魔兽世界4文档

xxxxxxxxxx xxx

# 说明

在对应的文件夹中有类图classdiagram.svg，使用浏览器打开即可查看。

# 需求变化分析

增加了一种武器，magic，编号为3，只能使用一次，和bomb一样。只不过增加了能吸血的效果。

武士战斗胜利或战平后攻击力增加。

# 需求实现

由于需求是逐步变更的，这里主要说明针对变化的部分的做法，关于魔兽世界3的部分不再赘述。

## 针对武士建造者的改进

在基类中增加一个成员变量warrior，同时将原来的buildWarrior方法改造成模板方法，在其中首先检查生命元是否足够，如果不足就直接返回null，够就调用constructWarrior方法（这个方法是抽象方法，由子类实现，会把构造好的武士用setWarrior方法设置到warrior中），等待子类将武士构造完成后再在buildWarrior方法中统一给武士做冠名、赋生命值、攻击力等操作。因为除了狼所有武士都有两个magic，所以只需要在constructWarrior调用完成返回后，在buildWarrior方法中查看this.warrior是不是Wolf，如果不是就使用武士的addWeapon方法增加两个magic武器即可（这个硬编码的问题在魔兽世界5中解决）。最后再返回this.warrior，这样既做到代码重用，又有解耦的效果。

相比于魔兽世界3，各个武士建造者子类只需要将原来的buildWarrior方法改名为constructWarrior，然后将return的操作改为setWarrior即可。由于外部依赖的是抽象的稳定的武士建造者基类，使用的是buildWarrior方法构造武士，而我们对buildWarrior的改动只影响到内部逻辑，接口签名没有改动，所以外部的代码也不需要改动。

## Magic

由于我们代码的良好解耦，我们只需要编写一个Magic类继承Weapon类，重写部分方法，并在Main方法最开头使用WeaponFactory的addWeapon方法添加到武器库中即可，其余关于武器的代码无需修改。

Magic类的实现也相当的简单，在构造方法中将这把武器的编号设置为3，然后名字起做“magic”。

在canDamage方法中永远返回真，原因同Bomb。

在use方法中，首先计算可以吸取多少血量，也就是给对方造成的伤害和对方生命值的最小值（由于之前没有考虑生命值变为负数时当作0处理的需求，所以这里要处理一下），之后就是按照正常的流程给自己加生命，给对方减生命，因为Magic是一次性消耗品，所以直接返回假，代表这把武器无法被再次使用。

## 战斗后攻击力变化

只需要在魔兽世界3的**战斗结算**部分中添加部分代码即可

如果双方均存活，双方攻击力均增加20%（使用武士类的getAttack和setAttack方法实现）

如果一方胜利，胜利方攻击力增加30%（使用武士类的getAttack和setAttack方法实现）

其余代码不用改动