STUDIA PODYPLOMOWE

Java EE — produkcja oprogramowania

(do użytku wewnętrznego)

Klasy i obiekty

Roman Podraza

Instytut Informatyki Politechniki Warszawskiej

O materiałach

- Wszystkie treści niniejszych materiałów zostały zaczerpnięte z podręcznika:

  Cay S. Horstmann, Gary Cornell, „Java. Podstawy. Wydanie IX”, Helion SA, 2013, ISBN 978-83-246-7758-0

Podręcznik

- Cay S. Horstmann, Gary Cornell, „Java. Podstawy. Wydanie IX”, Helion SA, 2013, ISBN 978-83-246-7758-0
  - Rozdział 4 Obiekty i klasy
- Cay S. Horstmann, Gary Cornell, Core Java Volume I--Fundamentals (9th Edition) (Core Series), Copyright © 2013 Oracle and/or its affiliates, ISBN-13: 978-037081899
- Kody źródłowe http://horstmann.com/corejava

Wstęp do programowania obiektowego

- Klasy
- Obiekty
- Identyfikacja klas
- Relacje między klasami
  - Dziedziczenie („jest czymś”)
  - Agregacja („ma coś”)
  - Zależność (asocjacja) („używa czegoś”)

Używanie klas predefiniowanych

- Obiekty i odnośniki (zmienne obiektów)
- Klasa *GregorianCalendar* z biblioteki Javy
- Metody udostępniające i zmieniające wartość elementu (ang. accessor, mutator)
- Przykład - *CalendarTest.java*
- *java.util.GregorianCalendar (1.1)*
  - *GregorianCalendar()*
  - *GregorianCalendar(int year, int month, int day)*
  - *GregorianCalendar(int year, int month, int day, int hour, int minutes,int seconds)*

Używanie klas predefiniowanych

- *int get(int field)*
  - *Field: Calendar.ERA, Calendar.YEAR, Calendar.MONTH, Calendar.WEEK_OF_YEAR, Calendar.WEEK_OF_MONTH, Calendar.DAY_OF_MONTH, Calendar.DAY_OF_YEAR, Calendar.DAY_OF_WEEK, Calendar.DAY_OF_WEEK_IN_MONTH, Calendar.AM_PM, Calendar.HOUR, Calendar.HOUR_OF_DAY, Calendar.MINUTE, Calendar.SECOND, Calendar.MILLISECOND, Calendar.ZONE_OFFSET, Calendar.DST_OFFSET*
- *void set(int field, int value)*
- *void set(int year, int month, int day)*
- *void set(int year, int month, int day, int hour, int minutes, int seconds)*

Używanie klas predefiniowanych

- *void add(int field, int amount)*
- *int getFirstDayOfWeek()*
- *void setTime(Date time)*
- *Date getTime()*
- *String[] getShortWeekdays()*
- *String[] getShortMonths()*
- *String[] getWeekdays()*
- *String[] getMonths()*

Uwaga: miesiące przyjmują wartości od 0 do 11!

Tworzenie własnych klas

- Przykład - *EmployeeTest.java*
- Używanie wielu plików źródłowych
- Pierwsze kroki w tworzeniu konstruktorów
- Parametry jawne i niejawne (domniemane)
- Korzyści z hermetyzacji (enkapsulacji)
- Reguły dostępu do elementów klasy
- Metody prywatne
- Stałe jako pola klasy

Pola i metody statyczne

- Pola statyczne
- Stałe statyczne (publiczne)
- Metody statyczne
- Metody fabryczne

```
NumberFormat currencyFormatter = NumberFormat.getCurrencyInstance();
NumberFormat percentFormatter = NumberFormat.getPercentInstance();
double x = 0.1;
System.out.println(currencyFormatter.format(x)); // prints $0.10
System.out.println(percentFormatter.format(x)); // prints 10%
```

- Metoda *main()*
- Przykład - *StaticTest.java*

Parametry metod

- Parametry metod są przekazywane przez wartość
- Przykład - *ParamTest.java*

Konstruowanie (tworzenie) obiektów

- Przeciążanie (nazw funkcji)
- Inicjacja pól wartościami domyślnymi
  - Zero (odpowiedniego typu)
- Konstruktor bezparametrowy (bezargumentowy)
- Jawna inicjalizacja pól (przy ich deklaracji)
  - Również przez wywołanie funkcji
- Nazywanie parametrów
  - Przesłanianie nazw
- Wywoływanie innego konstruktora

