项目总体架构

我们将本项目分为前端实现和后端实现。前端实现主要负责加载各种图像，后端实现则负责读取、修改游戏配置文件以及玩家存档中的内容。

前端实现

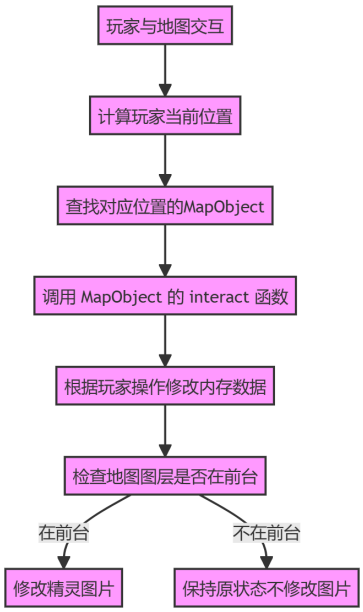
前端实现又分为地图图层和 UI 图层，而两部分又使用一个单例类（ SceneManager 类）统一加载、管理。本项目中，每张瓦片地图都被视作一个独立的场景，而 UI 图层被添加至一个永久节点中，每次切换场景时，都会组装整张地图，并将地图图层和永久节点添加到场景中，假如要展示 UI 场景，可以通过调整 UI 图层的上下顺序实现。

后端实现

后端实现又分为时间管理器，主角类和各种地图对象。时间管理器是一个单例类，在加载存档时实例化，在卸载存档时析构。负责管理游戏场景内的所有时间，在进入下一天时调用结算函数，进行清算并存档。主角类是一个单例类，同样在加载存档时实例化，在卸载存档时析构。负责管理游戏中主角的所有信息的修改和存档的写入，包括：持有物品，精力，金钱等。各种地图对象都继承自一个 MapObject 类，该类是一个抽象接口类，该类中预留了各种接口函数，可以由地图图层统一管理、调用。

游戏场景实现细节

游戏场景中的函数调用顺序：



从文件创建游戏场景

我们将场景的具体信息写入文件（存档、配置），每个地图图层都对应一个独立的场景文件，该场景文件中包含了该地图图层的名字、瓦片地图、背景音乐、背景颜色等各项信息。

该文件还包含一个在该图层上的常驻对象列表；在存档中，还有一个可修改对象列表。对象列表中储存了所有对象的必要信息，包括位置、对象的种类以及每个对象的独立信息。

地图逻辑常驻加载

在本项目中，地图图层的数据逻辑是常驻加载的，而图像只有在主角在对应场景时才会加载。每当存档加载时，SceneManager 会创建所有地图图层（一个 MapLayer 类实例），在创建地图图层时，不会加载任何一个精灵，也不会读取任何一个图片，MapLayer 只会从场景文件读取所有的逻辑信息并加载。这样的设计可以方便网络层修改数据，为后续开发预留了接口。而在玩家进入到对应的地图场景时，对应的地图图层才会被加载，其中地图对象的图片信息也会对应加载。

工厂函数方法创建与交互地图

地图图层上的所有地图对象都继承自一个抽象接口类：MapObject，该类继承自

cocos2d-x 中的 Ref 类，方便使用自动释放池管理对象的生命周期。

当 MapLayer 创建时，都读取两个场景文件中的对象列表。MapObject 的创建使用一个工厂函数，该函数为 MapObejct 中的 static 方法，该函数会读取对象列表中的类型参数决定调用哪个具体子类的的工厂函数。该工厂函数会返回一个 MapObejct 指针。

为了储存所有的 MapObejct 指针，我们将整张地图以瓦片为单位离散化，将其抽象为一个二维数组，其中储存所有 MapObejct 对象的指针，我们将这个储存所有对象指针的二维数组称为交互地图。我们可以使用交互地图快速的查找到对象的指针，实现快速的对象查找、创建、交互、删除操作。

MapObject 抽象类

MapObject 是一个抽象接口类 ，该类中有多个接口，用于实现不同功能。

- **init** init 函数用于加载当前对象的图像信息，该函数会在地图图层被加载时调用。
- **clear** clear 函数用于在场景进入后台时清除对象掌握的无效信息，防止 Object 对象使用悬挂指针。
- **interact** ineract 函数用于在玩家与地图对象交互时调用
- **settle** settle 函数用于在对象每天结算时写入存档
- **static create** 部分子类含有，如耕地在游戏时创建。
- **继承自此基类**：farm,crop,npc,animal,gate,mineral

UI 实现细节

UI 实现细节

UILayer

UILayer 类管理游戏中的 UI 的图层的创建和初始化，在该类中根据功能创建所有的 UI

图层并且初始化图层上的按钮和文本元素。主要功能包括：

- **initWithType**：根据不同的 UI 类型调用 UILayer 和 UIlogoc 中的初始化函数初始化 UILayer，该函数在 SceneManager 实例化时被调用，创建出 UI 图层的 permanent node。

