Wzorzec projektowy - Łańcuch zobowiązań

Łańcuch zobowiązań to behawioralny wzorzec projektowy. Implementujemy go w celu umożliwienia przekazywania żądań wzdłuż lancucha potencjalnych obiektów az zadanie zostanie obsluzone.

1 Opis problemu

Czasami w aplikacji mamy do czynienia z sytuacją, gdzie żądanie powinno być przetworzone przez jeden z wielu obiektów, ale nie chcemy, żeby nadawca żądania znał konkretny obiekt, który je obsłuży. Przykładem mogą być:

- systemy autoryzacji,

- system logowania

Warunki wstępne zastosowania:

-Chcemy uniknąć silnego powiązania między nadawcą żądania a jego odbiorcą.

-Możliwych obsługujących żądanie jest wielu, ale tylko jeden (lub kilku) powinien je przetworzyć.

-Chcemy, aby obiekty obsługujące żądanie były organizowane dynamicznie i tworzyły łańcuch.

2 Rozwiązanie

Poniżej można znaleźć strukturę z jakiej składa się ten wzorzec projektowy:

Handler – jest to interfejs definiujący metodę do przetwarzania żądania oraz referencję do następnego elementu w łańcuchu.

AbstractHandler - Podstawowa klasa implementujące zachowanie handlera

ConcreteHandler - Implementuje logikę obsługi żądania. Jeśli nie jest w stanie przetworzyć żądania, przekazuje je dalej.

Client - Tworzy i konfiguruje łańcuch odpowiedzialności oraz przekazuje żądanie do pierwszego elementu.

3 Konsekwencje

Zalety:

-Zmniejsza zależności między nadawcą a odbiorcą żądania.

-Łatwe dodawanie nowych typów handlerów bez zmiany istniejącego kodu klienta.

-Handlerzy mogą być łatwo reorganizowani.

Wady:

- Trudno przewidzieć, który handler faktycznie obsłuży żądanie(debugowanie może być trudniejsze).

- Jeśli żaden handler nie obsłuży żądania – może zostać ono "zgubione".

- Może dojść do nieefektywności, jeśli łańcuch jest długi lub źle zaprojektowany.