《软件设计文档》

1. 软件技术选型理由
2. 开发平台

候选1：Linux

候选2：Windows

最终选择：Windows

选择原因：

1. Windows的开发工具丰富，开发环境搭建难度较低
2. Windows系统是大部分目标用户的常用系统，开发基于此系统的软件符合用户需求
3. 软件形式

候选1：桌面应用

候选2：WEB应用

最终选择：桌面应用

选择原因：

1. 一般的游戏开发都是以桌面软件的形式，各种游戏引擎，开发工具库齐全且成熟
2. WEB应用一般需要网络和服务器支持，游戏质量容易受到网络服务器的影响
3. WEB应用需要依赖浏览器，需要进行不同版本的浏览器兼容这一额外工作
4. 我们的游戏是单人游戏，没有多人互动的信息交互需求
5. 框架选择

候选1：Unity3D

候选2：Cocos2d-x

最终选择：Cocos2d-x

选择原因：

1. Cocos2d-x是一款开源的且十分成熟的游戏引擎软件，具有优秀的性能以及较为全面的技术文档，能够很好地满足我们的游戏开发需求。
2. Cocos2d-x使用C++进行游戏开发，语言门槛较低。
3. 小组成员都曾经专门学习使用Cocos2d-x进行游戏开发，有一定的编程经验。
4. 软件架构设计

游戏主进程由Cocos2d-x游戏引擎主导，负责各个自定义游戏场景类的调度和渲染。

场景类负责游戏场景的设计以及用户交互逻辑的实现，供Cocos2d-x游戏引擎调用。

资源类中的成员函数提供游戏资源的加载与控制的数据接口，供各场景类使用。

资源类中的变量用于保存已加载的游戏资源以及游戏实时状态信息。

游戏信息使用配置文件进行持久化，玩家信息使用SQLite数据库进行保存。

1. 软件模块划分

本作品可分为以下几个模块：

1. 游戏菜单模块：主要用于游戏页面的切换控制
2. 游戏主模块：主要进行与玩家的交互以及游戏逻辑的实现
3. 游戏音乐模块：主要用于实现游戏音乐的控制
4. 游戏砖块生成模块：主要用于生成游戏中的跳跃砖块
5. 游戏奖励生成模块：主要用于生成游戏中的奖励物品
6. 游戏数据库模块：主要用于保存游戏设置以及游戏玩家的相关信息
7. 游戏设置模块：主要用于设置游戏音乐等游戏内容
8. 游戏规则说明模块：主要用于显示游戏的规则说明等内容
9. 游戏排行榜模块：主要用于显示游戏最高历史得分等内容
10. 游戏提示模块：主要用于显示游戏提示信息等内容
11. 游戏资源模块：主要用于存放本项目所使用的图片，音乐等内容
12. 游戏自定义地图模块：主要用于让玩家自主设计游戏地图。
13. 游戏测试模块：主要用于测试数据库交互，音效播放以及游戏内物品的属性。
14. 软件设计技术
15. Structure Programming（结构编程）

以MainScene中的init函数为例，其执行操作顺序如下：

|  |
| --- |
| 代码分析 |
| 1. 渲染基础场景      1. 获取当前游戏窗口的大小以及窗口原点      1. 获取玩家基础信息      1. 创建玩家以及玩家的物品      1. 创建游戏场景以及事件监听器      1. 创建轮询事件监听器      1. 渲染场景内物品      1. 更新场景内物品的状态 |
| 所有代码 |
|  |