- **getTypeLayout**：根据不同的 UI 类型返回不同的 UILayer。

- **createTypeLayout**：根据不同的 UI 类型初始化 UILayer，该函数在 UILayer 的 initwithType 函数中被调用。

UILogic

UILogic 类管理游戏中的 UI 的图层的更新逻辑和显示逻辑，在该类中根据功能修改 UI 图层上的显示并给出相应的反馈。主要功能包括：

- **initTypeNode**：初始化 UILogic 类中的各个图层的 Node 使得其与 UILayer 中的各个图层相对应。

- **bindTypeEvents**：根据不同的 UI 类型给相应的按钮绑定相应的事件监听器和回调函数。

- **updateBagItems**：根据 MainCharacter 传入的 inventory 数组更新 bagItems，该函数在 MainCharacter 中被调用后会调用 refreshBagUI，更新背包栏的物品显示情况以及可点击状况。

- **refreshtime/power/money/levelUI**：根据 TimeManager 传入的时间以及 MainCharacter 传入的 power/money/level 更新在 timeUI 上各个数据的显示。

- **refreshNpc/FishUI**：npc 会根据传入的人物修改不同的 portrait，两个函数都会在被调用时随机输出对话 / 钓上来的物品。

- **refreshArchiveUI**：根据现有的存档个数修改 loadArchiveUI，使其正确显示并可供选择。

- **LoadArchive**：在 loadArchiveUI 上的按钮的回调函数中调用，根据不同存档的 index 去内存中读相应的存档数据，并通过该数据初始化游戏。

- **onButtonClicked**：不同按钮的回调函数，在按钮被按下后调用。

后台数据实现细节

MainCharactor

MainCharacter 类代表游戏中的主角，包含多个属性和方法来管理角色的状态、物品和能量等。类采用单例模式，确保在整个游戏中只有一个主角实例。主要功能包括：

- **体力管理**:提供修改体力和获取当前能量的功能。体力随着不同操作的消耗(如耕地、浇水等）变化，可以通过食物恢复（如花菜、土豆、南瓜等）。
- **物品管理**：角色有一个背包（inventory）来存储物品。可以通过添加、删除或修改物品数量（modifyItemQuantity）来管理物品。可以检查背包中是否包含特定物品（hasItem），并设置当前持有的物品。
- **金钱管理**：角色拥有金钱属性，提供修改金钱和获取当前金钱的功能。
- **角色等级**：角色有等级（level），每次升级可以触发奖励（如 level_gift()）。提供修改等级和获取等级的功能。
- **单例模式**:使用 getInstance() 方法获取唯一的主角实例，禁止直接创建实例或拷贝。
- **归档功能**：支持从 JSON 文件加载物品数据（loadInventoryFromArchive），并能在内存中更新归档数据。

