Laboratorium 2: Adapter oraz dekorator w przykładach

PIOTR SZUSTER, MGR. INŻ.

1. WEKTORY

Rozważmy następujący problem. Na diagramie S1 przedstawione są klasy, odwzorowujące wektory przestrzeni dwuwymiarowej. Można zauważyć, że klasa Vector2D reprezentuje wektor

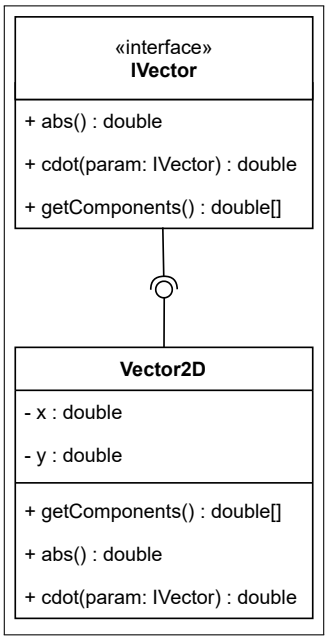

Fig. S1. Diagram UML wektorów

w kartezjańskim układzie współrzędnych. Interfejs IVector deklaruje metody:

abs() - oblicza moduł wektora
cdot() - oblicza iloczyn skalarny między dwoma wektorami (wywoływana na obiekcie, przyjmuje w parametrze r-wartość operatora binarnego $\cdot$
getComponents() - zwraca wartości składowych wektora

Należy nadmienić, że Vector2D jest wektorem wodzącym - jego początek znajduje się w punkcie $(0, 0)$ układu współrzędnych.

Wektor dwuwymiarowy można również przedstawić w innej postaci - pary: moduł oraz kąt między osią OX a kierunkiem wektora. Wektor tej postaci posiada współrzędne w układzie biegunowym. Powstaje pytanie w jakis sposób zmodyfikować istniejący diagram klas tak, by nie naruszyć zasad programowania obiektowego? Jednym ze sposobów, który wydaje się być oczywisty jest skorzystanie z dziedziczenia. Rysunek S2 przedstawia przykład rozwiązania problemu z wykorzystaniem dziedziczenia (część bladoczerwona). W istocie klasa 2DPolarInheritance dodaje nową metodę getAngle(), która za pomocą funkcji cyklometrycznych i definicji funkcji trygonometrycznych ma na celu obliczenie kąta między osią OX oraz kierunkiem wektora. Dziedziczy ona również komplet cech klasy Vector2D. Istotą dziedziczenia jest jednak jego wykonywanie w czasie kompilacji. Na etapie działania programu nie ma możliwości zmiany intefejsu już istniejących wektorów, co może stanowić istotne ograniczenie. Ograniczenie to można ominąć stosując rozwiązanie innego typu.

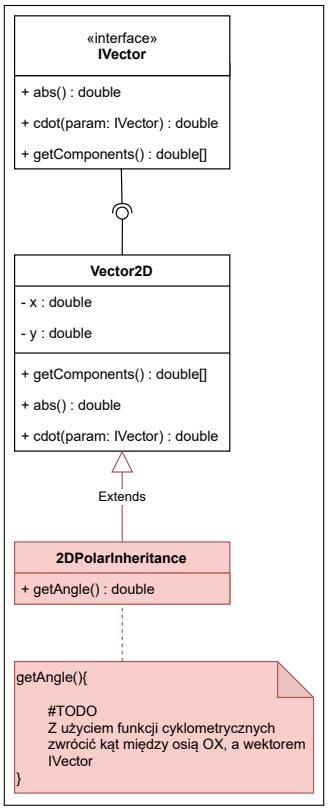

Fig. S2. Wykorzystanie dziedziczenia

2. ADAPTER

Adapter jest strukturalnym wzorcem projektowym. Ma na celu dostosowanie istniejącego intefejsu do potrzeb użytkownika. Aby to zrealizować adapter posiada interfejs, który jest kompatybilny z oczekiwaniami użytkownika. Obiekty i klasy zdefiniowane przez użytkownika wywołują metody uzgodnionego interfejsu adaptera. Adapter wykonuje stosowne operacje tak, aby dane i żądania podawane przez użytkownika przekształcić do postaci obsługiwanej przez obiekty adaptowane w czasie działania programu. Wykorzystanie wzorca projektowego Adapter równolegle z dziedziczeniem przedstawia rysunek S3. Adapter jest przedstawiony kolorem jasnozielonym.