Konstruowanie (tworzenie) obiektów

- Bloki inicjujące
  - Statyczne bloki inicjujące(dla statycznych pól)
- Przykład - *ConstructorTest.java*
- *java.util.Random (1.0)*
  - *Random()*
  - *int nextInt(int n) (1.2)*
- Niszczenie obiektu i metoda *finalize()*

Pakiety

- Importowanie klasy
- Importy statyczne
- Dodawanie klasy do pakietu
- Przykład - *PackageTest.java*
- Zasięg pakietów (zakres widoczności)

Ścieżka (do) klas

- Specyfikacja w komendzie uruchomienia programu

```
java -classpath /home/user/classdir:.:/home/user/archives/archive.jar MyProg
```

```
java -classpath c:\classdir;.;c:\archives\archive.jar MyProg
```

- Ustawienie zmiennej środowiskowej *CLASSPATH*

```
export CLASSPATH=/home/user/classdir:.:/home/user/archives/archive.jar
```

```
setenv CLASSPATH /home/user/classdir:.:/home/user/archives/archive.jar
```

```
set CLASSPATH=c:\classdir;.;c:\archives\archive.jar
```

Komentarze dokumentacyjne : narzędzie ***javadoc***

- Wstawianie komentarzy
  - Komentarze zaczynające się od specjalnej sekwencji znaków **`/**`**
  - Narzędzie javadoc pobiera informacje dotyczące następujących elementów:
    - pakietów,
    - klas i interfejsów publicznych,
    - publicznych i chronionych (protected) metod i konstruktorów,
    - pól publicznych i chronionych.
  - Komentarz powinien się znajdować bezpośrednio nad tym, czego dotyczy

Komentarze dokumentacyjne : narzędzie ***javadoc***

- Komentarze do klas
- Komentarze do metod
  - **@param** opis zmiennej: Dodaje pozycję do sekcji Parameters metody. Opis może zajmować kilka wierszy i zawierać znaczniki HTML. Wszystkie znaczniki @param dotyczące jednej metody powinny się znajdować w jednym miejscu.
  - **@return** opis: Dodaje sekcję Returns. Opis może zajmować kilka wierszy i zawierać znaczniki HTML.
  - **@throws** opis klasy: Dodaje informację, że dana metoda może spowodować wyjątek.

Komentarze dokumentacyjne : narzędzie ***javadoc***

- Komentarze do pól
  - **@author** imię i nazwisko: Dodaje pozycję Author. Jeśli jest kilku autorów, można zastosować kilka znaczników @author.
  - **@version** tekst: Dodaje pozycję Version. Tekst może być opisem aktualnej wersji.
  - **@since** tekst: Dodaje pozycję Since. Tekst to opis wersji, w której wprowadzono daną zmianę
  - **@deprecated** tekst: Dodaje komentarz informujący, że dana klasa, metoda lub zmienna nie powinny być używane oraz informację o zamienniku.

Komentarze dokumentacyjne : narzędzie ***javadoc***

- **@see** odwołanie: Dodaje hiperłącze w sekcji *See also*. Odwołanie może mieć jedną z poniższych form:
  - pakiet.klasa#struktura etykieta
  - etykieta
  - "tekst"Komentarze ogólne
- **@link** odwołanie: Hiperłącze w tekście dokumentacji - jak dla znacznika @see

- Komentarze do pakietów i ogólne
  - plik HTML o nazwie *package.html*. Zostanie pobrane wszystko, co znajduje się pomiędzy znacznikami <body> i </body>.

Komentarze dokumentacyjne : narzędzie ***javadoc***

  - plik HTML o nazwie *overview.html*. Zostanie pobrane wszystko, co znajduje się pomiędzy znacznikami <body> i </body>.

- Generowanie dokumentacji

Porady dotyczące projektowania klas

- Dane powinny być prywatne.
- Dane powinny być zawsze zainicjowane.
- Nie należy stosować zbyt wielu różnych podstawowych typów danych w jednej klasie.
- Nie wszystkie pola wymagają własnych metod dostępu i zmiany.
- Klasy o zbyt dużej funkcjonalności powinny być dzielone.
- Nazwy metod i klas (oraz zmiennych) powinny odpowiadać ich przeznaczeniu.


POLITECHNIKA WARSZAWSKA
WARSAW UNIVERSITY OF TECHNOLOGY

Dziękuję