没有凋落物直接放入背包（背包足够大）、没有遮盖关系（全是碰撞体积）、右键与物品互动，左键与地图互动，全局变量放到HelperClasses.h

对于前两个类：

**（****任务一）Map对象：**

Crop ：

资源：植物贴图（不同阶段）、生长逻辑、描述；

逻辑：浇水、种植、生长、灌溉、收获（不加掉落物），全部直接交互（都+种植exp）

-----Crop不直接存为交互地图的指针，但是会由landform中的指针指向他。

landform ：

资源：农场、矿洞的地图；

逻辑：耕地：由锄地创建、可以浇水（+种植exp），其他情况交给Crop交互；水面：钓鱼（音游快速判定，+钓鱼exp）；传送门：切换场景；采矿：矿物持有（+矿物exp）

**（任务二）活物：**需要使用cocos2dx的schedule方法实现每隔一段时间就定时移动

Animal ：（养殖场在游戏开局即解锁）

资源：养殖场地图制作、固定范围走动的动画贴图、价格及产出、动物不同阶段要换贴图

逻辑：饲养、抚摸、售卖，交互给出对话框选择

NPC ：

资源：小镇（外加庆典版）以及NPC房子地图、NPC贴图、对话、人物喜好、庆典互动、固定范围走动的动画；

逻辑：交流（前端给出一个交流的窗口）、送礼（送出背包栏选中物品）、关系（需要由前面两个的效果决定增减）

**（任务三）菜单后台：（需要与前台更深讨论）**

（以下前台都给出一个类似的背包界面，制造部分有上下两个背包界面）

BagManager：

资源：所有可制造物品的贴图、所有物品的字段说明

逻辑：单例类，在地图界面，可以选中不同的物品，右键直接和这个物品互动，比如说吃东西（给出互动的接口）；在背包界面，需要有一个鼠标背包栏，，鼠标背包栏有东西的时候不能关闭界面，点击背包任意栏可以与鼠标背包栏进行交换。

CookBook：

逻辑：单例类，上下两个背包界面，上面的背包界面展示所有可制造物品，已解锁不可制造的显示灰色图案，已解锁可以制造的显示有颜色的图案，点击有颜色的图案，将物品放入鼠标背包栏。

（以下前台都给出一个相似的列表接口）

TaskManager：

资源：任务的文本

逻辑：单例类，在地图上放置特殊点，互动即可完成任务，可以相应的要求删减资源（或按顺序进行多次互动），或设置次数累计型任务（收获了多少作物，类似于技能）

Shop：

逻辑：购买与售卖，加减金钱和物品

SaveManager：

逻辑：单例类，选择存档，加载存档，保存存档，删除存档的接口

Config：

逻辑：单例类，更改全局配置，包括是否全屏，是否静音

写AudioEngine的接口

**（任务四）前端：**

SceneManager：

逻辑：单例类，管理所有图层，给出图层切换、场景切换的接口

MapLayer：

逻辑：展示tmx地图，展示人物动画，通过人物动画的监听效果，展示地图移动，人物移动

UILayer：

资源：任务栏和背包的框图，开始画面

逻辑：展示所有任务栏图片

Player：

资源：玩家的所有动作动画，使用工具，WASD走动

逻辑：不属于可交互对象，给出动作接口，包括钓鱼，走动等

HelloSene：

资源：开始界面的所有资源，对应的按钮对应特别的效果等

其他：（自己接活）

前后端交互接口：虚基类，工厂模式（谁第一个做先把这个搞出来）

ToNextDay：将所有天数加一、存档、接算的全局函数

高亮表示已完成

设计理念

一、场景

场景统一管理：前台设计

由SceneManager统一加载、管理所有场景，将场景的具体配置写入文件，通过文件中的内容加载场景

二、物品

**1.继承关系**

引擎定义：Ref（基）

Note

Sprite

程序实现：每一个物品如耕地

有继承自Ref的Object

Object中有一个\*sprite

**2.调用时机+主要功能**

1. 启动游戏： 遍历数据库，Object对象实例化
2. 走入当前地图 Sprite对象实例化，渲染画面，即切换场景
3. 交互
4. 一天结束
5. 关闭游戏

