



Въведение в Java



Ще разгледаме...

Вградени типове данни

Условия и разклонения

Итерация / Цикли

Низовете по-подробно

Масиви

Функции

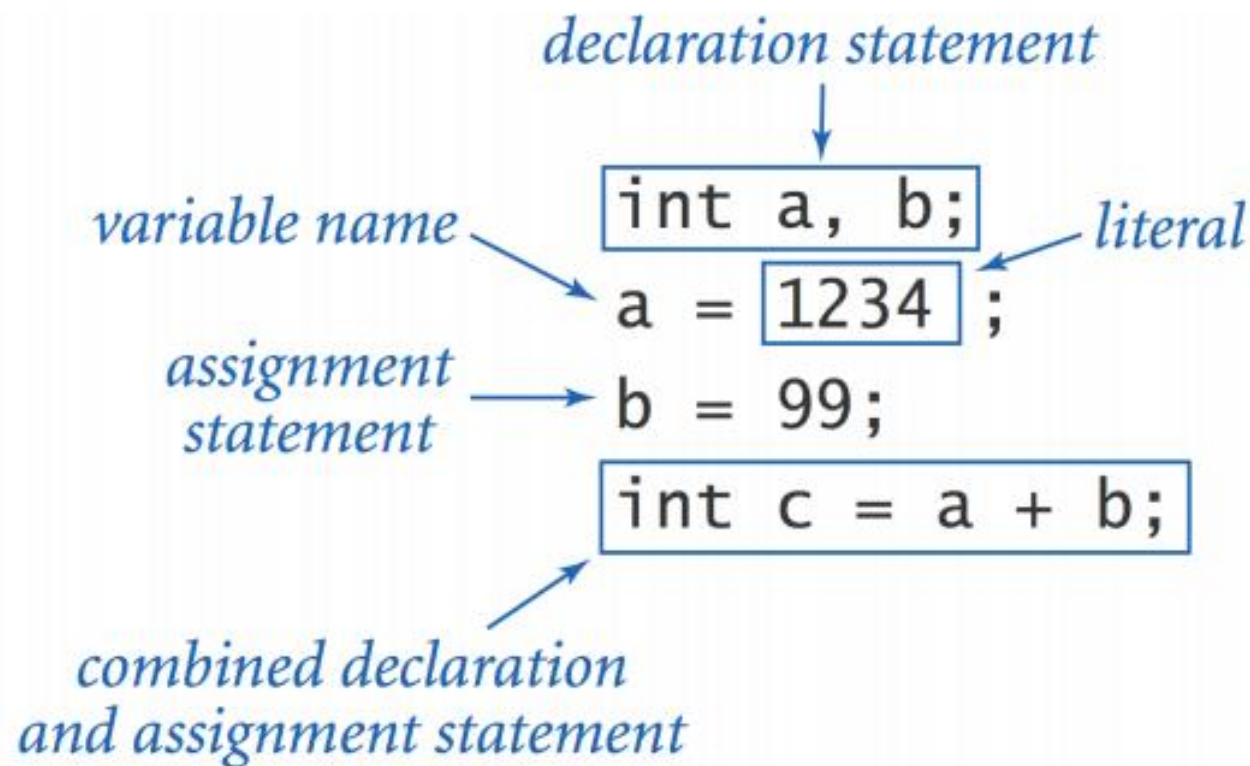
Вградени типове

Java е статично типизиран език → всички променливи трябва да бъдат декларирани преди да бъдат използвани. Декларацията включва името и типа.

```
int gear = 1;
```


