**任务：**

构建一个回合制走格子的“Roguelike”类型小游戏。

* 玩家每次可以移动一个格子，同时消耗一个食物点；
* 食物点可以通过拾取地图中的食物获得；
* 敌人会攻击玩家并偷取食物点。

**游戏组件：**

**渲染层 SpriteRenderer.SortingLayers**

游戏中渲染顺序依次为Floor，Items，Units，渲染层低的物体绘制较晚，所以会遮挡渲染层高的物体。

* Floor放置背景(Floor、OuterWall)；
* Item放置出口、障碍食物和苏打水(Exit、Wall、Food\Drink)；
* Units层放置玩家和敌人(Player、Enemy)；

**帧动画 Animation**

帧动画控制器包括Player和两个Enemy。

Player的序列帧动画包含闲置、攻击、被攻击 (Idle，Chop，Hit)三个状态。

* 当Trigger.playerChop触发时，Idle立即转为Chop，1s后Chop自动转为Idle；
* 当Trigger.playerHit触发时，Idle立即转为Hit，1s后Hit自动转为Idle。

Enemy的序列帧动画包含闲置、攻击 (Idle，Attack)两个状态。

* 当Trigger.enemyAttack触发时，Idle立即转为Attack，1s后Attack自动转为Idle；
* 不同Enemy通过Override Controller共用状态机相同的帧动画控制器。

**碰撞检测逻辑**

设置碰撞逻辑层BlockingLayer，Player、Enemy、Wall的所有碰撞检测(BoxCollider2D)都将在这个层里进行。

* Player和Enemy均为运动刚体(Kinematic Rigidbody2D)，可通过物理系统移动，但不受环境力影响；
* Player的碰撞器尺寸为0.9，防止角色移动时与周围发生不必要碰撞。
* Exit、Food(Drink)为碰撞触发器(trigger)，不会产生碰撞物理效果。

**界面 UI**

界面包括LevelImage和FoodText

LevelImage 在进入游戏时显示当前关卡数，然后隐藏；在游戏结束时显示成绩。

FoodText 显示游戏进程中的食物量

**音效**

Audio Source 包括背景音乐和音效。

背景音乐循环播放。

音效包含玩家移动音效、玩家攻击音效、玩家受伤音效、拾取食物音效，通过SoundManager类进行控制。

Code Tips:

SoundManager

* 播放指定音效 PlaySingle(AudioClip clip)
* 播放随机音效 RandomizeSfx (params AudioClip[] clips)

**体系结构：**

**关卡和整体结构**

**关卡管理类 board manager**

这个脚本会基于当前关卡的编号，自动随机生成每个关卡。

Code Tips:

对象串行化 [Serializable]

prefab实例化 Instantiate(original, position, rotation)

**游戏管理类 game manager**

加载关卡，管理分数。

Code Tips:

单例模式 Singleton

* 类似全局变量，使game manager不会在关卡开始时重置
* Loader脚本组织Singleton，把loader脚本放在maincamera上，awake时建立singleton
* 初始化进程 Awake()
* DontDestroyOnLoad(gameObject)

加载board manager类，随机生成关卡

* 获取组件 GetComponent<>()
* 通过 lock inspector对数组批量赋值

Enemy管理

* 使用List获取Enemy位置
* 清空上一关敌人，在InitGame()中清空List
* 协程MoveEnemies() 让敌人按一定顺序移动

**Scene被装载后回调**

* OnLevelWasLoaded(int level)已停用，用OnSceneLoad(Scene scene, LoadSceneMode sceneMode)代替
* 在Awake()中加入SceneManager.sceneLoaded += OnSceneLoad;
* 删除Awake()中的InitGame()，InitGame()仅在OnSceneLoad()中使用（？？？）

**游戏逻辑**

**moving object**

实现player和enemy的移动功能。

Code Tips:

实现平滑移动

* 协程 StartCoroutine
* 计算平方 sqrMagnitude 比 magnitude 更有效率
* float.Epsilon 替代 0
* Vector3 把一个点直线移动到目标点 Vector3.MoveTowards(current, target, maxDistanceDelta)
* RigidBody2D移动到指定位置 RigidBody2D.MovePosition(position)

尝试移动，泛型 T

* 泛型方法定义了尝试移动的过程，由子类来决定自己关心的地形物体和处理方法。
* 关键字 abstract virtual override out
* RaycastHit2D Physics2D.Linecast 光线投射碰撞检测 （使用时禁用自身碰撞器，防止射线碰撞到自身）

**player**

player交互类，继承自**moving object**

Code Tips:

变量不在Inspector中显示 [HideInInspector]

在Update()中进行移动

* 调用AttemptMove(T)，指定泛型T为期望交互组件Wall
* OnCantMove<Wall> Chop动画触发，Wall实现可破坏效果（Wall.DamageWall(loss)）

回调trigger碰撞检测，进行对象交互 OnTriggerEnter2D (Collider2D other)

* 到达Exit处时加载延时下一关Invoke(methodName, timeDelay);
* **重新加载场景**  
  Application.LoadLevel(Application.loadedLevel)已过时，  
  用SceneManager.LoadSceneAsync(0);替代
* 对象禁用(隐藏?) enabled = false，  
  当player对象被禁用时（OnDisable()），利用Singleton保存数据

**enemy**

enemy交互类，继承自**moving object**

Code Tips:

初始化时注册GameManager的List，调用 GameManager.instance.AddEnemyToList(this)

敌人移动，由GameManager调用MoveEnemy()

* 调用AttemptMove(T)，指定泛型T为期望交互组件Player
* OnCantMove<Player>Attack动画触发，Player受伤并触发Hit动画  
  （Player.LoseFood(playerDamage)）
* 每隔一回合移动一次
* 通过player和enemy当前位置判断移动方向