SceneManager 类是一个单例类，用于管理游戏场景的切换、地图的加载和 UI 层的显示。

**一、数据成员**

1. **static SceneManager\* instance\_**：

 **作用**：单例模式的实例指针。

 **调用时机**：在 getInstance() 方法中初始化并返回。

 **调用方法**：通过 SceneManager::getInstance() 获取。

2. **std::string current\_map\_name\_**：

 **作用**：存储当前地图的名称。

 **调用时机**：在地图切换时更新。

 **调用方法**：内部使用，更新当前地图。

3. **cocos2d::Director\* director\_**：

 **作用**：指向 Cocos2d-x 的导演类，用于场景管理。

 **调用时机**：在构造函数中初始化。

 **调用方法**：内部使用，管理场景切换。

4.**std::unordered\_map<std::string, cocos2d::Layer\*> layers\_**：

 **作用**：存储 UI 层的映射。

 **调用时机**：在 UI 层创建和管理时使用。

 **调用方法**：内部使用，管理 UI 层。

5.**std::unordered\_map<std::string, MapLayer\*> map\_**：

 **作用**：存储地图层的映射。

 **调用时机**：在地图创建和管理时使用。

 **调用方法**：内部使用，管理地图层。

6.**cocos2d::Node\* permanent\_node\_**：

 **作用**：用于存储永久性 UI 节点。

 **调用时机**：在构造函数中创建。

 **调用方法**：内部使用，管理永久性 UI。

**二、函数成员**

**私有函数**

**SceneManager()**：

 **作用**：构造函数，初始化导演和永久节点。

 **调用时机**：在 instance\_ 初始化时调用。

 **调用方法**：私有构造函数，单例模式。

**~SceneManager()**：

 **作用**：析构函数，释放永久节点。

 **调用时机**：程序结束。

 **调用方法**：自动调用。



**void createMapWithDocument(rapidjson::Document\* doc)**：

 **作用**：根据文档创建地图。

 **调用时机**：在加载地图时调用。

 **调用方法**：地图初始化。

**void createMaps()**：

 **作用**：创建所有地图。

 **调用时机**：在游戏初始化或地图切换时调用。

**void clearMaps()**：

 **作用**：清除所有地图。

 **调用时机**：在地图切换或游戏重置时调用。

**公有函数**

**static SceneManager\* getInstance()**：

 **作用**：获取单例实例。

 **调用时机**：需要访问 SceneManager 时调用。

 **调用方法**：SceneManager::getInstance()。

**void settle()**：

 **作用**：在一天结束时结算所有地图对象。

 **调用时机**：在游戏一天结束时调用。被TimeManager调用

 **调用方法**：void Game::endOfDay() {

    SceneManager::getInstance()->settle();

    // 其他结束日逻辑

}

**void showUILayer(const std::string& UI\_name) const**：

 **作用**：显示指定的 UI 层。

 **调用时机**：需要显示特定 UI 时调用。

 **调用方法**：SceneManager::getInstance()->showUILayer("UI\_name")。

**void changeUILayer(const std::string& UI\_name) const**：

 **作用**：切换到指定的 UI 层。

 **调用时机**：需要切换 UI 时调用。

 **调用方法**：SceneManager::getInstance()->changeUILayer("UI\_name")。

**void hideUILayer() const**：

 **作用**：隐藏所有 UI 层。

 **调用时机**：需要隐藏 UI 时调用。

 **调用方法**：SceneManager::getInstance()->hideUILayer()。

**void NextMap(const std::string& map\_name, const std::string& pos = "default") const**：

 **作用**：切换到下一个地图。

 **调用时机**：需要切换地图时调用。

 **调用方法**：SceneManager::getInstance()->NextMap("map\_name", "pos")。

**NextMapCallBack 类**切换为下一个地图的回调函数

**1. 构造函数**

NextMapCallBack(std::string map\_name, std::string pos);

 **作用**：构造函数用于初始化 NextMapCallBack 对象，设置要切换到的地图名称和位置。

 **调用时机**：切换地图时创建 NextMapCallBack 对象，

**2. 操作符重载**

void operator()();

 **作用**：重载 () 操作符，使得 NextMapCallBack 对象可以像函数一样被调用。这个方法执行地图切换的主要逻辑。

**3. 开始加载新地图**

void start();

 **作用**：开始加载新地图的逻辑，包括初始化加载进度条。

 **调用时机**：用户触发地图切换后立即执行。

**4. 创建新地图**

void create();

 **作用**：创建新地图的逻辑，包括从文件加载地图数据、初始化地图对象等。

 **调用时机**：在地图切换过程中调用，通常在加载新地图的资源后执行。

**5. 渲染新地图**

void render();

 **作用**：渲染新地图的逻辑，将地图对象添加到场景中并进行显示。

 **调用时机**：在地图切换过程中调用，通常在地图创建完成后执行。

**6. 组装新地图**

void assemble();

 **作用**：组装新地图的逻辑，包括设置地图的初始状态、调整摄像机位置等。

 **调用时机**：在地图切换结束时调用，通常在所有地图资源加载和渲染完成后执行。

 **调用方法**：地图切换流程的最后一步。