

Zaawansowane programowanie obiektowe

Lab. 6

(Proste wzorce projektowe, polimorfizm, Date and Time API)

Zadanie nr 1 (1.25 pkt + 0.75 pkt = 2 pkt)

Zapoznaj się z artykułami:

[http://pl.wikipedia.org/wiki/Dekorator_\(wzorzec_projektowy\)](http://pl.wikipedia.org/wiki/Dekorator_(wzorzec_projektowy))

[http://pl.wikipedia.org/wiki/Fabryka_abstrakcyjna_\(wzorzec_projektowy\)](http://pl.wikipedia.org/wiki/Fabryka_abstrakcyjna_(wzorzec_projektowy))

Podstawową modelu ma być interfejs **Publikacja**, w którym są deklaracje metod zwracających autora, tytuł i liczbę stron publikacji. Publikację implementuje klasa konkretna **Ksiazka** (zawierająca odpowiednie pola, czyli posiadająca stan). Dla książki k przeddefiniuj m.in. funkcję toString(...), tak aby instrukcja:

```
System.out.println(k);
```

wyświetlała tekst w rodzaju:

```
| Adam Mickiewicz | Pan Tadeusz | 340 |
```

1. (Dekorator)

Dekorator pozwala tworzyć udekorowane książki w locie – w oparciu o istniejące instancje książek – bez kopiowania zawartości. Dekoracja działa podobnie jak dziedziczenie, tyle że odbywa się w czasie uruchomienia, a nie kompilacji.

Dodatkowo, udekorowaną książką można posługiwać się tak, jak książką zwykłą.

Dekorację wykonamy poprzez dodanie okładek:

KsiazkaZOkladkaZwykla,

KsiazkaZOkladkaTwarda,

KsiazkaZObwoluta,

oraz dodanie autografu autora:

KsiazkaZAutografem.

Informacje o „udekorowaniu” danej książki mają być zwracane przez funkcję toString.

Użycie dekoratora:

```
Publikacja k1 = new Ksiazka("Adam Mickiewicz", "Pan Tadeusz", 340);
```

```
Publikacja k2 = new Ksiazka("Adam Mickiewicz", "Dziady", 130);
```

```
Publikacja kk1 = new KsiazkaZOkladkaZwykla(k1);
```

```
Publikacja kk2 = new KsiazkaZOkladkaTwarda(k2);
```

```
// Publikacja fakeBook = new KsiazkaZObwoluta(k1);
```

```
// wyjątek! Nie można obłożyć obwolutą książki, która nie posiada okładki
```

```
Publikacja kkk2 = new KsiazkaZObwoluta(kk2); // a tu OK
```

```
// Publikacja odrzut = new KsiazkaZObwoluta(kkk2);
```

```
// wyjątek! Obwoluta może być jedna
```

```
Publikacja dziadyZAutografemWieszczka =
```

```
    new KsiazkaZAutografem(kk2, "Drogiej Hani - Adam Mickiewicz");
```

```
System.out.println(dziadyZAutografemWieszczka);
```

// wypisuje: | Adam Mickiewicz | Dziady | 130 | Okładka twarda | Drogiej Hani - Adam Mickiewicz |

```
Publikacja dziadyZDwomaAutografami = new KsiazkaZAutografem(  
    dziadyZAutografemWieszczu, "Haniu, to nieprawda, Dziady napisałem ja,  
    Julek Słowacki!");  
    // wyjątek! Autograf może być tylko jeden
```

Ogólnie biorąc, obowiązują następujące zastrzeżenia:

- okładka może być tylko jedna,
- nie można obłożyć obwolutą książki, która nie posiada okładki,
- obwoluta może być tylko jedna,
- książka może mieć tylko 1 autograf (ale nawet książka bez okładki może posiadać autograf).

2. (Fabryka abstrakcyjna)

Jest to wzorzec, który umożliwia tworzenie obiektów należących do rodziny.

Założmy, że mamy trzy gatunki książek:

- powieść historyczna (PowiescHistoryczna),
- thriller (Thriller),
- poemat (Poemat).

Książki te dziedziczą z klasy Ksiazka.

Mamy też abstrakcyjne wydawnictwo, z którego dziedziczą:

- wydawnictwo powieści historycznych,
- wydawnictwo thrillerów,
- wydawnictwo poematów.

Użycie:

```
Wydawnictwo w = Wydawnictwo.getInstance("Józef Ignacy Kraszewski");  
/* W zależności od autora wybieramy odpowiednie wydawnictwo. Wpisać kilka wariantów.  
Tu powstanie wydawnictwo powieści historycznych */  
Ksiazka k = w.createBook("Masław", 280);  
/* Tworzy książkę klasy PowiescHistoryczna z podanym tytułem i liczbą  
stron. Autor przekazany będzie z wydawnictwa */
```

Zadanie nr 2 (0.5 pkt)

Korzystając z Date and Time API Javy 8 policz:

- a) ile dni trwała II wojna światowa (wliczając zarówno 1.09.1939, jak i 8.05.1945);
- b) którego dnia i miesiąca wypada 68-my dzień roku 2016;
- c) ile w swoim życiu przeżyłeś/aś dni 29 lutego. Nie zakładaj z góry, że np. bieżący rok nie jest przestępny.