Тип данни == множество стойности + операции върху тях

Променливи



Примитивни типове

Primitive type	Size	Minimum	Maximum
boolean	—	—	—
char	16 bits	Unicode 0	Unicode $2^{16}-1$
byte	8 bits	-128	+127
short	16 bits	-2^{15}	$+2^{15}-1$
int	32 bits	-2^{31}	$+2^{31}-1$
long	64 bits	-2^{63}	$+2^{63}-1$
float	32 bits	IEEE754	IEEE754
double	64 bits	IEEE754	IEEE754
void	—	—	—



range at full precision	precision*
$\pm 1.18 \times 10^{-38}$ to $\pm 3.4 \times 10^{38}$	approx. 7 decimal digits
$\pm 2.23 \times 10^{-308}$ to $\pm 1.80 \times 10^{308}$	approx. 15 decimal digits

Литерали

```
int i = 1; // int by default
```

```
long l = 1L; // L or l
```

```
double d = 0.1; // d or D is optional
```

```
double d2 = 1e-1; // same, in scientific
```

```
float f = 0.1; // will not compile, why?
```

```
char c = 'a';
```

```
String s = "cool";
```

Литерали

```
// The number 26, in decimal  
int decVal = 26;
```

```
// The number 26, in hexadecimal  
int hexVal = 0x1a;
```

```
// The number 26, in binary  
int binVal = 0b11010;
```

```
// The number 26, in octal  
int octVal = 032;
```

Литералите: какво ново от Java 7 насам?

Числови литерали с подчертавка

```
int thousand = 1_000;
```

```
int million = 1_000_000;
```

```
long magic = 0xCAFE_BABE;
```

Числови литерали в двоична бройна система

```
int one = 0b1;
```

```
int mask = 0b1010_1010_1010;
```


Стойности по подразбиране

Компилаторът **не** присвоява стойности по подразбиране на неинициализираните локални променливи!

Data Type	Default Value (for fields)
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d
char	'\u0000'
String (or any object)	null
boolean	false

Конвертиране на типовете

- ИмPLICITно: без загуба на точност; с низ
- ЕКСПЛИЦИТНО: чрез cast

<i>expression</i>	<i>expression type</i>	<i>expression value</i>
"1234" + 99	String	"123499"
(int) 2.71828	int	2
11 * 0.3	double	3.3
(int) 11 * 0.3	double	3.3
11 * (int) 0.3	int	0
(int) (11 * 0.3)	int	3

Защо ни трябват типове?

- За да ни помага компилаторът



През 1996, ракетата Ариана 5 експлодира след излитане поради софтуерна грешка в конвертирането на типове (опит да „набута“ 64-битово число в 16 бита)

Оператори

Operators

postfix

unary

multiplicative

additive

shift

relational

equality

bitwise AND

bitwise exclusive OR

bitwise inclusive OR

logical AND

logical OR

ternary

assignment

Precedence

expr++ *expr*--

++*expr* --*expr* +*expr* -*expr* ~ !

* / %

+ -

<< >> >>>

< > <= >= instanceof

== !=

&

^

|

&&

||

? :

= += -= *= /= %= &= ^= |= <<= >>= >>>=

Scoping

```
{  
    int x = 12;  
    // Only x available  
  
    {  
  
        int q = 96;  
        // Both x & q available  
  
    }  
  
    // Only x available  
    // q is "out of scope"  
  
}
```

Низове

`String // immutable`

`StringBuilder // mutable, fast, single-threaded`

`StringBuffer // mutable, slower, thread-safe`

Низове - обхождане

```
String s = "Firebird";
```

```
char ic = s.charAt(i);
```

```
char[] ca = s.toCharArray();
```

```
for (int i = 0; i < ca.size(); i++) {...}
```

```
for (c in ca) {...} // enhanced for-loop
```

```
String sorted = Arrays.sort(ca).toString(); // "Fbdeir"
```

Низове

Може да конкатенираме низове с оператора ,+‘

Ако аргумент на ,+‘ е нещо различно от низ, той се конвертира към низ

```
String str1 = "Current";
```

```
String str2 = str1 + " year is " + 2016;
```


Разбиване на поднизове по разделител

```
String str1 = "Current year is 2017";
```

```
String[] sa = s.split(" "); // разделител - интервал
```

```
// sa[0] е „Current“, sa[1] е „year“, sa[2] е „is“, sa[3]  
е „2017“
```

```
int year = Integer.parseInt(sa[3]); // year == 2017
```

