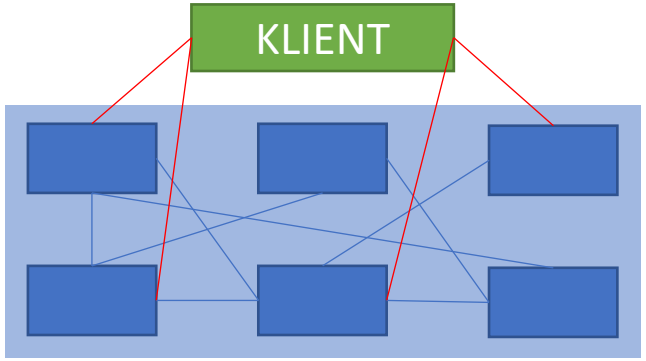
Wzorce projektowe - fasada

MarcinN

19.10.2020

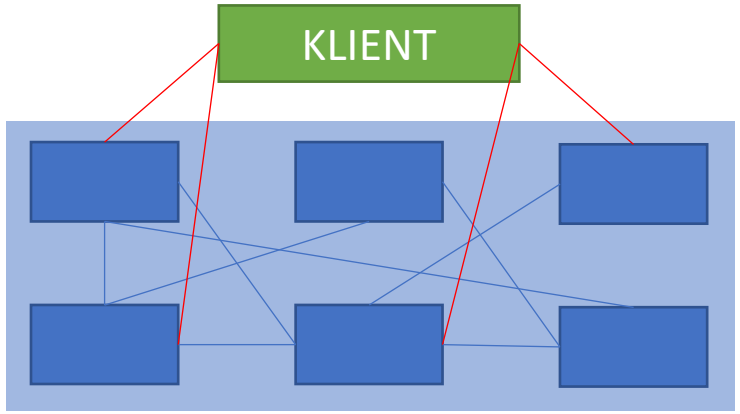
Kiedy występuje problem?

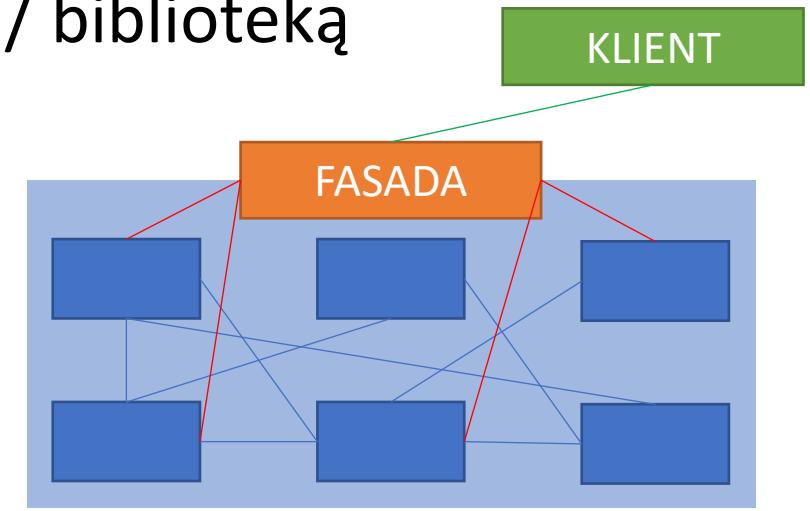


- Klient musi korzystać bezpośrednio ze skomplikowanej biblioteki/podsystemu
- Przykładowo – jeżeli klient ma do dyspozycji wielką bibliotekę do przetwarzania chmur punktów 3D umożliwiającą skomplikowane operacje transformacji danych oraz pomiary i obliczenia, a tak naprawdę chciałby tylko wyświetlić chmurę punktów na ekranie…
- Należałoby ułatwić życie klientowi i nie zmuszać go do korzystania z takiego wielkiego kombajnu, a jedynie dać mu to czego potrzebuje
- A może by tak użyć fasady…?

Czym jest fasada?

- Fasada – strukturalny wzorzec projektowy
- Dostarcza uproszczony interfejs do obsługi bardziej rozbudowanej biblioteki / zestawu klas / frameworku / podsystemu
- Dostarczony interfejs unifikuje mnogość interfejsów podsystemu
- Interfejs dostarczony przez fasadę jest pośrednikiem pomiędzy klientem, a obsługiwanym podsystemem / biblioteką





Co właściwie robi fasada?

- Zapewnia wygodny dostęp do określonego zakresu funkcjonalności podsystemu
- Hermetyzuje tę funkcjonalność i ukrywa ją przed resztą aplikacji
- Pośredniczy w przekazywaniu żądań klienta do właściwych miejsc w podsystemie
- Sama fasada jest niewidoczna z punktu widzenia podsystemu

Co zyskujemy z fasadą?

- Zwiększenie czytelności kodu i łatwość jego wykorzystania, dzięki uproszczonemu interfejsowi
- Zastąpienie zestawu kiepskich interfejsów, jednym dobrze zaprojektowanym
- Łatwość podmiany podsystemu na inny – z punktu widzenia aplikacji, która używa fasady, nic się nie zmieni
- Warstwowość aplikacji

- Ale! Trzeba uważać, żeby fasada nie stała się „god object”, który ma w sobie wszystko co się tylko da ;)

Źródła

- https://refactoring.guru/pl/design-patterns/facade
- https://refactoring.guru/pl/design-patterns/facade/cpp/example
- https://infotraining.bitbucket.io/cpp-dp/facade.html