

Programowanie obiektowe

Wykład 11.

Marcin Młotkowski

17 maja 2024

Plan wykładu

- 1 Tworzenie oprogramowania
- 2 Unified Modeling Language
 - Diagramy klas
 - Przykład
- 3 Analiza obiektowa (Yourdon & Coad)
- 4 Przykład

Plan wykładu

- 1 Tworzenie oprogramowania
- 2 Unified Modeling Language
 - Diagramy klas
 - Przykład
- 3 Analiza obiektowa (Yourdon & Coad)
- 4 Przykład



```
import java.awt.*;

public class Aplikacja implements Runnable
{
    private int xPos;
    private int yPos;

    private String tempDir = "/tmp";

    public static void main(String[] args)
    {
        Aplikacja app = new Aplikacja();
        app.run();
    }
}
```

Przypomnienie (akwarystyka)

Specyfikacja programu

„Napisz mi program, w którym bym notował elektronicznie różne fakty związane z moim akwariem, na przykład liczbę i gatunki ryb, temperaturę, częstotliwość karmienia”.

Metody projektowania: prosta dziedzina

Przykłady: numeryczna analiza danych, przetwarzanie danych finansowych.

Dane wejściowe



Przypomnienie (programowanie funkcyjne)

Definicja silni

definicja matematyczna

$$\text{silnia}(n) = \begin{cases} 1 & \text{gd}y\ n = 0 \\ n * \text{silnia}(n - 1) & \text{wpp} \end{cases}$$

Implementacja w Ocaml'u

```
let rec silnia n =  
  if n=0 then 1  
  else n*silnia(n-1);;
```

Metody projektowania

Dekompozycja funkcjonalna

podział modułów ze względu na funkcje

Graf przepływu danych

Ustalenie kolejności wykonywania operacji na danych

Rozbudowana dziedzina

- Modelowanie danych;
- modelowanie związków między danymi.

Przypomnienie

Specyfikacja programu

„Napisz mi program, w którym bym notował elektronicznie różne fakty związane z moim akwariem, na przykład liczbę i gatunki ryb, temperaturę, częstotliwość karmienia”.

Przypomnienie

Specyfikacja programu

„Napisz mi program, w którym bym notował elektronicznie różne fakty związane z moim akwarium, na przykład liczbę i gatunki ryb, temperaturę, częstotliwość karmienia”.

Przypomnienie

Specyfikacja programu

„Napisz mi program, w którym bym **notował** elektronicznie różne **fakty** związane z moim **akwarium**, na przykład liczbę i **gatunki ryb**, **temperaturę**, częstotliwość **karmienia**”.

Analiza problemu

- Proces poznawania dziedziny;
- zrozumienie szczegółów;
- ustalenie granic dziedziny.

Przyczyny kłopotów z tworzeniem oprogramowania

- Rozbudowa oprogramowania: zwiększanie zarówno dziedziny jak i funkcjonalności;
- błędne założenia o czasie użytkowania;
- zmienność potrzeb, w tym zmienność prawa;
- problemy zapanowania nad dużym projektem.

Cechy projektu

- Abstrakcja danych i abstrakcja operacji
- Enkapsulacja: ukrycie detali implementacyjnej
- Modularność: podział programu na wyraźne fragmenty
- Hierarchia pojęć

Waga dobrego projektu

- Dziedzina zmienia się wolno
- Funkcjonalność może się szybko zmieniać

Plan wykładu

- 1 Tworzenie oprogramowania
- 2 **Unified Modeling Language**
 - Diagramy klas
 - Przykład
- 3 Analiza obiektowa (Yourdon & Coad)
- 4 Przykład

Unified Modeling Language

Unified Modeling Language (UML)

Notacja graficzna, umożliwiającą ilustrację zagadnień związanych z oprogramowaniem.

Rodzaje diagramów:

- diagramy klas;
- diagramy obiektów;
- diagramy czynności;
- diagramy interakcji;
- ... i jeszcze kilka innych.

Do czego służą diagramy klas

Prezentacja koncepcji.

Do czego służą diagramy klas

Prezentacja koncepcji.

Prezentacja specyfikacji.

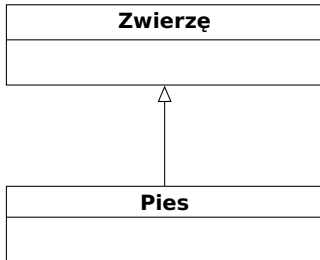
Do czego służą diagramy klas

Prezentacja koncepcji.

