Java

wprowadzenie

typy danych, zmienne

Czego się dowiesz?

- Jakie typy danych wyróżniamy w Javie?
- Jak przechowywać informacje w programie?
- Jaki jest jest czas "życia" zmiennej?
- Jaki jest jej zakres "widoczności"?

Typy danych

Java jest językiem ze ścisłą kontrolą typów. Co to oznacza?

Dokładnie tyle, że jeśli powiemy, że coś ma być tekstem, to nim będzie tak długo dopóki tego nie usuniemy. Java mocno pilnuje typu, który określimy dla jakiejś danej.

Na początek poznamy 8 typów prosty. Będą to jedyne typy, których będziemy używać do momentu poznania obiektowości.

Typy całkowitoliczbowe

- byte
- short
- **int**
- long

Różnią się między sobą tylko tym, że przechowują różne zakresy wartości. Np. dla typu byte będą to wartości od -128 do 127. Dla long będą, to liczby od ok -9 to +9 trylionów. Najczęściej korzystamy z typu prostego int (od ok -2 do +2 miliardów).

Przykłady: 234234234, 12, 0, -5123

Typy zmiennoprzecinkowe

- float
- **double**

Różnią się między sobą tylko tym, że przechowują różne zakresy wartości. Są one bardzo duże i nie trzeba ich znać. Najczęściej spotkasz double.

Przykłady: 2.44, 62341.125, -623.12

Typ znakowy

- **char**

Reprezentuje pojedynczy znak. W Javie do reprezentacji znaków stosuje się standard Unicode.
https://en.wikipedia.org/wiki/List_of_Unicode_characters

Przykłady: 5, ?, Z, $

Typ logiczny

- **boolean**

Umożliwia przechowywanie jednej z dwóch wartości. True albo false. Oznacza, to oczywiście prawdę lub fałsz.

Przykłady: true, false

Zmienna

Zmienna jest podstawową jednostką przechowywania informacji w programie. Zmienne są przechowywane przez Javę na stosie (Stack). Jest to specjalna struktura pamięci zarządzana przez JVM.

Deklaracja zmiennej:
typ nazwa;

Inicjalizacja zmiennej:
nazwa = wartość;

Deklaracja mówi "potrzebuję zmiennej o określonej nazwie i typie", inicjalizacja mówi "przypisz zmiennej wskazaną wartość".

Przykładowe zmienne

```
int age;
age = 12;
double price = 23.21;
char firstLetterOfName = 's';
boolean isEven = false;
price = 44.92;
```

Zauważ, że deklarację oraz inicjalizację możemy zrobić w jednej linii. Wartość przecinkową w double rozdzielamy kropką. Znak dodajemy w apostrofach. Każda instrukcja kończy się średnikiem. Deklarację wystarczy wykonać jeden raz.

Ciekawostka o typie char

Typ znakowy może przechować znaki Unicode. Możemy je przypisywać nie tylko poprzez konkretny znak w apostrofie np. char znak = 'ś', ale też poprzez wartość dziesiętną tego znaku. Czyli np. char znak = 347; Oba podejścia są poprawne.

U+0158	Ř	344	Ř	Latin Capital Letter R with caron	0280
U+0159	ř	345	ř	Latin Small Letter R with caron	0281
U+015A	Ś	346	Ś	Latin Capital Letter S with acute	0282
U+015B	ś	347	ś	Latin Small Letter S with acute	0283
U+015C	Ŝ	348	Ŝ	Latin Capital Letter S with circumflex	0284
U+015D	ŝ	349	ŝ	Latin Small Letter S with circumflex	0285
U+015E	Ş	350	Ş	Latin Capital Letter S with cedilla	0286
U+015F	ş	351	ş	Latin Small Letter S with cedilla	0287

źródło: https://en.wikipedia.org/wiki/List_of_Unicode_characters

Zadanie 1

Napisz program przechowujący rok urodzenia oraz płeć osoby. Rok urodzenia to 2001, a płeć, to K. Wypisz przechowywane informacje na konsoli.

Zadanie 2

Napisz program przechowujący informację o tym czy osoba jest pełnoletnia, oraz przechowaj jej średnią ocen. W tym przypadku osoba jest pełnoletnia, a jej średnia to 3,5. Wypisz przechowywane informacje na konsoli.

