

Typy Danych i Zmienne

Typy danych

Zmienne (deklaracja, inicjalizacja)

Operacje na zmiennych

Typy danych

Typ	Przykład Wartości
byte (Liczby całkowite od -128 do 127)	123
short (Liczby całkowite od -32768 do 32767)	12345
int (Liczby całkowite od -2^{31} do $2^{31}-1$)	12345
long (Liczby całkowite od -2^{63} do $2^{63}-1$)	123456789123l
float (Duże liczby zmiennoprzecinkowe)	1234.56f
double (Bardzo duże liczby zmiennoprzecinkowe)	2345678.6789
char (Pojedyncze znaki unicode)	'a', '?', '\u2602'
boolean (Typ logiczny – prawda, fałsz)	true, false
String (Ciąg znaków)	"Kot", "Kot\nPies"

Typy danych, które na razie wystarczy pamiętać (demo)

Typ	Przykład Wartości
int (Liczby całkowite od -2^{31} do $2^{31}-1$)	12345
double (Bardzo duże liczby zmiennoprzecinkowe)	2345678.6789
char (Pojedyncze znaki unicode)	'a', '?', '\u2602'
boolean (Typ logiczny – prawda, fałsz)	true, false
String (Ciąg znaków)	"Kot", "Kot\nPies"

Zmienne

Zmienna pozwala nadać wartości nazwę a następnie się do niej po tej nazwie odwoływać

Tworzenie zmiennej dzielimy na dwa etapy:

- deklarację (określenie typu i nazwy)
- inicjalizację (przypisanie wartości)

Deklarację i inicjalizację można połączyć i zapisać razem

Deklaracja:

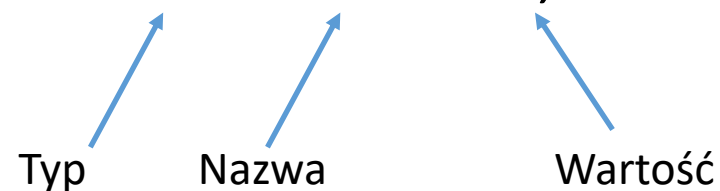
`int` liczba;

Inicjalizacja:

liczba = 5;

albo deklaracja + inicjalizacja:

`int` liczba = 5;



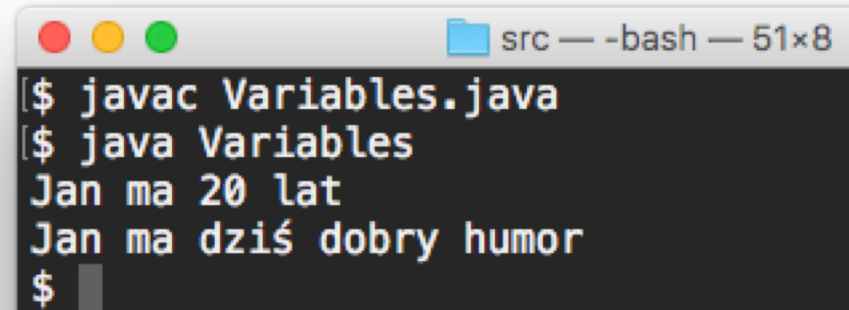
Zmienne – konwencja nazewnictwa

- Nazwa zmiennej powinna oddawać jej funkcję, unikamy nazw typu "a", "b", "xyz"
- Zmienne nazywamy zgodnie z konwencją camelCase, rozpoczynając od małej litery, np.
 - "nazwaZmiennej", "employeesNumber" ✓
 - "NazwaZmiennej", "employees_Number"
- Nazwy zmiennych muszą rozpoczynać się od litery lub znaku podkreślenia, ale mogą też zawierać liczby
- Nie można zadeklarować dwóch zmiennych o takiej samej nazwie w tym samym zasięgu (pomiędzy tymi samymi nawiasami klamrowymi), np.:

```
int number = 5;  
int number = 10; //błąd
```

Zmienne – przykład

```
class Variables {  
    public static void main(String[] args) {  
        String name = "Jan";  
        int age = 20;  
  
        System.out.println(name + " ma " + age + " lat");  
        System.out.println(name + " ma dziś dobry humor");  
    }  
}
```

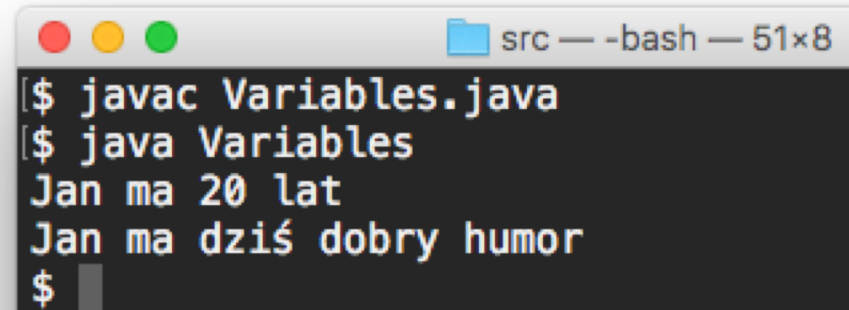


A terminal window titled 'src — -bash — 51x8' showing the compilation and execution of the Variables.java program. The commands and their outputs are as follows:

```
$ javac Variables.java  
$ java Variables  
Jan ma 20 lat  
Jan ma dziś dobry humor  
$
```

