Podpunkt 9 wzorce projektowe:

**Kompozyt:**

Jest strukturalnym wzorcem projektowym, który umożliwia komponowanie struktury drzewiastej z obiektywów i traktowanie jej jak pojedynczy obiekt.

Kompozyt stał się dość popularnym rozwiązaniem wielu problemów gdzie w grę wchodzi struktura drzewa. Zaletą tego wzorca jest możliwość uruchamiania metod rekurencyjnie na wszystkich elementach struktury i sumowanie wyników ich działania.

Wzorzec Kompozyt jest dość powszechny w kodzie. Często stosuje się go do modelowania hierarchii komponentów interfejsu użytkownika lub kodu który działa na grafach.

Zastosowanie:

Stosujemy wzorzec Kompozyt gdy musimy zaimplementować drzewiastą strukturę obiektów.

**Obserwator:**

Jest behawioralnym wzorcem projektowym pozwalającym obiektom powiadamiać inne obiekty o zmianach swojego stanu.

Obserwator daje możliwość subskrypcji lub zrezygnowania z subskrypcji zdarzeń dowolnego obiektu implementującego interfejs subskrybenta.

Zastosowanie:

Stosujemy wzorzec Obserwator gdy zmiany stanu jednego obiektu mogą wymagać zmiany w innych obiektach, a konkretny zestaw obiektów nie jest zawczasu znany lub ulega zmianom dynamicznie.

**Strategia:**

Jest to behawioralny wzorzec projektowy zakładający przekształcenie zestawu zachowań w obiekty, które można stosować zamiennie w pierwotnym obiekcie.

Wzorzec Strategia jest bardzo powszechny w kodzie C#. Często stosuje się go w różnych frameworkach by umożliwić użytkownikom zmianę funkcjonalności klasy bez konieczności rozszerzania jej.

Zastosowanie:

Stosuj wzorzec Strategia gdy chcesz używać różnych wariantów jednego algorytmu w obrębie obiektu i zyskać możliwość zmiany wyboru wariantu w trakcie działania programu.

Warto stosować ten wzorzec gdy masz w programie wiele podobnych klas, różniących się jedynie sposobem wykonywania jakichś zadań.

**Metoda wytwórcza:**

Jest kreacyjnym wzorcem projektowym rozwiązującym problem tworzenia obiektów-produktów bez określania ich konkretnych klas.

Wzorzec Metoda wytwórcza jest często stosowany w kodzie C#. Przydaje się gdy chcesz nadać swojemu kodowi wysoki poziom elastyczności. Metody wytwórcze można poznać po metodach kreacyjnych tworzących obiekty na podstawie konkretnych klas, ale zwracających typ abstrakcyjny lub interfejs.

Zastosowanie:

Stosujemy Metodę Wytwórczą gdy nie wiemy z góry jakie typy obiektów pojawią się w twoim programie i jakie będą między nimi zależności. Metoda Wytwórcza oddziela kod konstruujący produkty od kodu który faktycznie z tych produktów korzysta. Dlatego też łatwiej jest rozszerzać kod konstruujący produkty bez konieczności ingerencji w resztę kodu.

**Dekorator:**

Jest to strukturalny wzorzec pozwalający na dodawanie obiektom nowych obowiązków w sposób dynamiczny — poprzez “opakowywanie” ich w specjalne obiekty posiadające potrzebną funkcjonalność.

Dekorator można poznać po metodach kreacyjnych lub konstruktorach przyjmujących obiekty tej samej klasy lub interfejsu jako bieżącą klasę.

Zastosowanie:

Stosujemy wzorzec Dekorator gdy chcemy przypisywać dodatkowe obowiązki obiektom w trakcie działania programu, bez psucia kodu, który z tych obiektów korzysta.

Dekorator pozwala ustrukturyzować logikę biznesową w formie warstw, tworząc dekorator dla każdej warstwy i składać obiekty z różnymi kombinacjami tej logiki w czasie działania programu. Kod klienta może traktować wszystkie obiekty w taki sam sposób, ponieważ wszystkie są zgodne pod względem wspólnego interfejsu