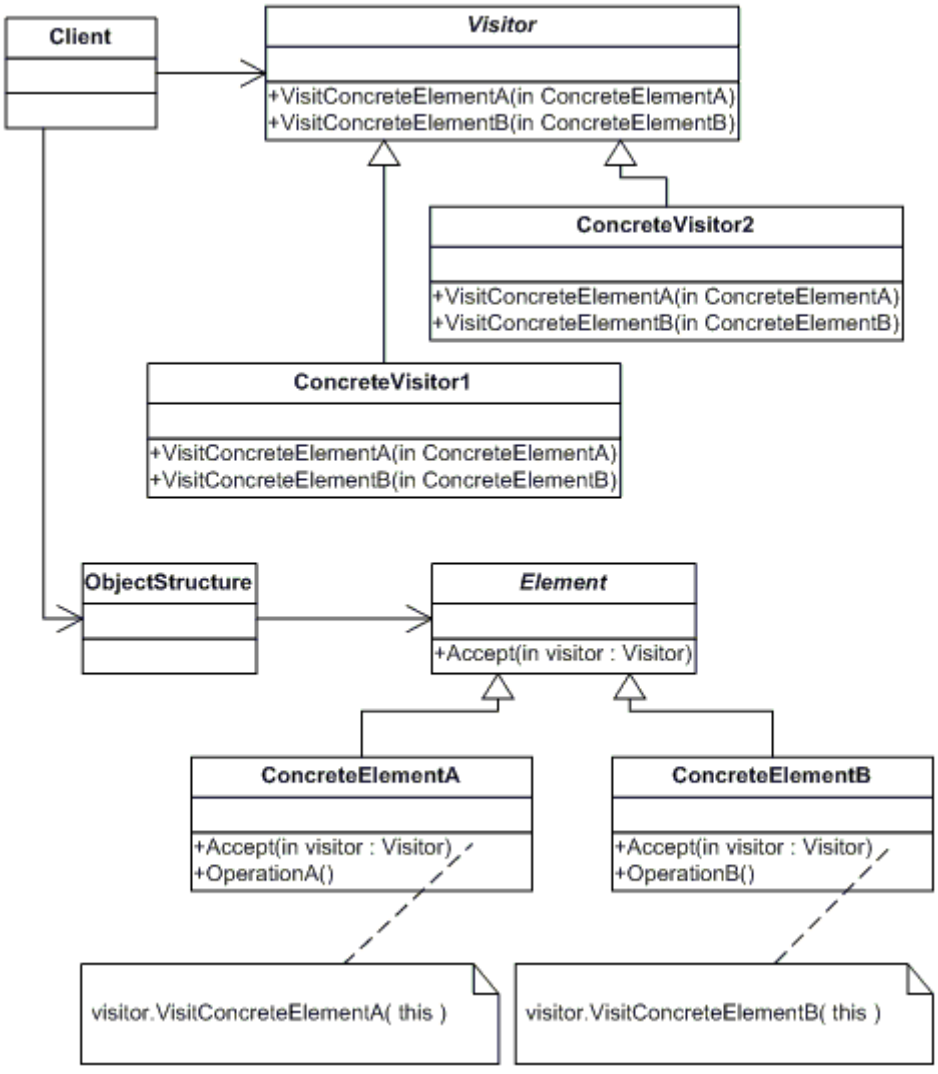
Projektowanie obiektowe oprogramowania
Wykład 6 – wzorce czynnościowe
Wiktor Zychla 2014

1 Od Composite structure do Visitor

Motto: definiowanie nowej operacji bez modyfikowania interfejsu
Kojarzyć: TreeVisitor, ExpressionVisitor z biblioteki standardowej .NET



Komentarz. Podstawowy problem tak skonstruowanej struktury to jej złożoność. Studenci, którzy już rozumieją na czym polega Visitor pytają często „po co jest cała część akceptowania struktury visitora w elementach struktury kompozytowej? Dlaczego nie wystarczy sama hierarchia visitorów?”