1. Object-Oriented Programming（面向对象编程）

本作品中的每个功能模块都被抽象成一个类，而模块中的所有操作都被封装成每个类的成员函数。

|  |  |
| --- | --- |
| 类名 | Database |
| 说明 | 用于与SQL数据库进行数据交互的模块。这个模块拥有的内部方法总共有三大类：一是用于保存玩家游戏状态，如：记录玩家的生命值，玩家的跳跃次数等；二是用于保存玩家得分，如：保存玩家本局游戏得分以及历史最高分等；三是用于查询玩家信息，如：获取玩家当前信息，玩家历史得分排行等。 |
| 代码 |  |
| 类名 | Music |
| 说明 | 用于管理游戏音乐资源。这个模块的操作主要有总共有两大类：一是游戏资源的加载，如：不同类型游戏音乐资源的加载；二是游戏音乐的播放控制，如：游戏音乐的暂停与播放，游戏音量的调节等。 |
| 代码 |  |
| 类名 | propsFactory |
| 说明 | 用于游戏奖励物品的生成。这个模块的成员函数总共有三大类：一是游戏基础环境的设置，用于管理物品渲染时的基础场景；二是奖励物品的生成函数，用于生成如：飞鞋，金币等的奖励物品；三是奖励物品的更新函数，用于根据游戏进度实时更新已有的奖励物品的属性，使其能够被合理地渲染。 |
| 代码 |  |
| 类名 | BlockCreator |
| 说明 | 用于生成游戏内的砖块。这个模块的功能分为三大类：一是新砖块的生成函数，根据一定规则生成新砖块；二是砖块的获取函数，这部分供主场景进行调用，用于获取已有砖块和新砖块的信息；三是砖块状态的更新函数，主要用于根据游戏进度调节砖块的生成，以及根据已有砖块的位置更新装块的位置。 |
| 代码 |  |
| 类名 | RuleScene |
| 说明 | 用于渲染规则说明页面的场景。这部分主要是有两部分的功能：一是重写了Scene类的场景渲染函数，绘制设置说明页面场景；二是为场景内部的按钮设置事件回调函数，响应用户输入。 |
| 代码 |  |
| 类名 | SettingScene |
| 说明 | 用于渲染游戏设置页面的场景。这部分主要是有三部分的功能：一是重写了Scene类的场景渲染函数，绘制设置页面场景；二是为场景内部的按钮设置事件回调函数，响应用户输入；三是游戏资源管理，将用户自定义的游戏设置保存到游戏文件中实现设置的持久化。 |
| 代码 |  |
| 类名 | RankScene |
| 说明 | 用于渲染游戏排行榜页面的场景。这部分主要是有三部分的功能：一是重写了Scene类的场景渲染函数，绘制排行榜页面场景；二是为场景内部的按钮设置事件回调函数，响应用户输入；三是游戏数据交互，从数据库中查询用户的历史游戏得分并显示。 |
| 代码 |  |
| 类名 | MenuScene |
| 说明 | 用于渲染游戏菜单页面的场景。这部分主要是有三部分的功能：一是重写了Scene类的场景渲染函数，绘制菜单页面场景；二是为场景内部的按钮设置事件回调函数，响应用户输入，进行页面跳转。 |
| 代码 |  |
| 类名 | Hit |
| 说明 | 用于渲染游戏提示信息页面的场景。这部分主要是有三部分的功能：一是重写了Scene类的场景渲染函数，绘制提示信息页面场景；二是为场景内部的按钮设置事件回调函数，响应用户输入，进行页面跳转。 |
| 代码 |  |
| 类名 | MainScene |
| 说明 | 用于渲染游戏主界面的场景。这部分主要是有三部分的功能：一是重写了Scene类的场景渲染函数，绘制游戏主页面场景；二是为场景内部的按钮设置事件回调函数，响应用户输入，进行页面跳转；三是游戏资源控制，负责与各资源模块进行数据交互，实时更新游戏内物品的状态和属性。 |
| 代码 |  |
| 类名 | AppDelegate |
| 说明 | 游戏主控制类。负责游戏窗口的建立以及场景的初始化。 |
| 代码 |  |

1. Design Patterns（设计模式）

项目中对资源管理模块均使用了单例模式，场景渲染模块则使用了工厂模式。

|  |  |  |
| --- | --- | --- |
| 设计模式 | 样例（节选） | 说明 |
| 单例模式 |  | 使用一个静态Database类指针。外部函数可以通过调用Database::getInstance()这个静态方法获取全局唯一的Database类实体。 |
| 工厂模式 |  | 每个场景渲染类都继承了cocos2d::Scene这个基类，不同的场景类通过重载基类中的init()函数实现不同场景的渲染。 |

1. 外部软件库以及资源

（1）Cocos2d-x游戏引擎程序文件（cocos2d.h）

（2）Sqlite3开源数据库引擎程序文件（sqlite3.h/sqlite3.cpp）

1. 测试

本项目的测试主要分为两个部分：数据测试和画面测试

数据测试部分主要测试玩家数据与数据库的交互以及音乐资源的加载和处理函数，使用Test类进行测试；画面测试主要测试游戏物品和游戏逻辑的实现，使用Cocos2d-x内置的DEBUG模式进行测试。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据测试部分 | | | | |
| 测例 | 测例说明 | 测例代码 | 测例输出 | 测试  结果 |
| 测例1 | 测试玩家生命值保存与读取 |  |  | 通过 |
| 测例2 | 测试玩家跳跃次数保存于读取 |  |  | 通过 |
| 测例3 | 测试玩家历史最高分的保存与读取 |  |  | 通过 |
| 测例4 | 测试玩家本局得分的保存与读取 |  |  | 通过 |
| 测例5 | 测试排行榜的输入与读取 |  |  | 通过 |
| 测例6 | 测试背景音效的播放 |  |  | 通过 |
| 测例7 | 测试背景音效的暂停 |  |  | 通过 |
| 测例8 | 测试背景音效的停止 |  |  | 通过 |
| 测例9 | 测试特效音效的播放 |  |  | 通过 |
| 测例10 | 测试特效音效的停止 |  |  | 通过 |

|  |  |
| --- | --- |
| 画面测试部分 | |
| 测试代码 |  |
| 测试画面 |  |
| 测试结果 |  |