Paradygmat programowania obiektowego, klasa, obiekt

dr inż. Jacek Czerniak

Zakres tematyczny (*zarys*)

- Wprowadzenie do podejścia obiektowego
- Przeciążanie nazw funkcji, przeciążanie operatorów.
- Klasy i obiekty. Enkapsulacja, polimorfizm, dziedziczenie.
- Funkcje zaprzyjaźnione, funkcje wirtualne, szablony funkcji.
- Szablony obiektów.
- Pojemniki (kontenery), szablony pojemników.
- Obsługa sytuacji wyjątkowych.
- Projektowanie programów zorientowanych obiektowo.

Paradygmat programowania obiektowego

programowanie obiektowe (OOP) —

(ang. object-oriented programming) to paradygmat rozwiązywania problemów programistycznych z wykorzystaniem **obiektów**, sposób interpretacji problemu jako zbioru **obiektów** i relacji pomiędzy nimi.

Paradygmat

paradygmat *(słownik PWN)* —

przyjęty sposób widzenia rzeczywistości w danej dziedzinie

Paradygmaty programowania

- programowanie proceduralne
- programowanie strukturalne
- programowanie obiektowe

Przykład - model rzeczywistości

- Rzeczywisty obiekt ***pralka*** w komputerze będzie zapisany jako zespół *liczb* i *zachowań*
- ***Liczby*** – cena, rok produkcji, wymiary, kolor itp.
- ***Zachowania*** – zbiór funkcji jakie pralka może dla nas wykonać

Przykład c.d.

- ***Liczby*** i ***funkcje*** zbieramy razem i budujemy z nich pewną całość. Powstaje typ pralka automatyczna.
- Mówimy ***typ***, dlatego, że kreujemy nie jeden *obiekt* ale raczej *klasę* obiektów. Czyli na razie wymyśliliśmy pralkę automatyczną, a jeszcze nie jakiś konkretny egzemplarz.

Przykład 2 - samochód

- Samochód – pewna klasa pojazdów
- Posiada wymiary, funkcjonalności
- Opel Astra III – konkretny obiekt klasy samochód
- Posiada przymioty samochodu, posiada szczegółowe procedury ruchu, bezpieczeństwa itp.

Obiekt

- Potoczne znaczenie słowa obiekt
- Znaczenie pojęcia obiektu w programowaniu
  - Reprezentuje na potrzeby programu obiekty ze świata rzeczywistego lub abstrakcyjne (w potocznym znaczeniu słowa obiekt)
  - Uogólniona zmienna (struktura)
  - Zdefiniowany i używany zgodnie ze składnią i semantyką języka

Obiekt - uogólniona zmienna (struktura)

- Struktura
  - zestaw danych, najczęściej różnych typów
- Uogólniona
  - obiekt = dane + metody operujące na tych danych

Dużo obiektów

- zazwyczaj wiele obiektów ma taki sam zbiór cech, potrzebujemy aby te cechy definiować raz, ale wykorzystywać wielokrotnie
- **klasa** (słownik PWN) — kategoria przedmiotów lub zjawisk wyróżnionych na podstawie wspólnych cech
- Potrzebujemy klasy dla podobnych obiektów

Klasa w programowaniu

- klasa w programowaniu — uogólniony typ zdefiniowany przez użytkownika języka
- służy do definiowania obiektów (uogólnionych zmiennych)
- Dostarcza wielu nowych możliwości
- Pojedyncza klasa powinna jasno reprezentować określone pojęcie, dla którego nie istnieje (jeszcze) odpowiedni typ

Założenia paradygmatu OOP

- Abstrakcja
- Enkapsulacja (hermetyzacja)
- Polimorfizm
- Dziedziczenie

Abstrakcja

Każdy obiekt w systemie służy jako model abstrakcyjnego "wykonawcy", który może wykonywać pracę, opisywać i zmieniać swój stan, oraz komunikować się z innymi obiektami w systemie, bez ujawniania, w jaki sposób zaimplementowano dane cechy. Procesy, funkcje lub metody mogą być również abstrahowane, a kiedy tak się dzieje, konieczne są rozmaite techniki rozszerzania abstrakcji.

Enkapsulacja (hermetyzacja)

Czyli ukrywanie implementacji, hermetyzacja. Zapewnia, że obiekt nie może zmieniać stanu wewnętrznego innych obiektów w nieoczekiwany sposób. Tylko wewnętrzne metody obiektu są uprawnione do zmiany jego stanu. Każdy typ obiektu prezentuje innym obiektom swój "interfejs", który określa dopuszczalne metody współpracy. Pewne języki osłabiają to założenie, dopuszczając pewien poziom bezpośredniego (kontrolowanego) dostępu do "wnętrzności" obiektu. Ograniczają w ten sposób poziom abstrakcji.