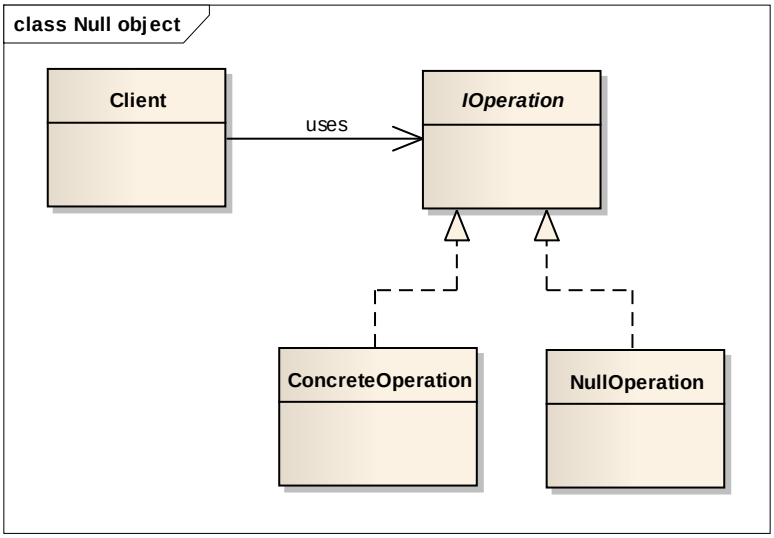
Odpowiedzi na to pytanie udzielimy na wykładzie – wszystko zależy od tego **kto** implementuje algorytm przechodzenia struktury kompozytowej.

Jeżeli robi to sam visitor – to wydaje się że nie ma w ogóle potrzeby aby istniała zależność od struktury kompozytowej do visitorów. Takie visitory są jednak bardziej złożone, choć jak pokazuje przykład z biblioteki standardowej, **ExpressionVisitor**, bywają wybierane przy implementacji.

Jeżeli robi to kompozyt – to visitory nie muszą znać się na przechodzeniu struktury kompozytowej i wtedy mamy taki podział odpowiedzialności między strukturą kompozytoową a visitorami jak na diagramie.

2 Null Object

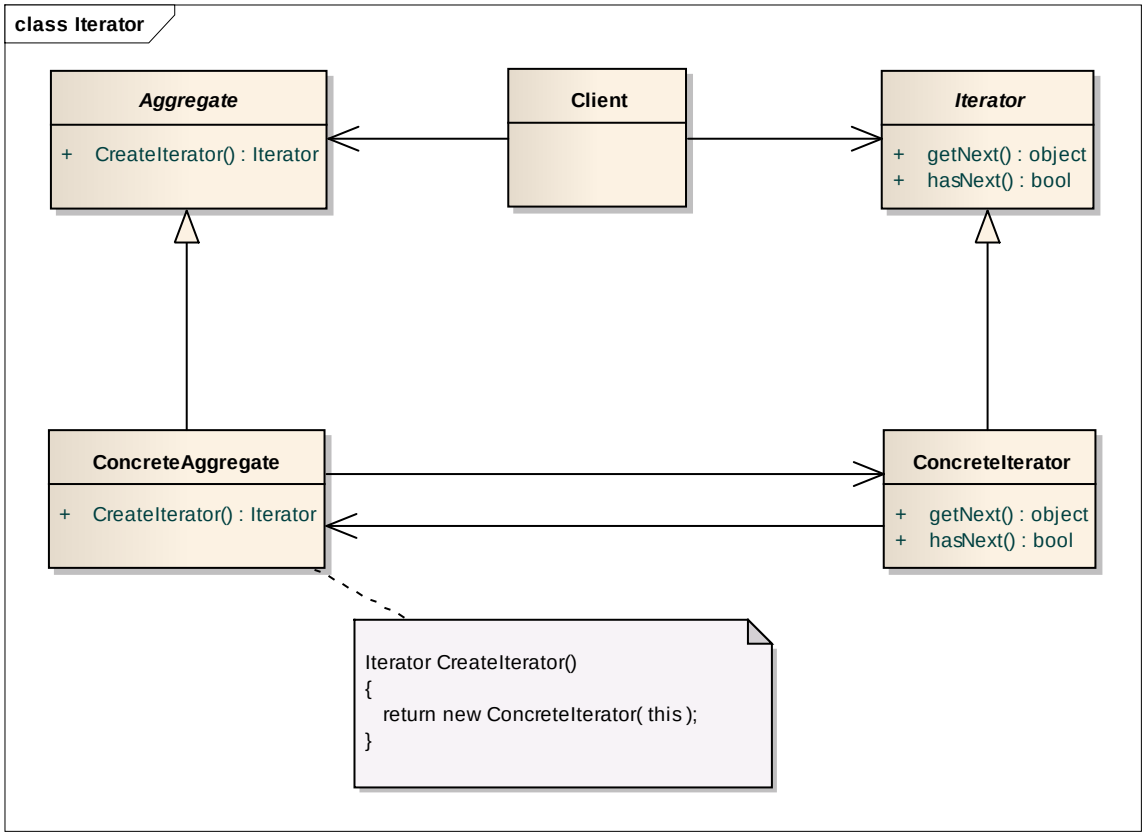
Motto: pusta implementacja zwalniająca klienta z testów **if** na *null*



