Język JAVA
podstawy programowania

Jacek Rumiński

Wykład 3, część 1

Plan wykładu:

1. Konstrukcja kodu programów w Javie
2. Identyfikatory, zmienne
3. Typy danych
4. Operatory, instrukcje sterujące – instrukcja warunkowe,
5. Instrukcje sterujące – pętle, instrukcje wyboru, instrukcje powrotu

Bloki kodu

Podstawą tworzenia kodu w sposób czytelny jest grupowanie kodu w bloki, wyróżnialne wcięciami. Każdy blok kodu oznaczony jest nawiasami klamrowymi, np.:

```
instrukcja_grupująca{
        kod...
        kod...
} // koniec instrukcja_grupująca
```

Przykłady instrukcji grupujących:
definicja klasy class{}, definicja metody metoda(){}, bloki kodu dla instrukcji sterującej (np. if (warunek){}), statyczne wykonanie instrukcji podczas wczytania klasy static {}, itd.

Cenne wskazówki kompozycji kodu

1. Zawsze stosuj wcięcie (tabulacja, spacje).
2. Stosuj komentarze oraz opisuj komentarzem koniec bloku kodu.
3. Rozdrabniaj funkcjonalność na wiele klas – każdą klasę przechowuj w oddzielnym pliku.
4. W wymyślanych nazwach (np. klas, metod) stosuj jeden język, np.
   nie *getWiek()*

1. Stosuj nadmiarowość celem poprawienia czytelności kodu, np.:
   *int a=(c * 2) + (b / 3);* **zamiast** *int a=c* 2 + b/ 3;*
5. Stosuj separatory, np.:
   *int a=(c * 2) + (b / 3);* **zamiast** *int a=(c*2)+(b/3);*
6. Bądź konsekwentny w oznaczeniach, np.:
   zawsze *int[] n;* ***albo*** *int n[].*

Staraj się wypracować u siebie dobre nawyki (kosztowne czasowo)!

Komentarze

Ważnym elementem czytelnej konstrukcji kodu jest używanie dokumentacji kodu za pomocą komentarzy. W Javie stosuje się trzy podstawowe typy komentarza:

- **komentarz liniowy (jedna linia):**

// miejsce na komentarz do końca linii

- **komentarz blokowy (wiele linii):**

```
/*
miejsce na komentarz w dowolnym miejscu bloku
*/
```

- **komentarz dokumentacyjny (wiele linii)**

```
/**
miejsce na komentarz w dowolnym miejscu bloku
*/
```

Znaczenie komentarzy->

Komentarze

Komentarz liniowy (jedna linia):
wykorzystuje się do krótkiego oznaczania kodu np. interpretacji zmiennych, oznaczania końca bloku, itp.

Komentarz blokowy (wiele linii):
stosuje się do wprowadzania szerszych opisów kodu a czasem do chwilowego wyłączania kodu z kompilacji (proste testowanie)

Komentarz dokumentacyjny (wiele linii) – nowość w Javie
używa się do tworzenia dokumentacji kodu polegającej na opisie klas i metod, ich argumentów, dziedziczenia, twórców kodu, itp.
Specjalny program w JDK (javadoc.exe) umożliwia automatyczną generację dokumentacji w formacie HTML na podstawie komentarzy dokumentacyjnych!

Komentarze dokumentacyjne

Komentarz dokumentacyjny w Javie może zawierać dodatkowe elementy takie jak:

@author - umożliwia podanie autora kodu,
@version - umożliwia podanie wersji kodu,
@see - umożliwia stworzenie referencji,
itp.

Pełny zestaw elementów formatujących znajduje się w opisie narzędzia *javadoc.exe* (dokumentacja JDK).

Zróbmy przykład!

Język JAVA – komentarz dokumentacyjny

Kod programu: OpisaniJedi.java

cdn.

```
 /**
<b>NowyRycerzJedi </b> określa klasę rycerzy Jedi
@author Jacek Rumiński
@version 1.0
*/
class NowyRycerzJedi{
        /**     Definiuje nazwę rycerza Jedi */
        String nazwa;
        /**     Określa kolor miecza  rycerza Jedi */
        String kolor_miecza;
        /**     Konstruktor umożliwia podanie właściwości obiektu */
        NowyRycerzJedi(String nazwa, String kolor_miecza){
                this.nazwa=nazwa;
                this.kolor_miecza=kolor_miecza;
        }
        /**     Wyświetla zestaw właściwości rycerza Jedi */
        void opis(){
                System.out.println("Rycerz "+nazwa+ " ma "+kolor_miecza+" miecz.");
        }
 }// koniec class NowyRycerzJedi
```

Język JAVA - komentarz dokumentacyjny

Kod programu: OpisaniJedi.java cd.

```
 /**
OpisaniJedi określa klasę główna aplikacji
@author Jacek Rumiński
@version 1.0
*/
public class OpisaniJedi{

        public static void main(String args[]){
                NowyRycerzJedi luke = new NowyRycerzJedi("Luke", "niebieski");
                NowyRycerzJedi ben = new NowyRycerzJedi("Obi-wan","biały");
                luke.opis();
                ben.opis();
        }// koniec public static void main(String args[])

}// koniec public class Jedi3
```

Tworząc dokumentację HTML dla kodu z przykładu 2.1 należy w następujący sposób wykorzystać narzędzie *javadoc*:

javadoc -private -author -version OpisaniJedi.java

Plan wykładu:

1. Konstrukcja kodu programów w Javie
2. Identyfikatory, zmienne
3. Typy danych
4. Operatory, instrukcje sterujące – instrukcja warunkowe,
5. Instrukcje sterujące – pętle, instrukcje wyboru, instrukcje powrotu

Identyfikatory – nazwy klas, metod, pól, pakietów, itd.

