Нека да напомним, че структурните шаблони се отнасят до това как класовете и обектите се подреждат (или съчетават) за образуване на по-големи структури.

Шаблон Декоратор

Идеята на този шаблон е да осигури възможност за динамично добавяне на нови полета и функционалност (поведение) към съществуващ обект, като запазва интерфейса му. Интересното е, че обекта няма да знае за своето „декориране“, което прави този шаблон особено популярен за системите които променят и развиват своята функционалност. Ключова точка за имплементирането на шаблона Декоратор е, че при него декораторите наследяват оригиналният клас и съдържат неговите инстанции. Декораторите предоставят гъвкава алтернатива на наследяването за разширяване на функционалността.

Пример:

Както подсказва името, шаблона декоратор взима съществуващия обект и добавя нещо към него. Като пример, може да си представим снимка която ще се показва на екрана. Има различни неща които могат да се добавят към нея, например рамка или определени тагове свързани със съдържанието. Комбинацията от оригиналната снимка и допълнителното съдържание за нея ще формира новия обект. Ако са добавени рамка и текст към снимката, може да се предположи че има два декоратора. Като се има предвид, че има много начини за декориране на снимка, то ние може да имаме много такива нови обекти.

Предимствата на декоратора са:

- Оригиналния обект остава непроменен;
- Няма само един клас натоварен с всички допълнителни функции и полета;
- Декорациите са независими;
- Декорациите могат да се използват заедно.

Структура

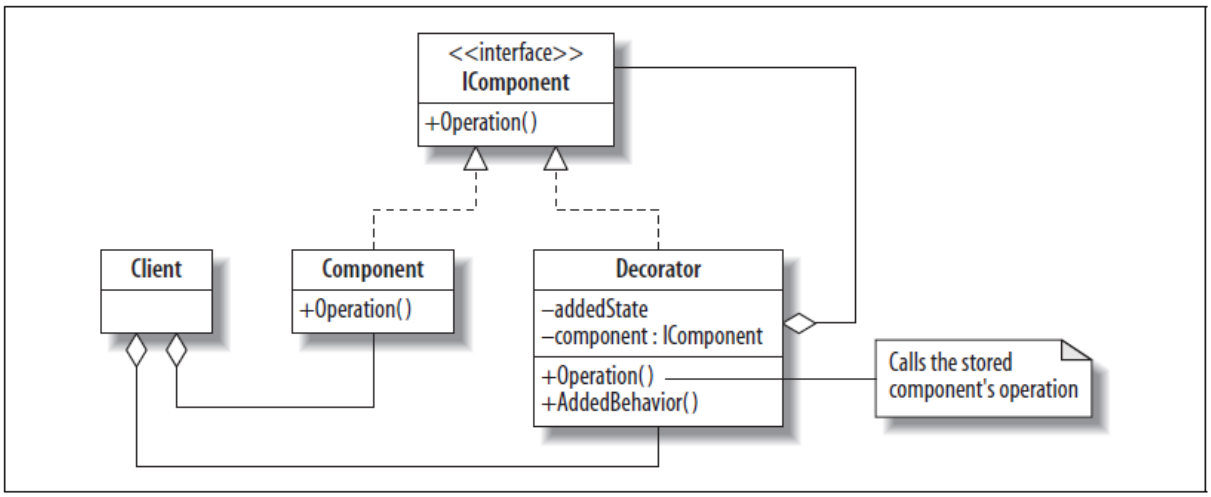
Ще представим основните елементи (главните герои) които участват в шаблона:

Component – оригиналният клас чийто обекти могат да имат добавени или променени операции (може да има повече от един такъв клас).

Operation –операцията за обектите от IComponent, която може да бъде заменена (съответно може да са няколко такива операции).

IComponent – интерфейсът който определя класовете на обектите , които могат да бъдат декорирани (например Component е един от тях).

Decorator – класът който е съвместим с интерфейса IComponent и добавя допълнителните полета и/или функционалност (може да са няколко такива класа).



В центъра на горната диаграма се вижда класа Decorator. Той има два вида връзки с интерфейса IComponent:

- Първата връзка е тази с пунктираната линия от Decorator до IComponent и показва, че Decorator реализира дадения интерфейс. Наследяването (т.е. имплементиране) на IComponent от Decorator означава, че обектите на Decorator могат да бъдат използвани навсякъде където се очакват обекти от IComponent. Класът Component също е в подобна връзка с IComponent и затова клиента може да използва обектите от Component или от Decorator, т.е. те са взаимнозаменяеми – същността на декоратор.
- Другата връзка е агрегация и свързва Decorator с IComponent. Това показва, че Decorator инстанцира един или повече IComponent обекта и това че декорираните обекти могат да заменят оригиналните. Decorator използва като атрибут типа IComponent за да се обърне към някоя от заменените Operation, която е била припокрита. Това е начинът по който работи Decorator.

//В Decorator може да се включват други полета или методи (AddedBehavior и addedState).

Използване:

- Декоратор може успешно да се ползва в графиката, както е примера DecoratorPhoto. Но спокойно могат да се дадат примери с аудио и видео. Например видеото може да бъде с различна степен на компресия и да имаме няколко обекта (филми с различно качество). А за аудиото може филмите да вървят с техния звук, така и да е добавен и дублиран превод;
- В реализацията на I/O интерфейса на много езици е използван шаблона Декоратор. Например при C# може да разгледаме следната йерархия:
  System.IO.Stream
    System.IO.BufferedStream
    System.IO.FileStream
    System.IO.MemoryStream

System.Net.Sockets.NetworkStream
System.Security.Cryptography.CryptoStream

Подкласовете декорират Stream, защото те го наследяват и ще съдържат инстанция на Stream когато те са инстанцирани. Много от техните свойства се отнасят до тази инстанция.

- В днешния свят мобилните устройства, уеб браузърите и мобилните приложения също използват шаблона декоратор. За да могат да се виждат на устройства с различни екрани те използват scroll bars, да изключват банери. Като цяло този шаблон е приет и за настолните браузъри.