Komentarz: Null object ma sens w połączeniu z fabryką – przy specyficznych lub niedostatecznych parametrach inicjalizacyjnych fabryka zwraca Null object.

3 Iterator

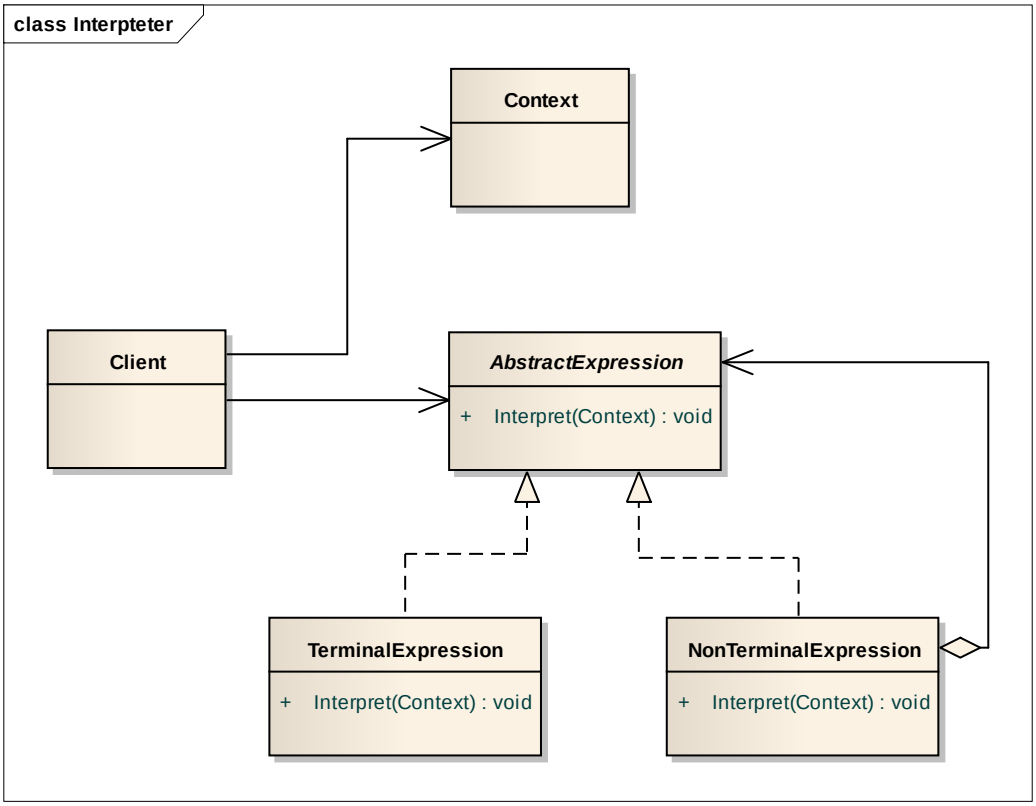
Motto: sekwencyjny dostęp do obiektu zagregowanego bez ujawniania jego struktury
Kojarzyć: *IEnumerator*, *IEnumerable*



Komentarz: ten wzorzec został z powodzeniem włączony do nowoczesnych języków programowania (Java, C#) stanowiąc podstawę dla lukru syntaktycznego (C# - **foreach**). To przykład jak wzorce projektowe silnie wpływają na języki.