*Zadanie 3

Napisz program, w którym przechowasz numer domu, numer mieszkania oraz stan konta. Założenia:

- numer domu, to liczba z przedziału od 1 do 1000
- numer mieszkania, to liczba z przedziału od 1 do 100
- stan konta, to liczba z przedziału od -1 999 999,99 do +1 999 999,99

Ustaw wartości, a następnie wyświetl w konsoli. Użyj jak najmniejszych typów, tak aby spełniały założenia.

*samodzielnie

Zasięg zmiennych

Dowiedzieliśmy się jak zadeklarować oraz zainicjalizować zmienną, ale w których częściach programu mogę z niej skorzystać?

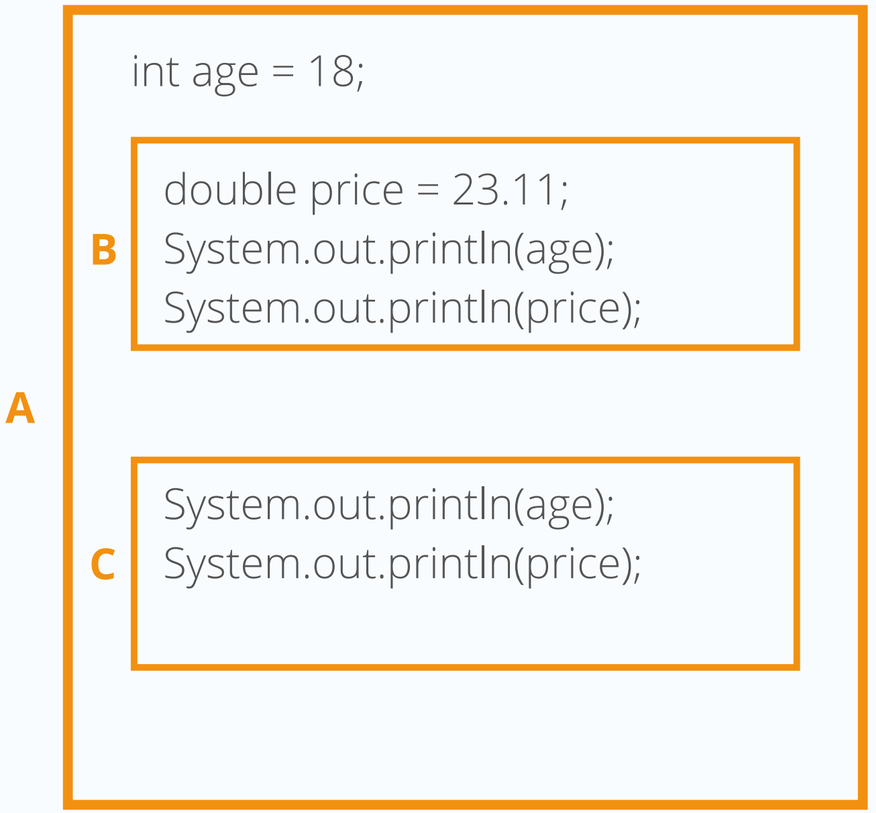
Zmienna jest dostępna w bloku, w którym została zadeklarowana oraz we wszystkich blokach zagnieżdżonych w stosunku do tego bloku.

Blok rozpoczyna się w miejscu otwarcia nawiasu klamrowego '{', a kończy w miejscu jego zamknięcia '}'.