Вход / изход

Писане на стандартния изход

```
System.out.println("Something printed on console");
```

Четене от стандартния вход

```
import java.util.Scanner; // това се слага над/преди дефиницията на класа
```

```
...
```

```
Scanner sc = new Scanner(System.in);
```

```
String lineRead = sc.nextLine();
```

Булеви изрази

`true` и `false`

за разлика от C/C++, не може да ползвате число вместо булев израз

Булеви логически оператори

A	B	A B	A&B	A^B	!A
false	false	false	false	false	true
true	false	true	false	true	false
false	true	true	false	true	true
true	true	true	true	false	false

| the OR operator

& the AND operator

^ the XOR operator

! the NOT operator

|| the short-circuit OR operator

&& the short-circuit AND operator

== the EQUAL TO operator

!= the NOT EQUAL TO operator

if-else

```
if (boolean_expression) {  
    statement  
}
```

```
if (boolean_expression) {  
    statement  
} else {  
    statement  
}
```

Операторът ? :

```
condition ? statement1 : statement2;
```

// единственият тринарен оператор в Java.
// Еквивалентно е на

```
if (condition) {  
    statement1  
  
} else {  
    statement2  
  
}
```

Итерация

```
while (boolean_expression) {  
    statement  
}
```

Итерация

```
do {  
    statement  
} while (boolean_expression);
```


Итерация

```
for (initialization; boolean_expression; step) {  
    statement  
}
```

Безусловно разклонение на логиката

`return [value]`

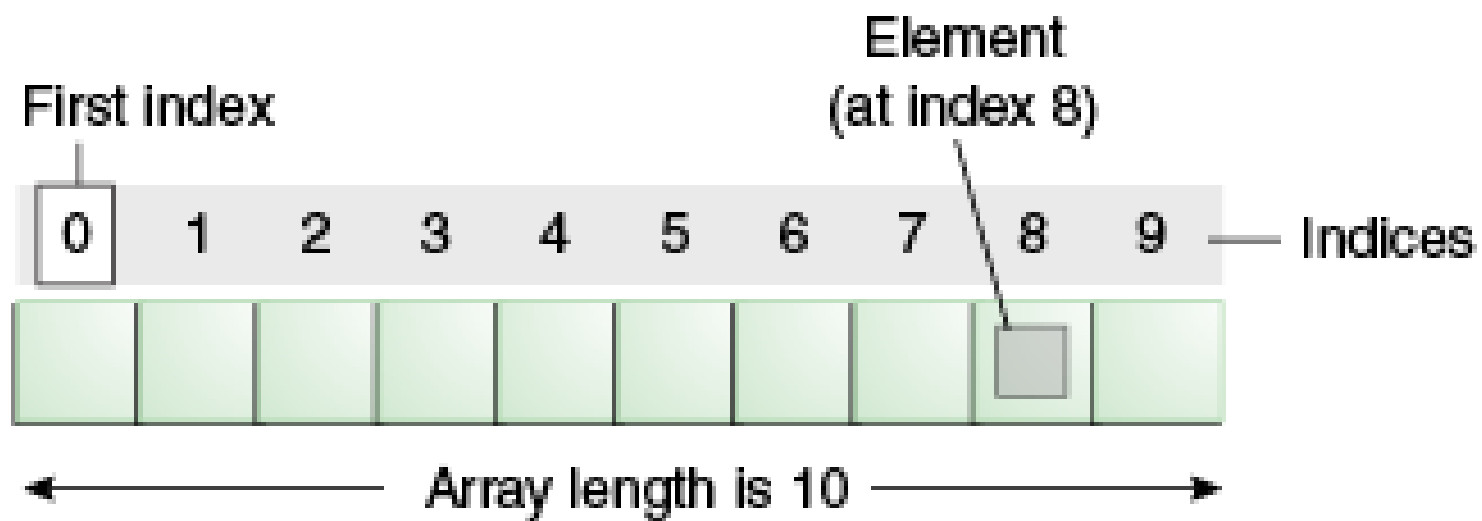
`break [label]`

`continue [label]`

Switch

```
switch (selector) {  
    case value1 : statement; break;  
    case value2 : statement; break;  
    case value3 : statement; break;  
  
    // ...  
  
    default: statement;  
  
}
```