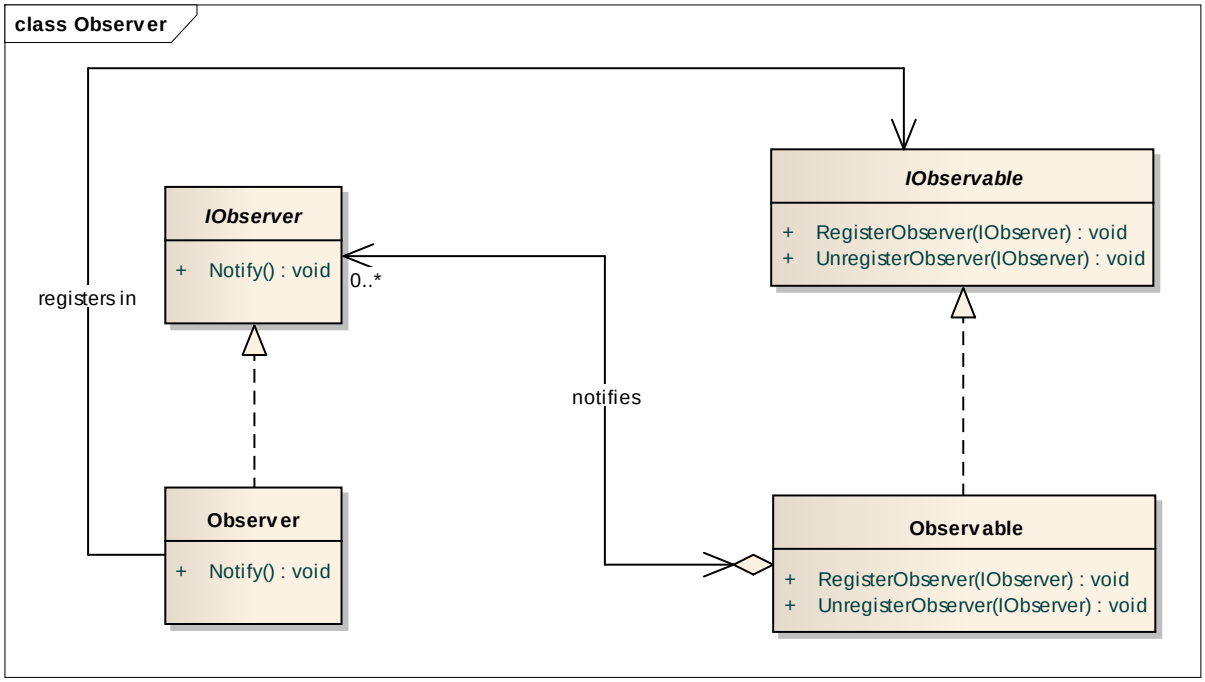
4 Interpreter (Little Language)

Motto: reprezentacja gramatyki języka i jego interpretera
Kojarzyć: kompozyt z interpreterem



5 Observer

Motto: powiadamianie zainteresowanych o zmianie stanu, dzięki czemu nie odwołują się one do siebie wprost.
Kojarzyć: zdarzenia w C#
Przykład z życia: architektura aplikacji oparta o powiadomienia między różnymi widokami (w środku okienka – Mediator, pomiędzy okienkami – Observer)
Jeszcze inaczej – Observer ujednolica interfejs „Colleagues” Mediatora, dzięki czemu obsługuje dowolną liczbę „Colleagues”



Komentarz: kolejny wzorzec który silnie wpływa na rozwój języków – C#-owe zdarzenia (events) to przykład uczynienia ze wzorca projektowego elementu języka.