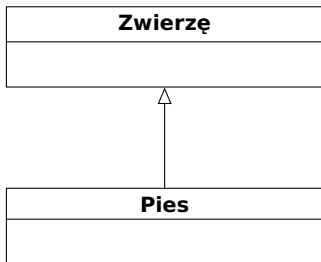
Prezentacja specyfikacji.

Prezentacja implementacji.

Rola UML



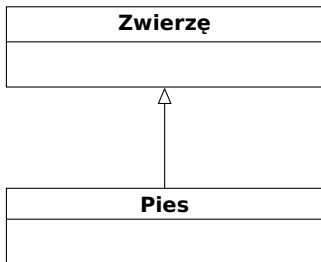
Rola UML



Poziom koncepcyjny

Pies jest zwierzęciem

Rola UML



Poziom koncepcyjny

Pies jest zwierzęciem

Poziom specyfikacji/implementacji

```
public class Zwierzę {}

public class Pies : Zwierzę {}
```


Opis klasy

Opis klasy

- nazwa klasy;

Nazwaklasy

Opis klasy

Nazwa klasy
+ atrybut: typ # ide

Opis klasy

- nazwa klasy;
- atrybuty klasy,
opcjonalnie z typem
attributu i
widzialnością (+:
publiczny, #:
zabezpieczony, -:
prywatny);

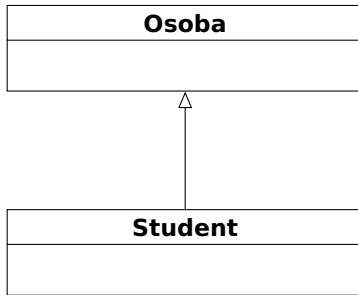
Opis klasy

Nazwakiasy
+ atrybut: typ # ide
operacja1(arg1, arg2) operacja2(arg)

Opis klasy

- nazwa klasy;
- atrybuty klasy,
opcjonalnie z typem
attributu i
widzialnością (+:
publiczny, #:
zabezpieczony, -:
prywatny);
- operacje na klasie,
opcjonalnie z
typami i
widzialnością.

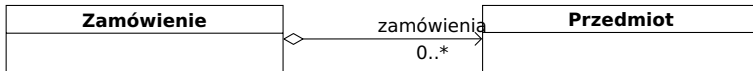
Generalizacja/Specjalizacja



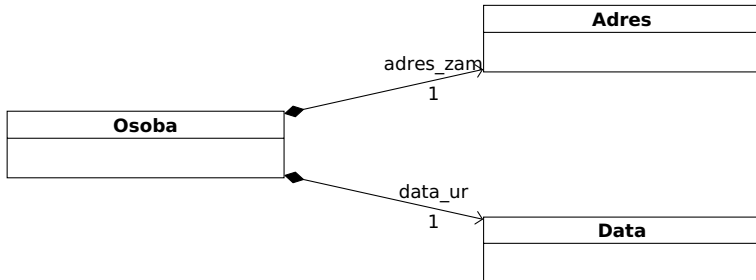
Generalizacja

Specjalizacja

Agregacja



Kompozycja



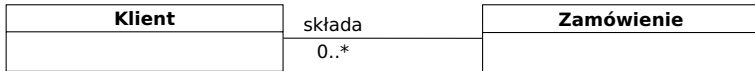
Agregacja vs. kompozycja

Przy **kompozycji** jeśli obiekt "główny" jest usuwany, usuwane są też jego komponenty.

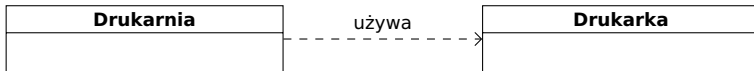
Przy **agregacji**, agregowane elementy mogą istnieć samodzielnie lub mogą być "podpinane" pod inne obiekty.

Asocjacja

Asocjacja to czasowe powiązanie pomiędzy obiektami dwóch klas.