Zmienne – przykład

```
class Variables {  
    public static void main(String[] args) {  
        String name = "Jan";  
        int age = 20;  
  
        System.out.println(name + " ma " + age + " lat");  
        System.out.println(name + " ma dziś dobry humor");  
    }  
}
```

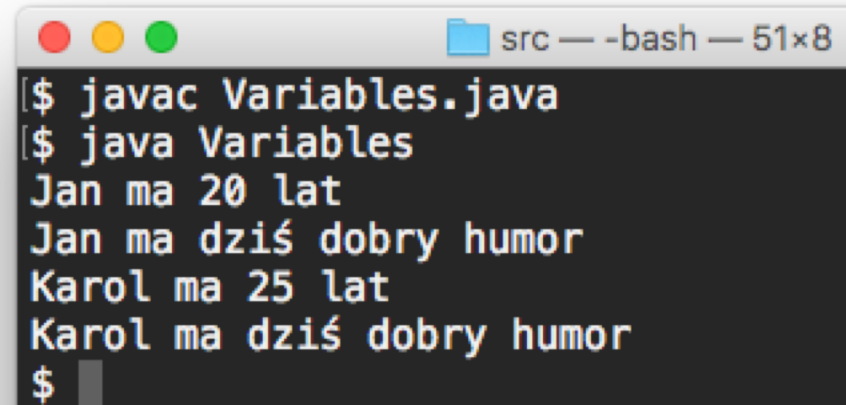


A terminal window titled 'src — -bash — 51x8' showing the compilation and execution of the Variables.java program. The commands and their outputs are as follows:

```
$ javac Variables.java  
$ java Variables  
Jan ma 20 lat  
Jan ma dziś dobry humor  
$
```

Zmienne – przykład

```
class Variables {  
    public static void main(String[] args) {  
        String name = "Jan";  
        int age = 20;  
  
        System.out.println(name + " ma " + age + " lat");  
        System.out.println(name + " ma dziś dobry humor");  
  
        //zmieniamy imię i wiek  
        name = "Karol";  
        age = 25;  
  
        System.out.println(name + " ma " + age + " lat");  
        System.out.println(name + " ma dziś dobry humor");  
    }  
}
```



A terminal window titled 'src — -bash — 51x8' showing the compilation and execution of the Variables.java program. The commands and their outputs are as follows:

```
$ javac Variables.java  
$ java Variables  
Jan ma 20 lat  
Jan ma dziś dobry humor  
Karol ma 25 lat  
Karol ma dziś dobry humor  
$
```


Zmienne finalne

- Zmienna poprzedzona słowem kluczowym **final** może być zainicjowana tylko raz, próba nadpisania wartości spowoduje błąd

```
public class FinalVariables {  
    public static void main(String[] args) {  
        final int number = 5;  
        number = 10; //błąd kompilacji  
    }  
}
```

Deklaracja:

final int liczba;

Inicjalizacja:

liczba = 5;

albo deklaracja + inicjalizacja:

final int liczba = 5;

Typ

Nazwa

Wartość

Ćwiczenie

```
Produkt1:  
Mleko Mlekowita 2.5  
Produkt2:  
Czekolada Wedel 2.19
```

Ćwiczenie*

Operacje matematyczne

Java daje nam możliwość wykonywania podstawowych operacji arytmetycznych na liczbach:

+ - dodawanie lub konkatencja napisów

- - odejmowanie

* - mnożenie

/ - dzielenie

% - modulo/reszta z dzielenia

++ - inkrementacja (+1)

-- - dekrementacja (-1)

inkrementacja i dekrementacja występują w wersji przed- i -przyrostkowej

$$2+3 = 5$$

$$5-10 = -5$$

$$4*8 = 32$$

$$9/2 = 4 \text{ (dzielenie całkowite)}$$

$$9.0/2.0 = 4.5$$

$$13\%5 = 3 \text{ (bo } 13/5 = 2 \text{ i zostaje 3 reszty)}$$

$$2++ \text{ daje } 3$$

$$2.5-- \text{ daje } 1.5$$

Promocja typów (demo)

Jeśli składniki działania arytmetycznego są różnych typów, następuje promocja do typu ogólniejszego ("większego")

$2 + 3 = 5;$

$\text{int} + \text{int} = \text{int};$

$2.5 + 3 = 5.5;$ //konwersja niejawna

$\text{double} + (\text{double})\text{int} = \text{double}$

Jeśli chcemy wymusić zmianę typu "większego", np. double na typ "mniejszy" np. int, możemy to zrobić poprzez konwersję.

$(\text{int})2.6 = 2$ //konwersja jawna

W przypadku konwersji "w dół" tracimy informację o części dziesiętnej.

Ćwiczenie

Ćwiczenie*

Operacje logiczne

Operatory

- `==` czy dwie wartości są równe
- `!=` czy dwie wartości są różne
- `>`, `>=`, `<`, `<=` większy / mniejszy
- `!` negacja, "nieprawda, że ..."
- `||` logiczna operacja "lub" / alternatywa
- `&&` logiczna operacja "i" / koniunkcja

Przykłady

`2 == 2` daje `true`

`2 != 2` daje `false`

`10 > 0` daje `true`

`!true` daje `false`

`true || false` daje `true`

`true && false` daje `false`

Operatory logiczne (demo)

<i>a</i>	<i>!a</i>	<i>a</i>	<i>b</i>	<i>a && b</i>	<i>a b</i>
true	false	false	false	false	false
false	true	false	true	false	true
		true	false	false	true
		true	true	true	true

Ćwiczenie