W S3 Client chce skorzystać z interakcji z klasą Vector2D za pomocą interfejsu umożliwiającego użycie układu współrzędnych biegunowych (IPolar2D). Klasa Polar2DAdapter implementuje ten interfejs razem z interfejem IVector. Implementacja IVector na tym etapie pozwala wymiennie wykorzystywać Polar2DAdapter zamiast Vector2D. Polar2DAdapter posiada referencję do obiektu klasy Vector2D aby mieć możliwość wyłuskania niezbędnych pól. Proszę zauważyć, że Polar2DAdapter deleguje wykonanie metod z interfejsu IVector do obiektu klasy, którego adres jest przechowywany w referencji srcVector. Jedynie metoda getAngle() jest novum i ma posiadać implementację z użyciem funkcji trygonometrycznych i cyklometrycznych.

3. DEKORATOR

Załóżnmy kolejne działanie matematyczne, które powinniśmy zaimplementować celem rozwinięcia rozwiązania. Tym działaniem będzie iloczyn wektorowy $\times$. Przyjmuje on dwa argumenty w postaci wektorów, a jego wynikiem jest nowy wektor, który jest prostopadły do płaszczyzny wyznaczanej przez wektory argumentów. Prostopadłość wyniku wymusza dodanie trzeciego wymiaru co stanowi rozwinięcie rozwiązania o nowe możliwości. Również tym razem istnieją dwa typowe sposoby implementacji. Dziedziczenie oraz wykorzystanie strukturalnego wzorca projektowego: dekoratora. Na rysunku S4 Dekorator przedstawiony jest kolorem jasnoniebieskim,

