实验二报告

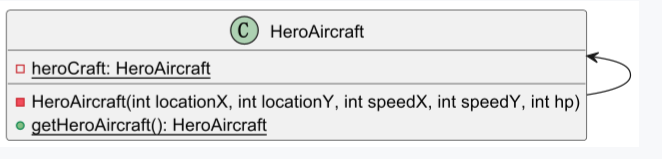
# 单例模式

1. 应用场景分析

飞机大战中，一局的英雄机只能有一个，但在目前的代码中，创建英雄机并没有限制，可能存在英雄机被过多创建的可能。

在英雄机的创建过程中使用单例模式能够保证英雄机类只有一个实例对象，即全局游戏中只有一个英雄机。

1. 解决方案



HeroAircraft 类应用单例模式，主要体现在它拥有私有属性 heroCraft，私有构造函数 HeroAircraft 和 公开的 getHeroAircraft 方法 。

调用者只有通过 getHeroAircraft 方法获得该类的实例对象，调用 getHeroAircraft 方法时，该方法会在 heroCraft不为 null 时调用 HeroAircraft 的构造函数创建实例对象赋值给 heroCraft 属性并将其返回给调用者，保证了该类的仅有一个实例对象。

# 工厂模式

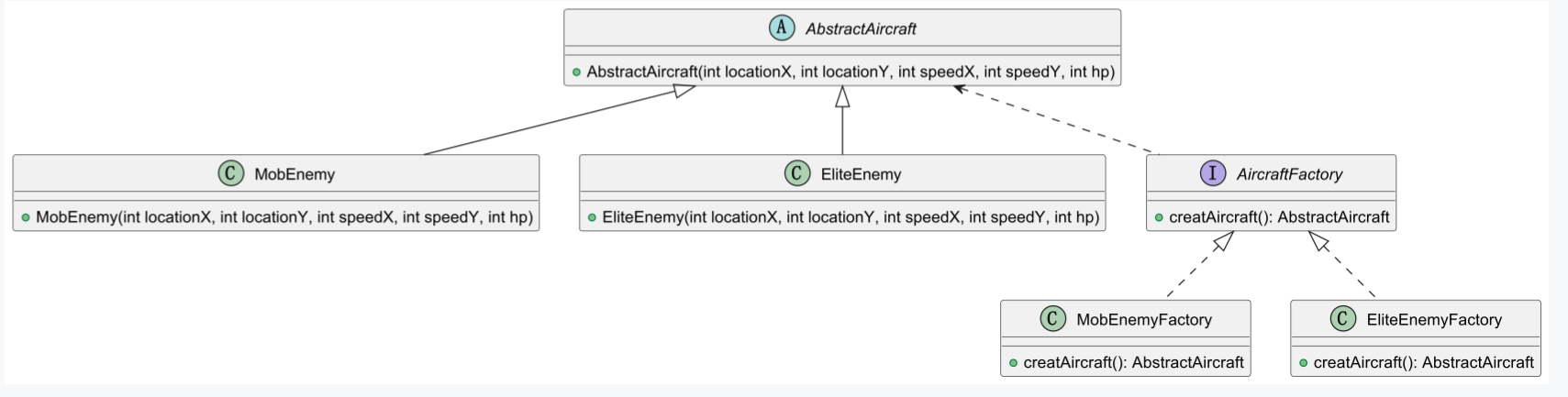
1. 应用场景分析

飞机大战的敌机和道具创建过程都是在 Game 模块中直接调用各个类的构造函数并传递较多的参数来创建的，破坏了类的封装。

使用工厂模式创建敌机和道具，可以将敌机和道具的创建和使用解耦，提高系统的灵活性和可维护性。

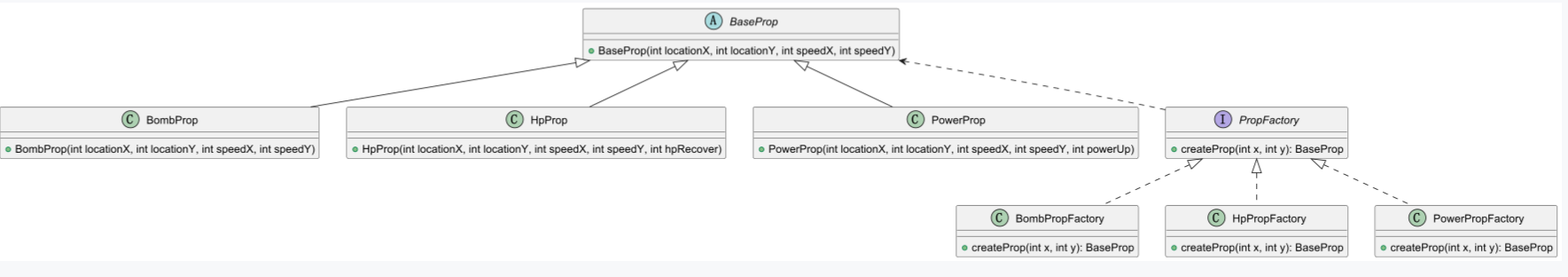
1. 解决方案

敌机的工厂模式：



AbstractAircraft、MobEnemy、EliteEnemy 为产品类，拥有各自的构造函数，AircraftFactory、 MobEnemyFactory、EliteEnemyFactory 为工厂类（接口），拥有 creatAircraft 方法，用于创建各种具体的敌机。

道具的工厂模式：



BaseProp、PowerProp、HpProp、BombProp 为产品类，拥有各自的构造函数，PropFactory、PowerPropFactory、HpPropFactory、BombPropFactory 为工厂类（接口），拥有 creatProp 方法，用于创建各种具体的道具。