初级程序员：资源管理模拟

项目简介

应用理念

概述

在本项目中，您将实现资源管理模拟应用程序的功能。资源管理是许多类型游戏的核心，但模拟也常用于教育环境（例如，探索可持续性和经济学）和工业中。

这种类型的项目特别适合帮助您探索编程系统和体系结构，因为它们通常包括：

- 用户交互，使他们能够影响模拟
- 场景之间的过渡，以便用户进一步自定义
- 专为扩展而设计的系统，以增加仿真的复杂性

参考示例

您可能希望了解的资源管理应用程序的一些示例包括：

- 鱼洲：用于教育用途的多人可再生资源管理模拟帝国时代或文明：需要资源管理的经典策略游戏

Junior Programmer: Resource management simulation

您的任务清单

在详细介绍简报之前，以下是您将在此项目中执行的作的高级清单：

场景管理

在两个场景之间创建过渡
配置按钮，以便用户可以控制这些过渡
配置一个按钮以退出应用程序（或在 Unity 编辑器中退出播放模式）

数据持久化

在一个场景中配置按钮，以将所选颜色应用于第二个场景中的对象
保存用户选择的最后颜色，并在下次启动应用程序时预先选择

继承和多态性

在模拟中创建新类型的对象，其行为变体派生自基类

抽象化

☐ 重构以减少重复代码并提高可重用性

封装

使用 getter 和 setter 保护数据免遭滥用

优化代码

分析示例代码以识别基本优化问题

Junior Programmer: Resource management simulation

Unity 项目概述

重要提示：当您首次打开项目时，仓库模拟将具有基本功能，但应用程序将不起作用！

应用

项目中的场景

Unity 项目中的应用程序有两个场景：

1. 开始菜单（Menu），用户可以在其中启动模拟并关闭应用程序
2. 模拟场景 （Main），其样式为仓库

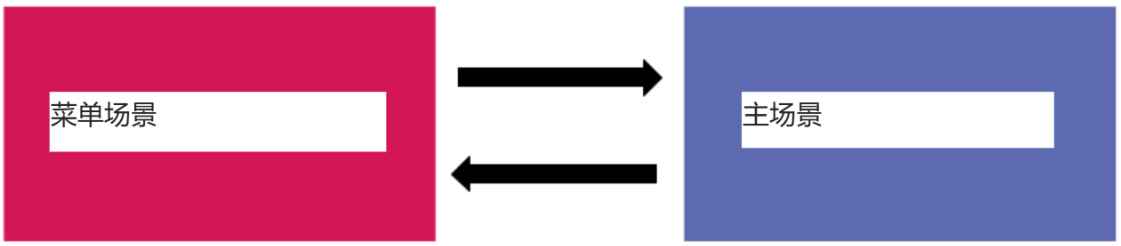
注意：还有一个名为 Optimization 的附加场景 — 这是一个优化案例研究，您将使用它来分析代码和识别问题。

用户交互

您将实现用户控制的过渡：

- 在两个场景之间
- 要退出应用程序，请仅从 Menu 场景

安查普尔



|Extaplen

用户交互

用户需要能够：

- 从开始菜单启动模拟场景
- 从模拟场景返回开始菜单
- 退出应用程序（或退出 Play 模式，用于编辑器内测试）
- 选择要应用于模拟中的 Transporter Units （forklift） 的颜色

用户上次选择的颜色也应在下次启动应用程序时预先选择。

Junior Programmer: Resource management simulation

仓库模拟

基本模拟

我们创建了一个基本模拟（在 Main 场景中），其中包含：

- 两种不同类型的资源堆（托盘上的物体），它们都以每秒 0.5 （/s） 的速度生产资源物品

- 两种机组：
  - 一个单元（工作程序），可以移动但没有其他功能
  - 运输单元（叉车），用户可以设置该单元将资源项从资源运输到预定义的基地（卡车前面的空间，用红色圆圈标记）

基本仿真功能

在基本模拟中，用户可以：

- 使用箭头键（或 WASD）在仓库空间内移动摄像头。
- 左键单击任何已定义的对象以将其选中，然后打开包含其详细信息的 UI 叠加层
- 选择单位后，右键单击：
  - 移动到哪一地方
  - 资源堆，用于将其用于将资源物品从对象运送到 Base

其他模拟要求

在从事此项目时，您将：

- 创建一个新的生产力单位，增加基地中的资源物品产量，并将其应用于仓库中的工人

- 应用面向对象编程的原则，改进和优化我们为您创建的代码

项目样式

我们将这个项目的主题围绕着一个仓库，采用轻质的低多边形风格。如果要在使用此功能时自定义项目主题，可以导入自己的资源并将其换出。

Junior Programmer: Resource management simulation

提供的脚本

该项目附带 7 个主要脚本，这些脚本已部分或全部编写。子文件夹中还有一些其他脚本可供您查看，但此处未包含这些脚本。

Base.cs

这是 Building 类的子类。它不会更改该类的行为，但它确实在其 Awake 函数中存储了对该类的单例引用。这意味着您可以从任何位置查询 base。当 Units 返回 base 时，将使用 Units。

Building.cs

这是两个子类的抽象基类：ResourcePile 和 Base。它保留类成员所包含内容的清单，还支持在该清单中添加和删除项。它还实现了 IUIInfoContent，这是模拟中的基本 UI 覆盖层。

ColorHandler.cs

此脚本创建颜色按钮，以便用户可以从开始菜单中选择要应用于 Transporter Units 的颜色。

ResourcePile.cs

这是 Building 类的子类。它采用对它将生成的 ResourceItem 的引用，并将有关它的以下信息存储在 Serializable 类中：

- 名字
- 唯一 ID
- 关联的 Sprite

它定义了一个 Update 函数，并在每次更新时按生产速度 / 秒递增生产计数器。一旦该值高于 1，它就会创建新的资源并递减计数器。

注： 它还覆盖了 Building 的 IUIInfoContent 的 GetData 函数，以将其生产速度作为数据提供给 InfoPanel。

TransporterUnit.cs

这是 Unit 的子类。它包含：

- 运输机单位当前设置的目标（资源堆）
- 一次可以携带的资源量
- 运输机单元当前正在运输的物品

TransporterUnit 子类覆盖其基类的三个函数：

- 它在范围内的建筑物周围的表现
- 两个 GoTo 函数，用于在 Transporter Unit 的目标发生变化时做出反应

Junior Programmer: Resource management simulation

Unit.cs

这是所有 Units 的抽象基类。它控制单元：

- 移动，包括它要移动到的目标点（可以是建筑物或位置）
- 颜色（如果已配置）
- 在其目标建筑范围内的行为

UserControl.cs

此类处理用户输入。在读取垂直和水平值时，它会在 xz 平面上移动摄像机，还可以控制左键和右键功能。