Масиви



Масиви

```
int[] a; // preferred syntax
```

```
int a[];
```

Декларация – не се
заделя памет за
елементите на масива

```
int[] a = {1, 2, 3, 4}; // explicit initialization  
                        // can be done only during  
                        // declaration
```

```
int[] b = new int[7];
```

```
b.length;
```

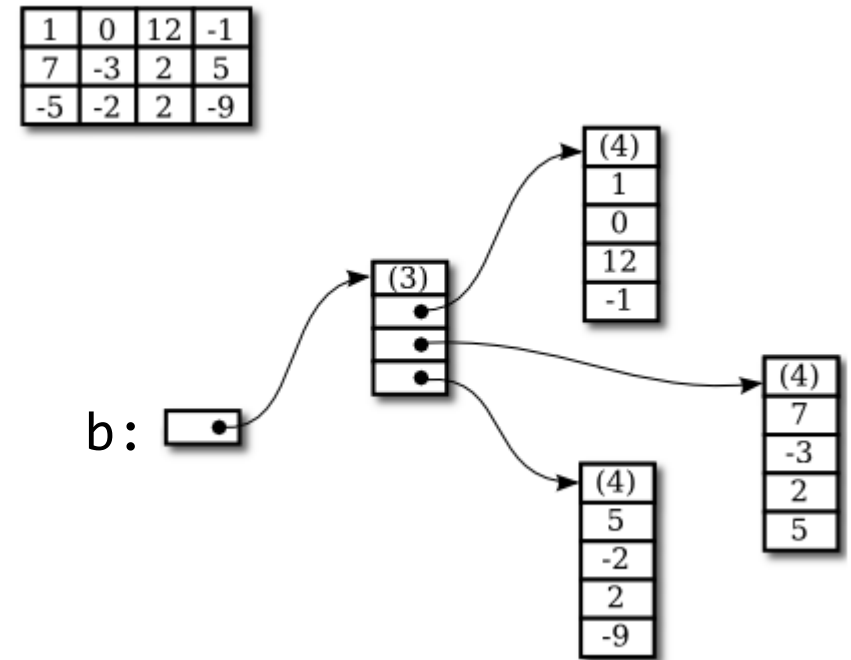
Инициализация – заделя
се памет за елементите
на масива

Масивите от примитивни типове се инициализират автоматично със стойността по подразбиране на съответния тип.

Многомерни масиви

```
int[][] a;  
a = new int[3][4];
```

```
int[][] b = { { 1, 0, 12, -1 },  
              { 7, -3, 2, 5 },  
              { -5, -2, 2, -9 }  
            };
```



Многомерни масиви

```
double[][] matrix = new double[7][];  
// rows have not yet been created!  
  
for (int i = 0; i < 7; i++) {  
    // Create row i with i + 1 elements.  
    matrix[i] = new double[i+1];  
}
```

Стандартни операции с масиви

```
System.arraycopy(from_arr, offset_from,  
to_arr, offset_to, num_elements);
```

```
Arrays.equals(arr1, arr2);
```

```
Arrays.fill(arr, value);
```

