DuckManager

说明：鸭子管理器主要是用来创建游戏场景中的鸭子，不涉及鸭子的移动等相关操作

重要属性：

```
public static DuckManager Instance;                                          //单例模式实例
public GameObject duckPrefab;                                                // 鸭子预制体
private Stack<DuckController> duckPool = new Stack<DuckController>();       // 鸭子对象池
private List<DuckController> currentDuckList = new List<DuckController>(); // 当前鸭子列表
private float currentTime = 0;                                               // 当前时间，用来控制鸭子生成曲线
```

Awake

1. 设置单例Instance

获取一个鸭子实例

1. 如果当前鸭子对象池中没有对象，则实例化一个duckPrefab，并使用GetComponent得到DuckController，将实例化出的对象放入对象池
2. 从对象池Stack顶端拿出鸭子对象并设置SetActive(true)

创建一个菜单鸭子

1. <***获取一个鸭子实例***>
2. 从配置管理器中获取一个随机鸭子配置
3. 调用DuckController<***初始化菜单鸭子***>
4. 将菜单鸭加入到当前鸭子列表中

创建一个游戏鸭子

1. <***获取一个鸭子实例***>

2. 从配置管理器中<***随机获取一个鸭子的生成信息***>，包含随机获取鸭子配置、位置、layer等
3. 调用DuckController<***初始化游戏鸭子***>
4. 将游戏鸭加入到当前鸭子列表中

回收鸭子资源

1. 传入DuckController，将gameObject.SetActive(false)
2. 将传入DuckController放入资源池并从当前鸭子列表中移除

清空所有鸭子

1. 遍历当前鸭子列表<***回收鸭子资源***>

进入游戏逻辑

1. 启动协程<***当场上鸭子数量小于指定数量时创建一个游戏鸭子***>
2. 启动协程<***间隔固定时间后创建一个游戏鸭子***>

当场上鸭子数量小于指定数量时创建一个游戏鸭子

1. 查看当前鸭子列表数量是否小于阈值，当小于阈值时<***创建一个游戏鸭子***>

间隔固定时间后创建一个游戏鸭子

1. 设置一个固定时间WaitForSeconds
2. 当前鸭子列表数量小于阈值时根据随机数去判断是否需要<***创建一个游戏鸭子***>

停止游戏

1. 停止所有协程，主要是生成鸭子的逻辑
2. <***清空所有鸭子***>并回收资源