Задание по программированию: Создание декоратора класса

Зачет · 1/1 баллов

Срок сдачи Сдайте это задание до 18 окт. г., 9:59 MSK

Инструкции Моя работа Обсуждения

Представьте себя ненадолго разработчиком компьютерной игры в стиле фэнтези. Вам поручено написать реализацию системы эффектов, которые могут быть наложены на героя вашей игры.

В игре есть герой, который обладает некоторым набором характеристик. Класс героя описан следующим образом:

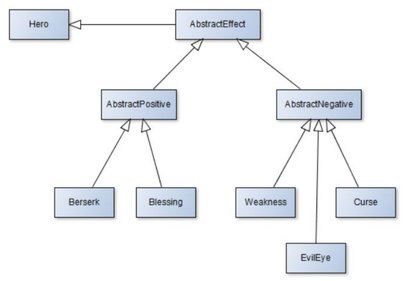
```
25          return self.stats.copy()
```

К основным характеристикам относятся: Сила (Strength), Восприятие (Perception), Выносливость (Endurance), Харизма (Charisma), Интеллект (Intelligence), Ловкость (Agility), Удача (Luck).

Враги и союзники могут накладывать на героя положительные и отрицательные эффекты. Эти эффекты изменяют характеристики героя, увеличивая или уменьшая значения определенных характеристик, в зависимости от того какие эффекты были наложены. На героя можно накладывать бесконечно много эффектов, действие одинаковых эффектов суммируется. Игрок должен знать, какие положительные и какие отрицательные эффекты на него были наложены и в каком порядке. Названия эффектов совпадают с названиями классов.

За получение данных о текущем состоянии героя отвечают методы get_stats, get_positive_effects, get_negative_effects.

Вам необходимо написать систему декораторов, представленную на UML-диаграмме:



Описание эффектов:

Берсерк (Berserk) -

- Увеличивает характеристики: Сила, Выносливость, Ловкость, Удача на 7;
- уменьшает характеристики: Восприятие, Харизма, Интеллект на 3;
- количество единиц здоровья увеличивается на 50.

Благословение (Blessing) -

- увеличивает все основные характеристики на 2.

Слабость (Weakness) -

- уменьшает характеристики: Сила, Выносливость, Ловкость на 4.

Сглаз (EvilEye) -

- уменьшает характеристику Удача на 10.

Проклятье (Curse) -

- уменьшает все основные характеристики на 2.

При выполнении задания необходимо учитывать, что:

- изначальные характеристики базового объекта не должны меняться.
- изменения характеристик и накладываемых эффектов (баффов/дебаффов) должны происходить динамически, то есть вычисляться при вызове методов get_stats, get_positive_effects, get_negative_effects
- абстрактные классы AbstractPositive, AbstractNegative и соответственно их потомки могут принимать любой параметр base при инициализации объекта (_ _ init _ _ (self, base))
- эффекты должны корректно сниматься, в том числе и из середины стека

Для тестирования правильности работы вашего решения вы можете повторить пример работы, приведенный ниже, или использовать тестовый скрипт. Заметим, что данные примеры не покрывают всех тестовых случаев, проверяемых тестовой системой, это всего лишь отправная точка для ваших экспериментов с кодом решения.

Важно! На проверку необходимо сдать фрагмент кода, содержащий только реализацию 8 классов: AbstractEffect, AbstractPositive, AbstractNegative, Berserk, Blessing, Weakness, Curse, EvilEye. Описывать класс Hero и импортировать его в коде НЕ НУЖНО (он уже реализован и будет импортирован тестовой системой).

Пример работы:

```
>>> from deco import *
>>> # создаем героя
>>> hero = Hero()
>>> hero.get_stats()
{'HP': 128, 'MP': 42, 'SP': 100, 'Strength': 15, 'Perception': 4, 'Endurance': 8, 'Charisma': 2, 'Intelligence': 3, 'Agility': 8, 'Luck': 1}
>>> hero.stats
{'HP': 128, 'MP': 42, 'SP': 100, 'Strength': 15, 'Perception': 4, 'Endurance': 8, 'Charisma': 2, 'Intelligence': 3, 'Agility': 8, 'Luck': 1}
>>> hero.get_negative_effects()
[ ]
>>> hero.get_positive_effects()
[ ]
>>> # накладываем эффект

>>> brs1 = Berserk(hero)
>>> brs1.get_stats()
{'HP': 178, 'MP': 42, 'SP': 100, 'Strength': 22, 'Perception': 1, 'Endurance': 15, 'Charisma': -1, 'Intelligence': 0, 'Agility': 15, 'Luck': 8}
>>> brs1.get_negative_effects()
[ ]
>>> brs1.get_positive_effects()
['Berserk']

>>> # накладываем эффекты
>>> brs2 = Berserk(brs1)

>>> cur1 = Curse(brs2)

>>> cur1.get_stats()
{'HP': 228, 'MP': 42, 'SP': 100, 'Strength': 27, 'Perception': -4, 'Endurance': 20, 'Charisma': -6, 'Intelligence': -5, 'Agility': 20, 'Luck': 13}
>>> cur1.get_positive_effects()
['Berserk', 'Berserk']
>>> cur1.get_negative_effects()
['Curse']
>>> # снимаем эффект Berserk
>>> cur1.base = brs1
>>> cur1.get_stats()
{'HP': 178, 'MP': 42, 'SP': 100, 'Strength': 20, 'Perception': -1, 'Endurance': 13, 'Charisma': -3, 'Intelligence': -2, 'Agility': 13, 'Luck': 6}
>>> cur1.get_positive_effects()
['Berserk']
>>> cur1.get_negative_effects()
['Curse']
```

Ссылка на список часто задаваемых вопросов по данной задаче - FAQ.

Как отправить

Когда работа будет готова, вы можете загрузить файлы для каждой части задания на вкладке 'Мои работы'.