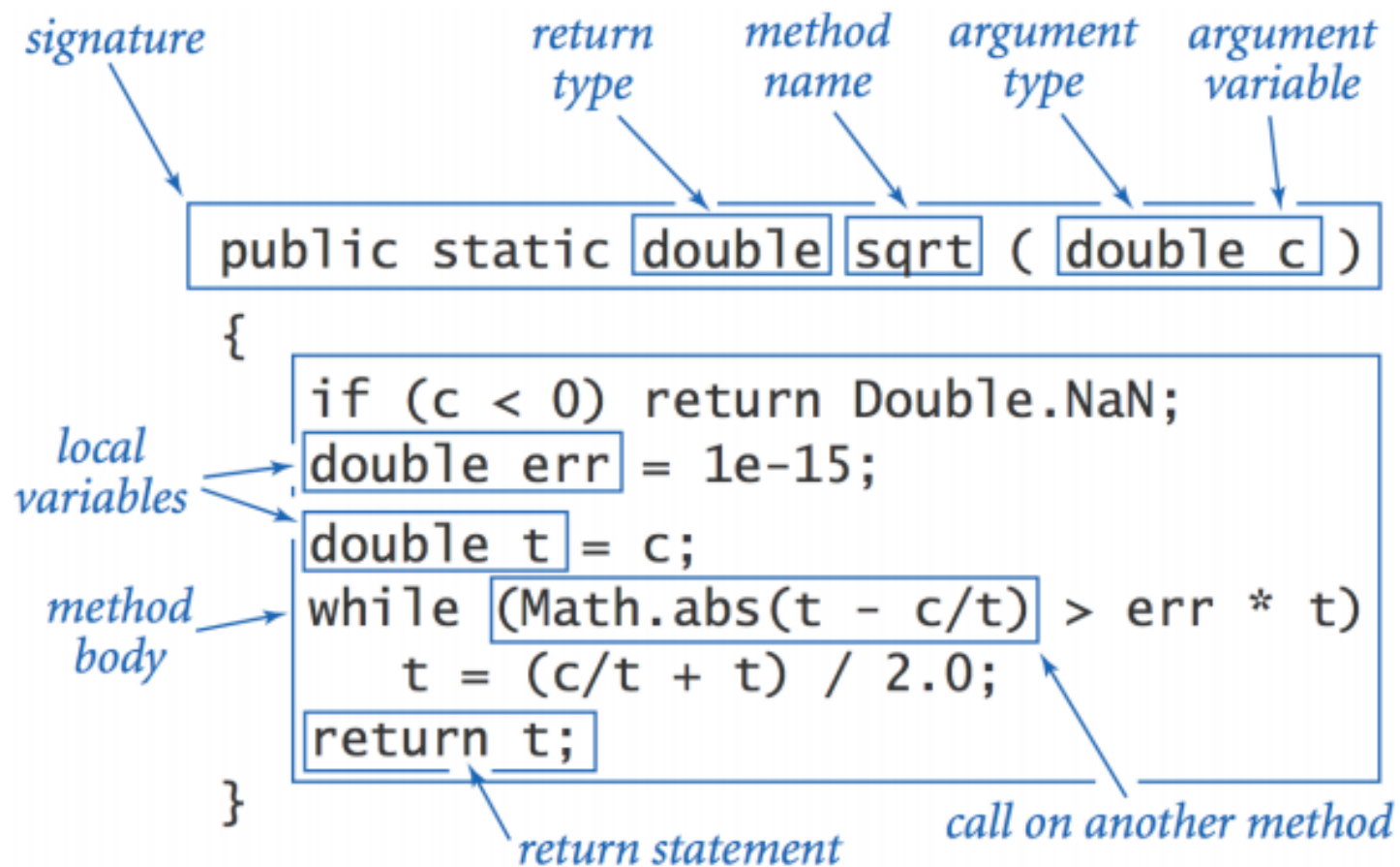
```
Arrays.toString(arr);
```


Стандартни операции с масиви

```
Arrays.sort(arr);
```

```
Arrays.sort(a, Collections.reverseOrder());
```

Функции



ПОЛЕЗНИ ЧЕТИВА

- [*Thinking in Java*](#)
- [Effective Java](#)
- [*Highlights of Technology Changes in Java SE 7*](#)
- [Learning the Java Language](#)

Сегга да пробваме!