Zasięg zmiennych

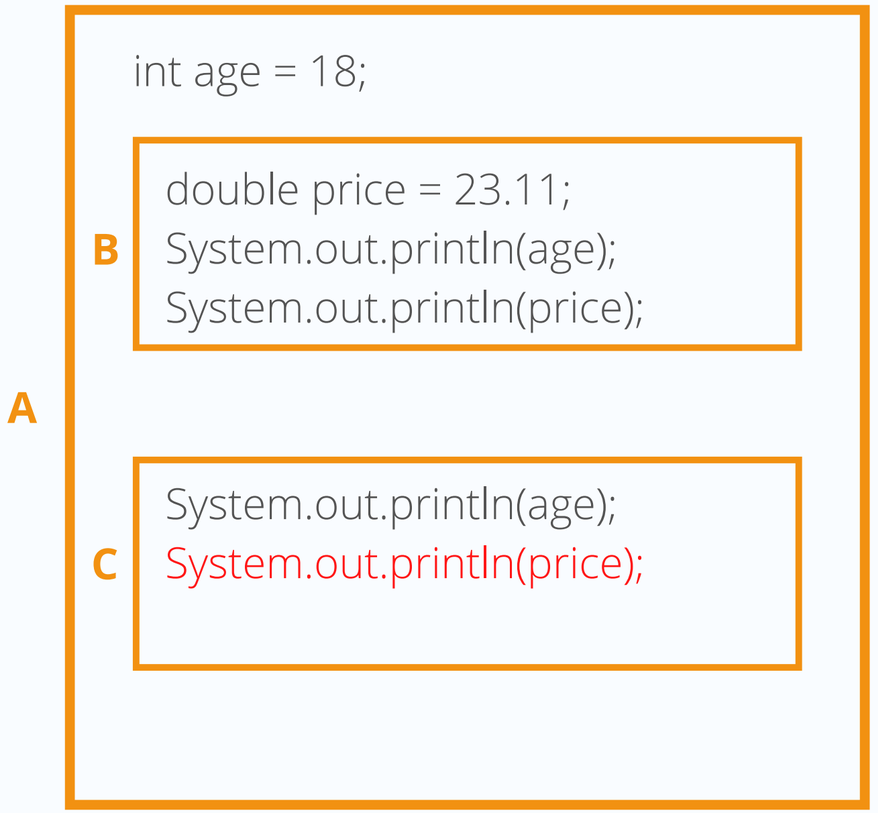
Spójrz na rysunek po prawej. Niech każdy prostokąt oznacza blok, a więc klamry { }. Mogą one dotyczyć klasy, metody, instrukcji nie ma to znaczenia. Istotna jest kolejność zagnieżdżenia

Jak zachowa się ten program?



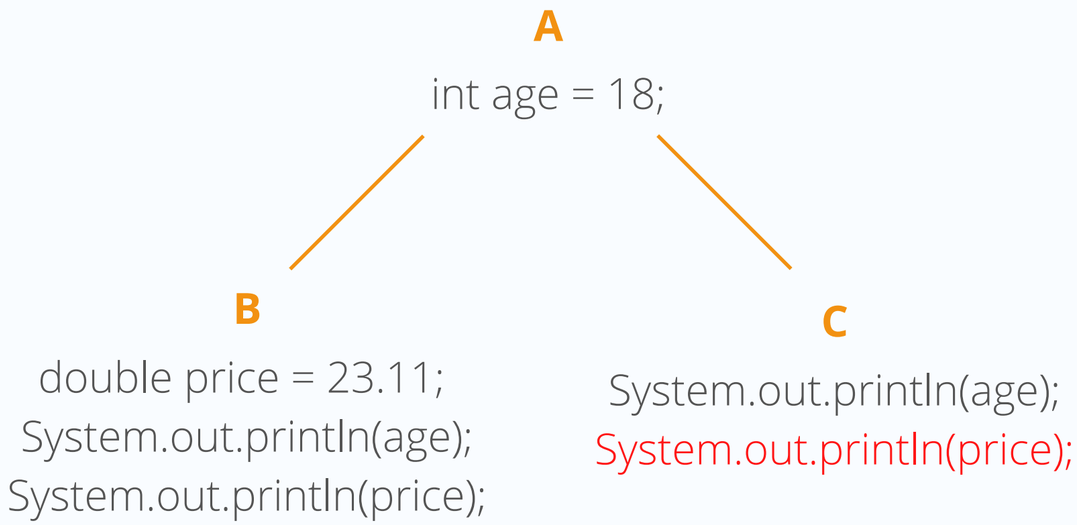
Zasięg zmiennych

- blok A deklaruje zmienną age
- blok B deklaruje zmienną price
- blok B wyświetla zmienną age, udaje mu się, ponieważ blok B jest wewnątrz A
- blok B wyświetla zmienną price, udaje mu się, ponieważ price zostało zadeklarowane w B
- blok C wyświetla zmienną age, udaje mu się, ponieważ blok C jest wewnątrz A
- blok C wyświetla zmienną price, nie udaje się, ponieważ price nie zostało zadeklarowane w tym samym bloku lub bloku nadrzędnym.



Zasięg zmiennych

Jeśli nie do końca trafił do Ciebie rysunek z blokami, to spróbujmy rozrysować go w postaci drzewa.



