

PODSTAWY PROGRAMOWANIA W JAVA

dr inż. Michał Tomaszewski

katedra Metod Programowania
Polsko-Japońska Akademia Technik Komputerowych

ZAPISY LITERAŁÓW

W języku Java przez literał rozumiany jest reprezentowana w kodzie źródłowym wartość:

- typu prostego (primitive type),
- typu złożonego `String`,
- lub typu `null`.

Literały stanu logicznego:

- `true`
- `false`

Literały liczb całkowitych w zapisie:

Literały liczb całkowitych w zapisie:

- dziesiętnym

Literały liczb całkowitych w zapisie:

- dziesiętnym
- szesnatkowym

Literały liczb całkowitych w zapisie:

- dziesiętnym
- szesnastkowym
- osemkowym

Literały liczb całkowitych w zapisie:

- dziesiętnym
- szesnastkowym
- osemkowym
- binarnym

Literały liczb całkowitych w zapisie dziesiętnym:

Literały liczb całkowitych w zapisie dziesiętnym:

- ciąg cyfr rozpoczynający się wartością inną niż 0

Literały liczb całkowitych w zapisie dziesiętnym:

- ciąg cyfr rozpoczynający się wartością inną niż 0
- opcjonalnie występująca końcówka \mathbb{L} \mathbb{L}

Literały liczb całkowitych w zapisie osemkowym:

- ciąg cyfr rozpoczynający się wartością 0
- opcjonalnie występująca końcówka \mathbb{L} L

Literały liczb całkowitych w zapisie szesnatkowym:

- ciąg cyfr rozpoczynający się wartością `0x`
- opcjonalnie występująca końcówka `l` `L`

Literały liczb całkowitych w zapisie binarnym:

- ciąg cyfr rozpoczynający się wartością **0b**
- opcjonalnie występująca końcówka **l L**

Literał znakowy:

Literał znakowy:

- pojedynczy znak objęty apostrofem np: 'a', 'A' lub '0'

Literał znakowy:

- pojedynczy znak objęty apostrofem np: `'a'`, `'A'` lub `'0'`
- objęty apostrofem znak ukośnik (backslash) i następujący po nim znak specjalny lub kod znaku np: `'\t'`, `'\177'`

Literał znakowy:

- pojedynczy znak objęty apostrofem np: `'a'`, `'A'` lub `'0'`
- objęty apostrofem znak ukośnik (backslash) i następujący po nim znak specjalny lub kod znaku np: `'\t'`, `'\177'`
- objęty apostrofem znak ukośnik (backslash), następujący po nim znak u i kod znaku reprezentowany w UTF-16 np: `'\u03a9'`, `'\uFFFF'`

Literał liczb zmiennoprzecinkowych:

- 2.f .3f
- 3.14 1e-9d 1e127

INSTRUKCJE

Czym jest instrukcja?

Instrukcja jest poleceniem wydanym procesorowi komputera, przez program. source:internet

Instrukcja jest poleceniem wydanym procesorowi komputera, przez program. source:internet

Instrukcje dzielą się na **czynne** i **bierne**.

Instrukcją bierną jest

- instrukcja **deklaracyjna**

Instrukcją bierną jest

- instrukcja **deklaracyjna**

Natomiast instrukcjami czynnymi są:

- instrukcja **pusta**,

Instrukcją bierną jest

- instrukcja **deklaracyjna**

Natomiast instrukcjami czynnymi są:

- instrukcja **pusta**,
- instrukcja **grupująca**,

Instrukcją bierną jest

- instrukcja **deklaracyjna**

Natomiast instrukcjami czynnymi są:

- instrukcja **pusta**,
- instrukcja **grupująca**,
- instrukcja **wyrażeniowa**,

Instrukcją bierną jest

- instrukcja **deklaracyjna**

Natomiast instrukcjami czynnymi są:

- instrukcja **pusta**,
- instrukcja **grupująca**,
- instrukcja **wyrazeniowa**,
- instrukcja **warunkowa**,

Instrukcją bierną jest

- instrukcja **deklaracyjna**

Natomiast instrukcjami czynnymi są:

- instrukcja **pusta**,
- instrukcja **grupująca**,
- instrukcja **wyrażeniowa**,
- instrukcja **warunkowa**,
- instrukcje **iteracyjne i decyzyjne**,

Instrukcją bierną jest

- instrukcja **deklaracyjna**

Natomiast instrukcjami czynnymi są:

- instrukcja **pusta**,
- instrukcja **grupująca**,
- instrukcja **wyrażeniowa**,
- instrukcja **warunkowa**,
- instrukcje **iteracyjne i decyzyjne**,
- instrukcje **zaniechania i kontynuowania**.

Instrukcją bierną jest

- instrukcja **deklaracyjna**

Natomiast instrukcjami czynnymi są:

- instrukcja **pusta**,
- instrukcja **grupująca**,
- instrukcja **wyrażeniowa**,
- instrukcja **warunkowa**,
- instrukcje **iteracyjne i decyzyjne**,
- instrukcje **zaniechania i kontynuowania**.

Każda instrukcja jest zakończona **klamrą** albo **średnikiem**.

Instrukcja pusta ma postać:

;

Instrukcja grupująca ma postać

$$\{ \text{Ins} \text{ Ins } \dots \text{Ins} \}$$

w której każde **Ins** jest dowolną instrukcją albo jest **puste**.

Instrukcja grupująca ma postać

```
{ Ins Ins ... Ins }
```

w której każde `Ins` jest dowolną instrukcją albo jest **puste**.

```
{  
    System.out.println("Hello old friend");  
}
```

Instrukcja wyrażeniowa ma postać:

`exp;`

w której `exp` jest: **przypisaniem**, **wywołaniem**, **zwiększeniem** albo **zmniejszeniem**.

Instrukcja:

`Typ nazwa;`

deklaruje zmienną `nazwa` typu `Typ`.

Instrukcja:

```
Typ nazwa;
```

deklaruje zmienną **nazwa** typu **Typ**.

W szczególności instrukcja:

```
int age = 7;
```

deklaruje zmienną **age** typu **int**(całkowitego), którą podczas deklarowania zainicjowano wartością 7 reprezentowaną przez literał.

DZIĘKUJE