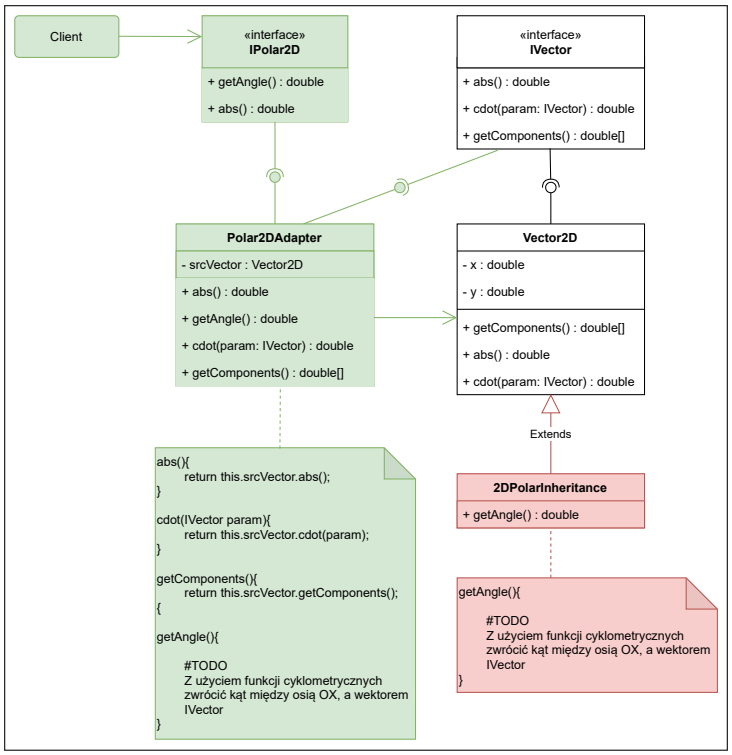

Fig. S3. Wykorzystanie adaptera

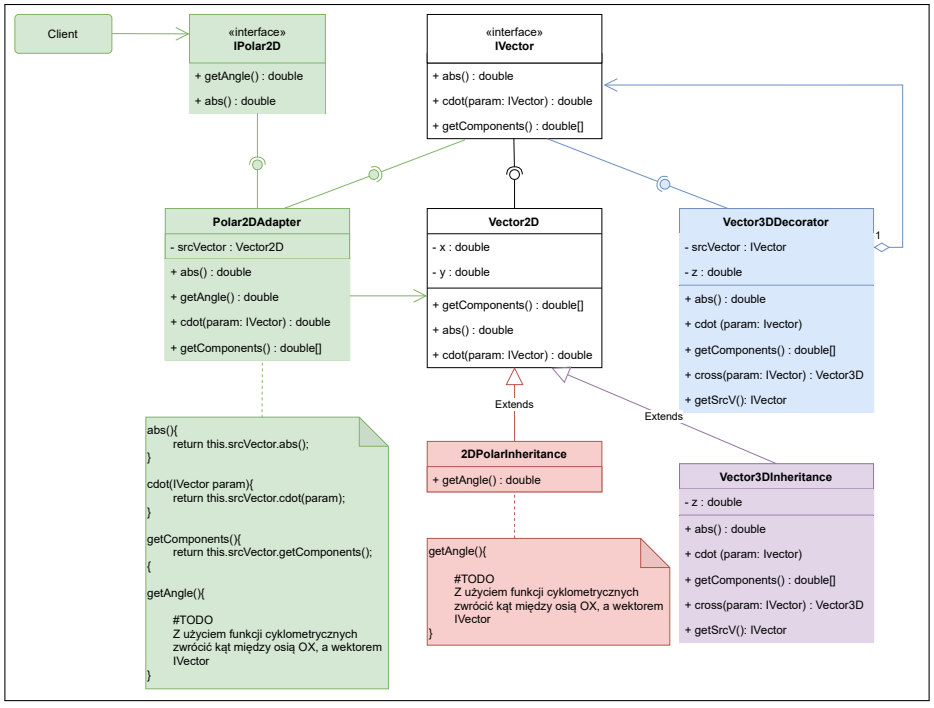

Fig. S4. Wykorzystanie dekoratora

a rozwiązanie z dziedziczeniem jasnofioletowym.

Vector3DDecorator zawiera referencję, która agreguje dowolny IVector w referencji srcVector. IVector został tutaj użyty celem zachowania ogólności. Pozwala to wykorzystać wektor dwuwymiarowy w dowolnej postaci. Metoda do obliczenia iloczynu wektorowego to cross. cross zawsze zwraca wektor trójwymiarowy. Metoda getSrcV() zwraca referencję do obiektu dekorowanego, a w przypadku dziedziczenia tworzy nowy obiekt.

4. ZADANIE LABORATORYJNE

Zaimplementować klasy przestawione na diagramie S4. W funkcji main() utworzyć trzy przykładowe wektory o wybranych współrzędnych. Dla każdego z nich wyświetlić na ekranie ich współrzędne w układach kartezjańskim oraz biegunowym. Wyświetlić wyniki iloczynu skalarnego oraz wektorowego (w formie współrzędnych kartezjańskich) dla wszystkich możliwych kombinacji.

UWAGI

Do realizacji iloczynu wektorowego użyć metody macierzowej
Wektor dwuwymiarowy $\Rightarrow z = 0$

A. Warunki zaliczenia zadania

Złożenie podczas zajęć laboratoryjnych projektu zawierającego

Implementację rozwiązania w dowolnym obiektowym języku programowania
Zaprezentowanie zaimplementowanego rozwiązania
Przedłożenie zwięzłego sprawozdania przedstawiającego wady i zalety użytych podejść

Wersja: 26 października 2021, 19:30