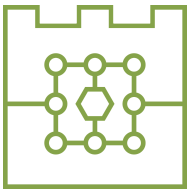

Programowanie w języku Java

Laboratorium nr 2

Politechnika Krakowska
Wydział Informatyki i Telekomunikacji
Katedra Informatyki

2024

Wprowadzenie do rekordów

Important Information

Rekordy (`record`) to specjalny rodzaj klasy w Javie, wprowadzony w wersji 14 (w trybie preview) i stabilny od Javy 16.
W przypadku korzystania z Javy 14 należy ustawić flagę:
javac –release 14 –enable-preview NazwaKlasy.java

Rekordy służą do reprezentowania **niezmiennych struktur danych** (immutable data).

Rekordy automatycznie generują:

- Konstruktor kanoniczny,
- Metody `toString()`, `equals()` i `hashCode()`,
- Gettery do wszystkich pól.

Rekordy eliminują potrzebę ręcznego pisania powtarzalnego kodu, takiego jak gettery czy metoda `toString()`.

Definicja rekordu

Rekord jest definiowany za pomocą słowa kluczowego `record`. Składnia wygląda następująco:

```
1 public record Book(String name, double price) { }
```

Cechy rekordów

1. **Niemutowalność:** Pola są automatycznie `final`, co oznacza, że raz utworzony obiekt nie może zmieniać wartości swoich pól.
2. **Prostota:** Minimalna ilość kodu do stworzenia kompletnej klasy danych.
3. **Konstruktor kanoniczny:** Rekordy mają automatycznie generowany konstruktor kanoniczny, który przyjmuje wszystkie pola rekordu w odpowiedniej kolejności.
4. **Alternatywne konstruktory:** Można zdefiniować dodatkowe konstruktory, które np. ustawiają domyślne wartości dla niektórych pól.

Porównanie rekordów z klasami

Konstruktor kanoniczny

Konstruktor kanoniczny to konstruktor automatycznie generowany przez kompilator, który inicjalizuje wszystkie pola rekordu.

Cecha	Rekord	Klasa
Słowo kluczowe	`record`	`class`
Niemutowalność	Tak (`final`)	Nie (należy ręcznie ustawić)
Generowanie kodu	Automatyczne (`toString`, `equals`, `hashCode`)	Ręczne pisanie kodu
Dziedziczenie	Nie	Tak
Zastosowanie	Proste struktury danych	Uniwersalne użycie

Tabela 1: Porównanie rekordów z klasami

```
    public Book(String name, double price) {
    this.name = name;
    this.price = price;}
```

Dodanie walidacji

Jeśli chcemy dodać logikę (np. sprawdzenie zakresu wartości), możemy jawnie zdefiniować konstruktor:

```
public record Book(String name, double price) {
    public Book {
        if (price < 0) {
            throw new IllegalArgumentException("Price can't
                ↪ be negative.");
        }
        if (name == null || name.isBlank()) {
            throw new IllegalArgumentException("Book name
                ↪ can't be empty.");
        }
    }
}
```

Alternatywny konstruktor

```
    public Book(String name) {
        this(name, 0.0); // Default price is set to 0.0}
```

Konstruktor public **Book(String name)** pozwala na utworzenie obiektu z nazwą książki, przypisując domyślną cenę (np. 0.0). Wewnątrz alternatywnego konstruktora wywoływany jest główny konstruktor poprzez **this(name, 0.0)**.

Dodatkowe materiały

- Dokumentacja: https://docs.oracle.com/en/java/javase/16/language/records.html
  https://javastart.pl/baza-wiedzy/slownik/rekordy

- Model RGB (RGBA) jest opisany tutaj:
https://pl.wikipedia.org/wiki/RGBA

Zadania:

1. Zaimplementuj klasę Color reprezentującą kolor w modelu RGB i korzystając z konceptu rekordów w Javie (słowo kluczowe record, rozwiązania tworzące klasę Color będą ocenione na 0 punktów).

2. Model RGB może być rozszerzony o kanał Alpha. Wszystkie cztery parametry powinny być obsługiwane przez konstruktor kanoniczny a dodatkowo powinna być możliwość tworzenia rekordu bez podawania wartości kanału alfa - wtedy ma on otrzymywać domyślną wartość 0.

3. Zademonstruj w funkcji main tworzenie obiektów typu Color, wypisywanie ich składowych oraz działanie metody toString().

Important Information

Jako wynik zadania należy przesłać **plik PDF** zawierający cały kod oraz wynik uruchomienia metody main. (PDF może zawierać link do repozytorium GitHub, natomiast w pliku README w repozytorium należy umieścić zrzut ekranu z wynikiem działania programu.)