndoManager类：包含一个UndoModel的栈用于存放记录。调用undo方法可以直接进行一次回退（游戏运行时可用）。

模型类存放所有有关的数据。

视图类定义了所有有关视图，但是部分运行时初始化交由控制器层实现。

1. 关键数据结构设计：

CardModel中使用枚举值定义花色和面值。

UndoModel中直接存放与操作有关的两个卡牌模型的指针、移动的卡牌属于哪个区域、移动的卡牌位于原数组的哪个位置和原来在节点中的Position。

1. **流程设计**

启动流程：

启动引擎->实例化游戏控制器->初始化游戏内容->启动游戏->加载默认场景->加载所有模型与视图->将场景设置为当前场景->等待用户操作

1. **异常与边界处理**

可能出现的异常：资源加载失败

处理方式：控制台输出加载失败的资源路径

1. **配置与资源说明**

**配置文件结构：**

关卡配置文件levelConfig.json：

{

"Playfield": [

{

"CardFace": 12,

"CardSuit": 0,

"Position": {"x": 250, "y": 1000}

},

……

],

"Stack": [

{

"CardFace": 2,

"CardSuit": 0,

"Position": {"x": 0, "y": 0}

},

……

]

}

如果有多个关卡需要加载请在levelConfigLoader中修改loadAllLevelConfig方法

Fonts文件夹中存放使用到的字体

Strings文件夹中存放游戏中会用到的中文字符（避免中文乱码问题）

Res/number文件夹中存放数字贴图（大号和小号）

Res/suits文件夹中存放花色有关贴图

**资源加载策略：**

图像资源：一次性加载到TextureCache中，需要时取用

字体资源：不预加载，仅在被用到的时候读取（只有在“回退”按钮用到）

Strings资源：不预加载，仅在被用到的时候读取（只有在“回退”按钮用到）

关卡配置资源：游戏初始化时一次性加载到关卡列表中（<int, Document>键值对）

1. **扩展性与变更说明**

已知可变需求：

1. 添加新卡牌：卡牌的设定应该与先前的设定相同，只支持扩展新的花色和新的大小（需要添加相应的贴图资源）。

2、新类型的回退功能：回退功能已经包装在一个函数中，如需多次回退或者回退到初始状态只需要反复模拟点击（触发回调函数）即可。

后续功能扩展：

1. 添加启动场景和关卡选择场景：游戏启动的接口预留在GameController的startGame(int level)中。选择界面完成后只需调用此方法即可进入游戏。
2. 回退功能的序列化和反序列化：目前的存档仅支持游戏数据存档，如果回退记录也要支持进行存档功能，最好增加弃牌堆进行保存。
3. **总结**

项目目前功能已满足所有需求。由于项目处于十分初期阶段，扩展性需求需按情况更改结构。