Zależność (dependency)



Asocjacja vs zależność

Asocjacja:

```
class Drukarnia {  
    Drukarka laser;  
    public void wydrukuj() {}  
}
```

Zależność:

```
class Drukarnia {  
    public void wydrukuj(Drukarka laser) {}  
}
```

Drzewo binarnych poszukiwań

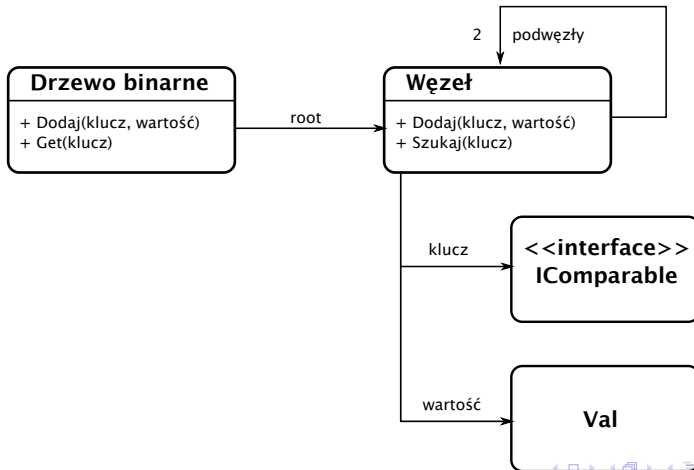
```
public class DrzewoBinarne<Key, Val> {  
    private Węzeł<Key, Val> root = null;  
    public void Dodaj(IComparable<Key> klucz,  
                      Val wartość) { ... }  
}
```

Drzewo binarnych poszukiwań

```
public class DrzewoBinarne<Key, Val> {  
    private Węzeł<Key, Val> root = null;  
    public void Dodaj(Comparable<Key> klucz,  
                     Val wartość) { ... }  
}
```

```
class Węzeł<Key, Val> {  
    Val wartość;  
    Comparable<Key> klucz;  
  
    Węzeł<Key, Val>[] podwężły = new Węzeł<Key, Val>[2];  
}
```

Diagram klas



Plan wykładu

- 1 Tworzenie oprogramowania
- 2 Unified Modeling Language
 - Diagramy klas
 - Przykład
- 3 Analiza obiektowa (Yourdon & Coad)
- 4 Przykład

Podział projektu na warstwy

- 1 Warstwa klas i obiektów
- 2 Warstwa związków
- 3 Warstwa tematów
- 4 Warstwa atrybutów
- 5 Warstwa usług

Klasy i obiekty

- Osoby, przedmioty, pojęcia
- Zdarzenia
- Role
- Jednostki organizacyjne

Warstwa związków

- generalizacja
- specjalizacja
- agregacja
- kompozycja
- asocjacja

Warstwa tematów

Grupowanie klas w tematyczne klastry

Warstwa tematów

Grupowanie klas w tematyczne klastry

Wielkość tematu: wskazówka

7 ± 2

Warstwa atrybutów

Atrybuty proste

Np imię, nazwisko

Warstwa atrybutów

Atrybuty proste

Np imię, nazwisko

Atrybuty złożone

Np. adres

Warstwa atrybutów

Atrybuty proste

Np imię, nazwisko

Atrybuty złożone

Np. adres

Atrybuty obliczane

Np. wiek,

Warstwa atrybutów

Atrybuty proste

Np imię, nazwisko

Atrybuty złożone

Np. adres

Atrybuty obliczane

Np. wiek,

Unikatowe identyfikatory

Własny 'numer seryjny' obiektu.

Warstwa atrybutów

Atrybuty proste

Np imię, nazwisko

Atrybuty złożone

Np. adres

Atrybuty obliczane

Np. wiek,

Unikatowe identyfikatory

Własny 'numer seryjny' obiektu.

Referencje do innych obiektów

Warstwa usług

- Inicjowanie
- Dostęp do atrybutów
- Zmiana stanu

Plan wykładu

- 1 Tworzenie oprogramowania
- 2 Unified Modeling Language
 - Diagramy klas
 - Przykład
- 3 Analiza obiektowa (Yourdon & Coad)
- 4 Przykład

Biblioteka osiedlowa: dane

- Książki i czasopisma
- Czytelnicy
- Karty biblioteczne
- Tymczasowi czytelnicy

Biblioteka osiedlowa: operacje

- Wypożyczenia książek; limity liczbowe i czasowe;
- Statystyki wypożyczeń;

Warstwa klas

Warstwa związków

Warstwa tematów

Warstwa atrybutów

Warstwa usług