**3.Object.h文件，对Object**

static Object\* create(rapidjson::Value& val, MapLayer\* parents);

1. 作用：由数据库创建Object对象，\*Object插入交互地图，\*sprite=nullptr
2. 被调用时机：启动游戏，也就是“加载存档”
3. 被调用方法：前端调用此函数，工厂模式根据json调用对应类（如Land类create方法，创建Land类对象）
4. 返回值：在交互地图对应坐标放上\*Object
5. 形参：

"Info": {

"CropName": "Potato",

"LiveDay": "1",

"MaturationDay": "5",

"Water": true,

"Fertilizer": 1

}

和MapLayer\* parent

virtual void init() = 0;

1. 作用：调用Land类的init（）函数，创建Sprite对象，假如有动作，还需要创建所有的动作序列。
2. 被调用时机：人物走进当前地图，前端需要sprite指针来渲染图片
3. 被调用方法：由director每帧对所有子节点自动调用
4. 实现：调用前台给出的接口（或自行加载）图片、精灵帧（plist和集合图等）

virtual void interact() = 0;

1. 作用：改变渲染；根据目前的人物状态（工具等级、所持物品类型），确定调用哪个状态改变函数，子类中的状态改变函数会改变图像状态把路径找出（写进当前状态json？把json当作类的数据成员），调用哪个函数渲染？、展示动作序列、平移等，并同时改变游戏的后台数据。
2. 被调用时机：交互时
3. 返回值：返回一个结构体，占据的坐标和图片，精灵指针

virtual void settle() = 0;

1. 被调用时机：游戏内一天结束时

virtual void clear() = 0;

1. 作用：delete对象
2. 被调用时机：关闭游戏，即存档结束

三、时间管理器

与ui logic/player对接

游戏中所有场景共用一个时间

时间管理器的实现，要求每天结束时自动调用所有物品的settle函数，时间流逝有自动计数，时间包括游戏天数和当前时间需要写进json文档以便下次打开游戏仍然是这个时间。

睡觉时机：由主角在任何时间与床交互触发；时间在游戏内两点时强制玩家睡觉（不实现）

睡觉功能：把当前数据保存至json，玩家体力值恢复为100%，时间变成第二天早上六点，随后睡醒

成员组成：

前端（渲染画面）

逻辑（更改Object对象，返回Sprite对象用于渲染，更改数据库）

数据库

前后端交互接口：工厂函数

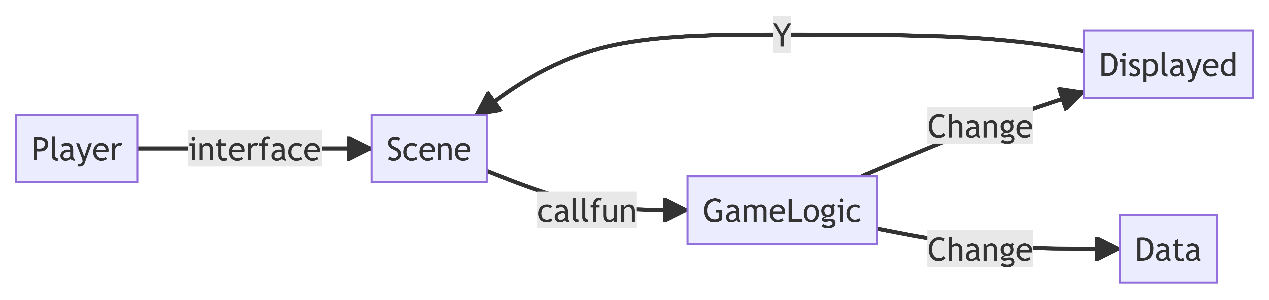
网格化地图：

因为是瓦片地图，所以可以将地图上的所有位置均除以网格大小（16pixel），然后每个格储存对应的网格信息，因而可以实现如下功能：

CollideMap：碰撞地图，通过位图形式储存，记录所有有碰撞体积的网格

Interfacemap：交互地图，储存一个地图上物品的抽象类的指针，用户与地图交互时，自动调用相应对象的interface函数，做出图像以及后台数据的改变。

程序调用方向：



TiledMap格式：