Identyfikator jest unikalny dla kompilatora, jednocześnie powinien wskazywać człowiekowi po co został utworzony (lub jakie jest jego znaczenie). Ważne jest zatem właściwe dobranie nazwanie identyfikatora!

Identyfikator tworzy się korzystając z liter (bez znaków narodowych), liczb, znaku podkreślenia „_” oraz znaku „$”. Tworzona nazwa nie może się jednak zaczynać liczbą. Pierwszym znakiem może być litera bądź symbol podkreślenia „_” lub „$”.

W Javie rozróżnialne są wielkie i małe litery w nazwach, tak więc nazwy: *Jedi* i *jedi* oznaczać będą dwa różne identyfikatory!
Tworząc własny kod (poza nauką, itp.) warto stosować wyłącznie nazwy w języku angielskim (globalna świat zasobów)!

Identyfikatory.

Tworząc identyfikator należy pamiętać, że nie może on mieć nazwy identycznej ze słowem kluczowym.
Słowa kluczowe to identyfikatory o specjalnym znaczeniu dla języka Java. Identyfikatory te są już zdefiniowane dla języka i posiadają określone znaczenie w kodzie programu.

Przykładowe słowa kluczowe: public, class, for, if, int, char, switch, itd.

Pełny wykaz słów kluczowych można znaleźć w specyfikacji języka Java:

http://java.sun.com/docs/books/jls/

(książka za darmo !)

Identyfikatory – wskazówki tworzenia

Kilka wskazówek tworzenia identyfikatorów:
-Identyfikatory klas z rozpoczynamy z wielkiej litery, np. Rycerz;
-Nazwy pakietów, pól, zmiennych, metod, funkcji rozpoczynamy z małej litery, np. org.jwr, wiek, wzrost, drukuj(), wyslij();
-Nazwy stałych (o czym później) wielkimi literami, np. PI.

Budując identyfikator złożony (z wielu słów) warto stosować jedną z następujących reguł:
-każde kolejne słowo zaczyna się od wielkiej litery (preferowany w Javie): np. RycerzJedi, drukujStanArmii(), liczbaDzial;
-każde kolejne słowo jest poprzedzone znakiem podkreślenia „_” (preferowane dla stałych), np. LICZBA_OKIEN

Zmienne

Zmienna to element dane typu, którego wartość może zostać zmieniona. Zmienna może być atrybutem obiektu.

Przykładowo atrybutem człowieka może być liczba zębów. Wartość tego atrybutu się zmienia w czasie. Co wiemy o takim atrybucie?

1. Możemy go nazwać (stosując reguły w Javie) np. liczbaZebow;
2. Atrybut ten może przechowywać wartości liczbowe od 0 (brak zębów) do 32 (typowo), np. liczbaZebow=10, liczbaZebow=31;
3. Atrybut ten jest zatem określonego typu (danych). W systemach komputerowych każdy typ danych musi być precyzyjnie określony i nazwany (słowa kluczowe). Dla naszego przykładu weźmy typ int (liczby całkowite), czyli: int liczbaZebow.
4. Każdy podstawowy typ danych oznacza jednocześnie rozmiar pamięci jaki jest rezerwowany do przechowywania wartości zmiennej danego typu, np. typ int – cztery bajty (4B).

Zmienne

5. Rozmiar przydzielanej pamięci (przechowywania wartości zmiennej) dla danego typu oznacza zakres możliwych wartości jakie może przyjmować zmienna, np. int, 4B=4*8bitów=32bity -> zakres = 2^{32}; w Javie wszystkie liczby są ze znakiem (czyli od ujemnych do dodatnich, 1 bit na oznaczenie znaku, pozostaje 31 bitów) zatem

   MIN_VALUE= -2^{31} = -2147483648
   MAX_VALUE= $2^{31}-1$ = 2147483647
   (w symetrii równego podziału na liczby dodatnie i ujemne wartość „0” jest traktowana jako dodatnia i wówczas po każdej stronie znaku mamy 2147483648 pozycji).
6. Istnieje wiele typów danych i każdy swój ma rozmiar i możliwe wartości!

Zmienne

7. Zmiennym można przypisywać wartość (zmieniać) oraz odczytywać wartość:
   przypisanie wartości (zastosowanie operatora „=”):
   (set) liczbaZebow=10;
   odczytanie wartości (podanie nazwy zmiennej):
   (get) System.out.println("Liczba zębów to: "+liczbaZebow);
8. Zmienna może zostać ograniczona do stałej poprzez użycie dwóch słów kluczowych (specyfikatorów):
   static – oznaczony element jest jeden, należy do danej klasy, czyli nie trzeba tworzyć obiektu, aby z takiego elementu korzystać,
   final – oznaczony element ma ostateczną (końcową) definicję, czyli przypisana wartość nie może ulec zmianie,
   przykładowo (public – stała dostępna dla każdego kodu):
   public static final int MAX_LICZBA_ZEBOW=32;

Plan wykładu:

1. Konstrukcja kodu programów w Javie
2. Identyfikatory, zmienne
3. Typy danych (wykład 3, część 2)
4. Operatory
5. Instrukcje sterujące – instrukcje warunkowe,
6. Instrukcje sterujące – pętle, instrukcje wyboru, instrukcje powrotu