Specyfikacja dostępu do składowych klasy

- private:
  // składowe prywatne
  // dostępne dla metod danej klasy
  // oraz metod i funkcji zaprzyjaźnionych
  // private – domyślne dla „class”
- public:
  // składowe publiczne
  // dostępne spoza klasy
  // domyślne dla „struct”
- protected:
  // składowe chronione
  // tak jak private, ale
  // mogą być dodatkowo dostępne dla klas potomnych

Polimorfizm *(wielopostaciowość)*

Referencje i kolekcje obiektów mogą dotyczyć obiektów różnego typu, a wywołanie metody dla referencji spowoduje zachowanie odpowiednie dla pełnego typu obiektu wywoływanego. Jeśli dzieje się to w czasie działania programu, to nazywa się to późnym wiązaniem lub wiązaniem dynamicznym. Niektóre języki udostępniają bardziej statyczne (w trakcie kompilacji) rozwiązania polimorfizmu - na przykład szablony i przeciążanie operatorów w C++.

Dziedziczenie

Porządkuje i wspomaga polimorfizm i enkapsulację dzięki umożliwieniu definiowania i tworzenia specjalizowanych obiektów na podstawie bardziej ogólnych. Dla obiektów specjalizowanych nie trzeba redefiniować całej funkcjonalności, lecz tylko tę, której nie ma obiekt ogólniejszy. W typowym przypadku powstają grupy obiektów zwane klasami, oraz grupy klas zwane drzewami. Odzwierciedlają one wspólne cechy obiektów.

Podejście obiektowe (Alan Kay)

1. Wszystko jest obiektem
2. Program jest zbiorem obiektów które wysyłają sobie komunikaty
3. Każdy obiekt posiada pamięć na która składają się inne obiekty
4. Każdy obiekt posiada swój typ
5. Wszystkie obiekty tego samego typu mogą otrzymywać te same komunikaty

Cechy obiektu

- **stan** – dane wewnętrzne
- **zachowanie** – zestaw metod do wykonania
- **identyfikacja** – każdy obiekt można w sposób jednoznaczny odróżnić od innych obiektów

Obiekty i klasy

- **Klasa** to złożony typ danych, składający się z pól, przechowujących dane, oraz posiadający metody, wykonujące zaprogramowane czynności.
- **Obiekt** może reprezentować cokolwiek. Każdy obiekt należy do pewnej klasy. Definicja klasy zawiera pola, z których składa się ów obiekt, oraz metody, którymi dysponuje.
- **Klasa** jest wiec **wzorcem** na podstawie którego powołujemy do życia obiekty

Pola i metody

- obiekty zawierają **pola**, czyli zmienne. Ich rola jest przechowywanie pewnych informacji o obiekcie - jego charakterystyki
- obiekt może wykonywać na sobie pewne działania, a wiec uruchamiać zaprogramowane funkcje; nazywamy je **metodami** albo funkcjami składowymi. Czynią one obiekt tworem aktywnym - nie jest on jedynie pojemnikiem na dane, lecz może samodzielnie nimi manipulować

Metody

- **Metoda** - w programowaniu obiektowym jest to funkcja składowa klasy, której zadaniem jest działanie na rzecz określonych elementów danej klasy lub klas z nią spokrewnionych (zob. też dziedziczenie).
- Metody wiąże się z klasami głównie po to, aby nie zaśmiecać kodu źródłowego i samego programu nadmierna ilością funkcji globalnych, które i tak nie zostaną użyte w celu innym, niż na rzecz konkretnej klasy.
- Inną ich zaletą jest to, że metoda wewnętrzna danej klasy ma dostęp do wszystkich składników tej klasy (także prywatnych i chronionych), bez konieczności deklarowania zaprzyjaźnienia.

Języki obiektowe

- Simula 67
- Ada, BASIC, Lisp, Pascal – *dodano obiektowość*
- Eiffel
- Smalltalk
- C++
- **Java**
- C# (Visual Studio.NET) } *usługi sieciowe*
- Perl, Python, Ruby, PHP - *obiektowe języki skryptowe*

Koniec

Dziękuję za uwagę

dr inż. J.Czerniak
jczerniak@ukw.edu.pl