TimeManager

TimeManager 类用于管理游戏中的时间流逝和相关的事件调度。它采用单例模式，确保在游戏中只有一个时间管理实例。以下是该类的主要功能概述：

- **时间管理**：

该类跟踪当前游戏的天数（current_day_）和时间（current_time_，以小时为单位）。

每个游戏日的时间以游戏内小时为单位，且现实中的 10 分钟等于游戏中的 1 天（MinutesOfOneDay = 10）。

提供 updateTime(float dt) 函数来实时更新游戏时间。

- **季节**：

游戏内的季节由 season 字符串表示（例如春季、秋季等）。

- **节日：**

  根据设定的节日天数（如 festival_days_），判断是否为节日（isFestivalDay()），并限制特定节日活动。

- **主角与存档管理**：

提供 startNewGame() 方法用于开始新游戏并初始化时间。

提供 sleep() 方法模拟主角睡觉，并自动保存游戏进度。

endOfDay() 和 settleAllObjects() 方法用于每日结束时结算所有对象的状态并保存进度。

- **存档与数据保存**：

saveGameData() 用于在主角睡觉时保存游戏数据，确保存档文件更新。

- **单例模式**：

getInstance() 用于获取时间管理的唯一实例，并在 SceneManager::createMaps() 中调用，启动计时。

cleanup() 用于清理单例实例。

存档管理

存档和 json 文档的读取和写回统一由 DocumentManager 这一单例类实现

- 初始存档，新建存档时把以下文件复制进存档

Resource/NewUerArchive 游戏存档（包括每个地图存的东西等）

Resource/NewUsrConfig 设置（包括语言选项等）

- 游戏启动时把游戏信息从硬盘读入内存或初始存档读入内存

游戏交互时在抽象类中更改，部分直接更改内存，部分在 24 点更改内存

睡觉时游戏信息内存到硬盘

文件系统

文件统一使用 json 作为储存格式

- 读 配置文件

Resources/global 是 Resource 所有文件的目录

Resources/config 配置文档 存 json 文本

Resources/fonts ui 字体（包括新宋体等字体库）

Resources/image ui 的 png 图片

Resources/tiledmap 瓦片地图 不用了解

Resources/plist plist 资源 物品的 png 图片

硬盘中游戏信息路径

C:\Users\<username>\AppData\Local\FarmingGame

- **JSON**：用于存储和传输结构化数据。
- **PLIST**：用于描述图集，通常与 PNG 图像配合使用。
- **PNG**：用于存储图像资源，通常被 PLIST 文件引用。

项目分工

左凌旭

- 项目环境配置
- 项目设计、分工、协调
- git 仓库管理
- MapObject 抽象类接口设计
- MapLayer 地图图层类，包括地图图层创建、加载、销毁函数，结算函数，一系列事件监听器以及响应的回调函数，地图图层上所有图像的添加和管理。
- SceneManager 管理所有图层，包括展示和隐藏 UI、切换到下一个场景、结算函数的总接口
- DocumentManager 存档和 json 文档的管理器，可以从硬盘读取存档到内存，也可以将存档写回硬盘，可以管理所有玩家存档
- PlayerSprite 可以移动精灵类，本类包装了 cocos2dx 中有关于精灵的接口函数，可以让一个精灵上下左右移动并做出一些动作。

王相

- MapObject* farm，crop 嵌套结构

  玩家可以耕种、种植并收获多种作物，作物的成熟时间根据种类和季节变化。

  农场操作包括耕地、浇　、施肥以及季节变换，未及时处理会导致作物死亡。

- MapObject* mineral

  挖掘矿物，矿物可以定时刷新

- MapObject* gate
- MainCharacter

  管理主角物品，如背包物品，手持工具，种　和肥料存储。

  体力消耗和恢复，判断当前体力能否完成当前动作

  游戏等级

  金币管理
- TimeManager

  时间管理和睡觉
- 上述内容能存档

薛皓天

- StartScreenUI　游戏的开始界面，包含NEW（创建新存档）、LOAD（加载loadArchiveUI）、EXIT（退出游戏）三个选项。
- BagUI　游戏的背包界面，常驻在台前。显示玩家其实拥有的物品种类以及数量，并能够及时更新。通过点击不同的格子来选中背包中的物品实现不同的功能，被选中的物品会有红框提示。
- TaskBarUI　游戏的任务界面，通过按"E"键来调出。如果任务未完成，则会显示为暗，不可点击。如果任务完成，则可以被点击，点击后该任务会消失。
- NpcUI　游戏的 NPC 对话界面，在点击 NPC 后被调出。与 NPC 的对话会随机显示，同时，左下角会显示你与该 NPC 的亲密度来显示你们当前的关系。
- LoadArchiveUI　游戏的存档界面。在 StartScreenUI 点击 LOAD 被调出。上面显示了所有存在的存档以及存档的天数和金钱。点击即可进入存档。
- TimeUI　游戏的时间、体力、等级、金钱显示界面。常驻在台前。会根据这些数据的变化实时改变显示。

- ShopUI　游戏的商店界面，通过按"F"键来调出。分为购买区和售卖区。购买区可以购置作物种子，售卖区可以售出成熟作物。

- ManufactureUI　游戏的制作界面 ,通过按"R"键来调出。通过点击该界面上的物品图标，可以消耗相应的原材料（如果原材料充足的话）来制作该物品。

- FishingUI　游戏的钓鱼界面。在海边钓鱼时调出。会显示你每次钓上来的东西，每次钓的东西并不相同。

- tiledmap 瓦片地图的制作　使用 tiledmap 制作了所有十张瓦片地图作为游戏的底层场景。

- 游戏使用的资源文件 image 和 plist 的切割以及组装　使用 texturepacker 和 pisxelart 拆分并组装了游戏所需的.png 和.plist 资源。

郭艺

- 养殖模块:玩家可以养殖动物，动物具有饱食度，价格和生育率的属性。玩家可以饲喂动物来提高动物的属性 ,不同的食物产生不同的效果 ,动物在夜间生长 ,价值生长提高，有概率会繁殖增加数量，饱食度过低会饿死而减少。

- 居民 NPC:NPC 会在特定的场合刷新和游走 ,玩家可以与 PC 进行交互 ,如送礼物提高亲密度，每个 NPC 特定的礼物会有额外加成。可以与 NPC 聊天

- 背景音乐:走入每个场景都会切换特定背景音乐