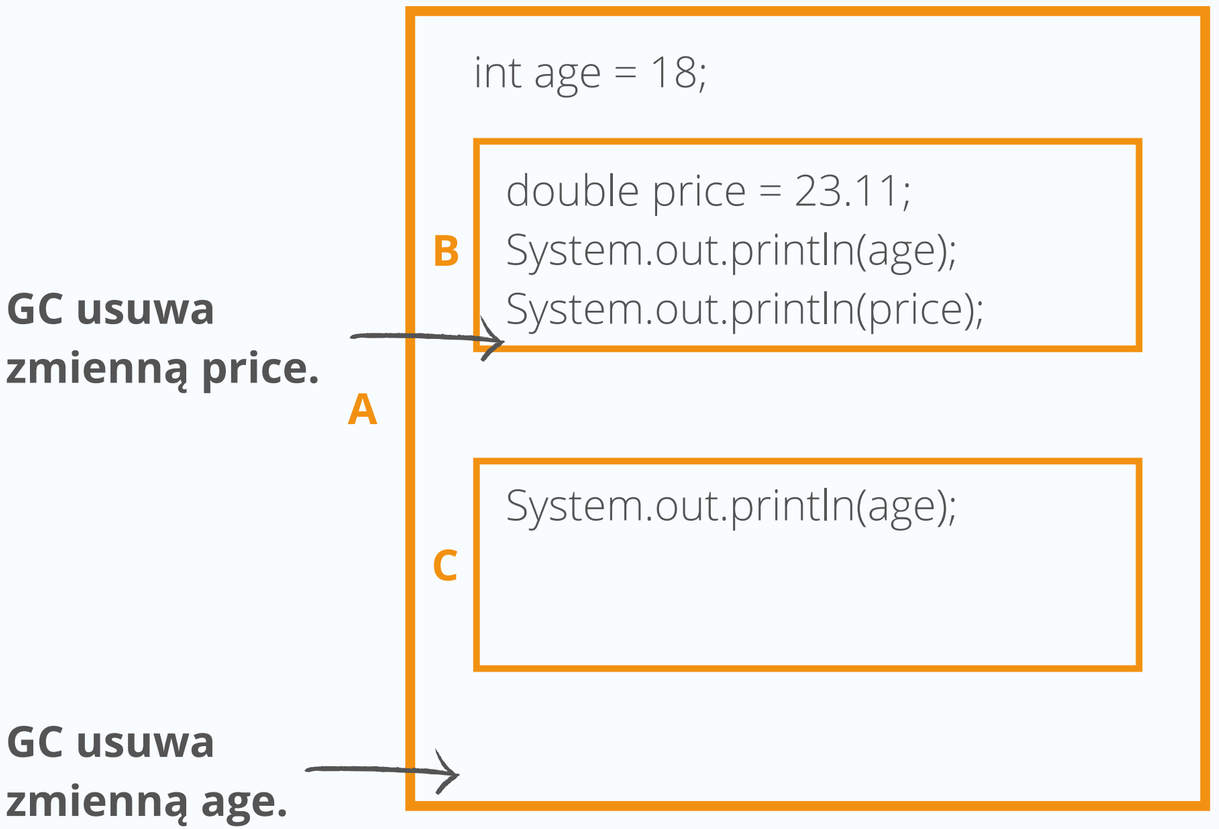
Czas życia zmiennych

W językach takich jak C++ należało samodzielnie czyścić, usuwać zmienną zwalniając pamięć. W Javie mamy dostępny mechanizm, który robi to za nas automatycznie. Nazywa się **garbage collector.**

Garbage collector (GC) usuwa zmienne z pamięci w momencie, gdy wychodzimy z zasięgu jej widoczności. Czyli na końcu bloku, którego została zadeklarowana.

Spójrzmy raz jeszcze na rysunek z blokami...

Czas życia zmiennych



Podsumowanie

W Javie mamy 8 typów prostych: byte, short, int, long, float, double, char i boolean.

Zmienne tworzymy określając im typ oraz podając nazwę. Java jest ściśle typowana, więc typ zmiennej będzie mocno pilnowany przez Javę.

Zasięg i czas życia zmiennej są ze sobą powiązane. Zmienna jest widoczna w bloku deklaracji oraz w blokach podrzędnych.
Garbage collector usuwa zmienną z pamięci, gdy wychodzi ze swego zakresu.

Dodatkowe materiały

Typy proste w Javie
https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html

Garbage Collector
https://www.baeldung.com/jvm-garbage-collectors

Zarządzania pamięcią w Javie
https://www.baeldung.com/java-stack-heap