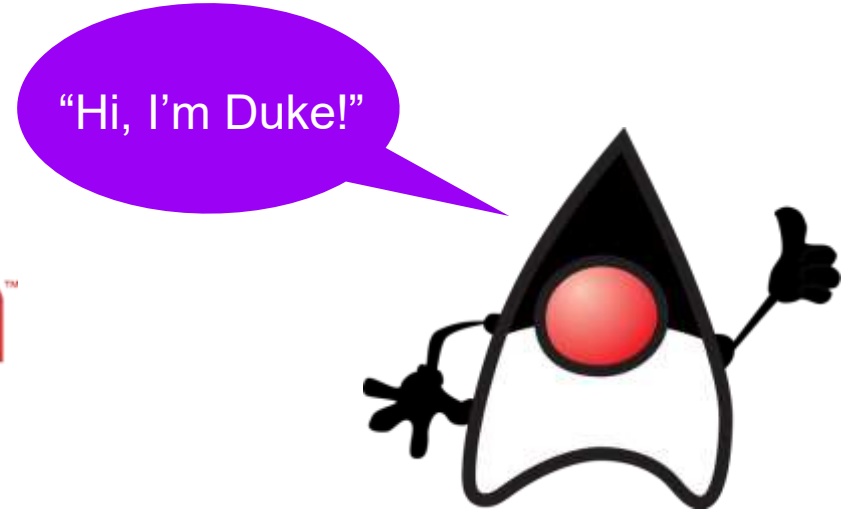
Въведение в Java

Езикът Java

- Създаден през 1995 от James Gosling (Sun Microsystems)
- Актуална версия: Java 12 (released 19.03.2019)



Езикът Java

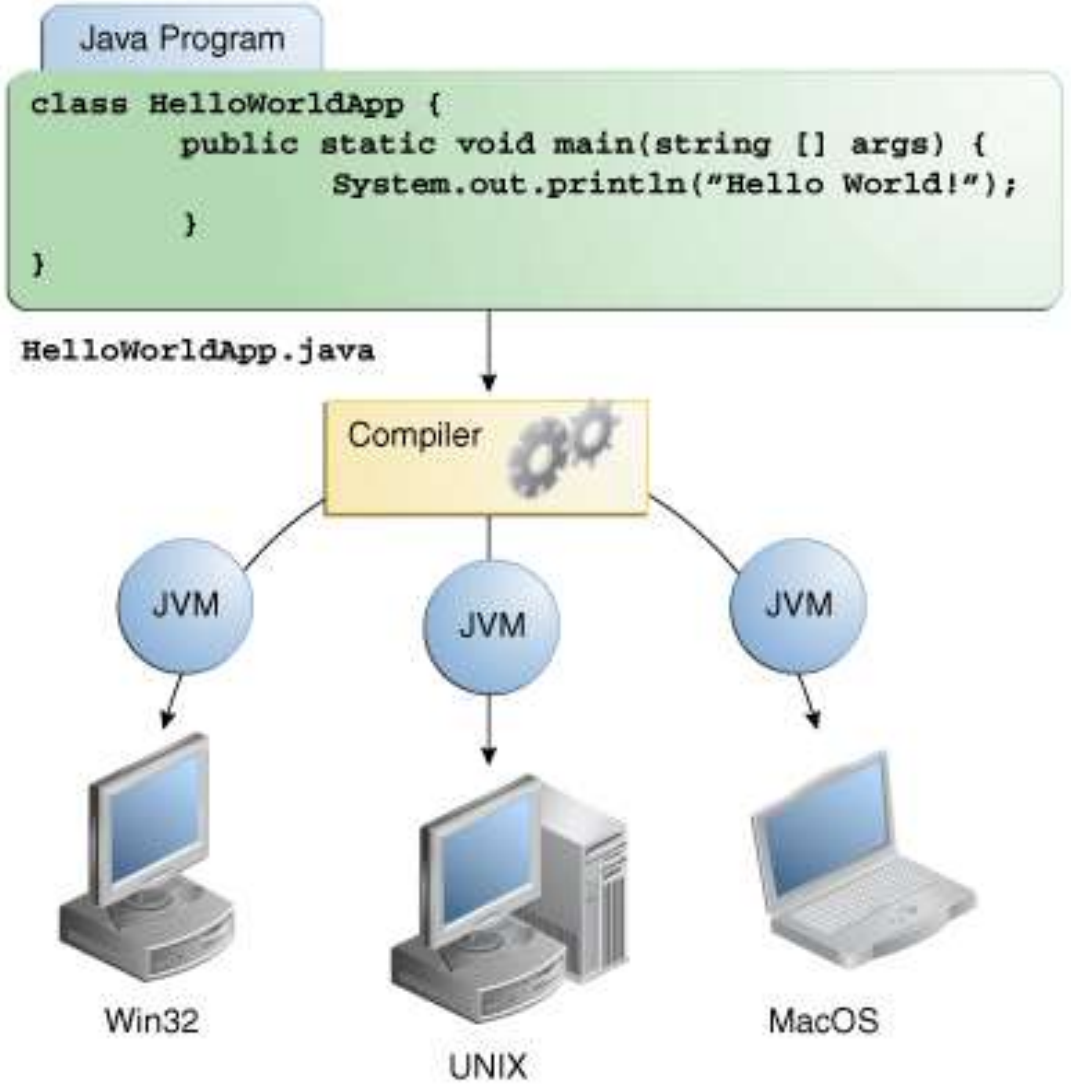
- Обектно-ориентиран
- Статично компилируем
- Със C/C++ синтаксис
- “Write once, run anywhere”

Hello world!

```
public class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

String args[]
String... args

Стандартно Java приложение



Java виртуалната машина (JVM)

- Интерпретира и изпълнява byte код инструкции
- Компилира по време на изпълнението byte кода до машинен код
- Заделя памет за оперативните данни
- Автоматично изчиства паметта
- Зарежда класове
- Стартира нишки
- Взаимодейства с операционната система

Формат на .class файла

0xCAFE_BABE

Java SE 12 = 0x38
Java SE 11 = 0x37
Java SE 10 = 0x36

```
ClassFile {
    u4              magic;
    u2              minor_version;
    u2              major_version;
    u2              constant_pool_count;
    cp_info         constant_pool[constant_pool_count-1];
    u2              access_flags;
    u2              this_class;
    u2              super_class;
```

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	CA	FE	BA	BE	00	00	00	35	00	4D	07	00	02	01	00	0E
00000010	53	74	72	69	6E	67	41	6E	61	6C	79	7A	65	72	07	00

```
    u2              methods_count;
    method_info     methods[methods_count];
    u2              attributes_count;
    attribute_info  attributes[attributes_count];
}
```

Ще започнем с Java от…

Вградените типове данни

Условия и разклонения

Итерация / Цикли

Низове

Масиви

Функции

Вградени типове

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

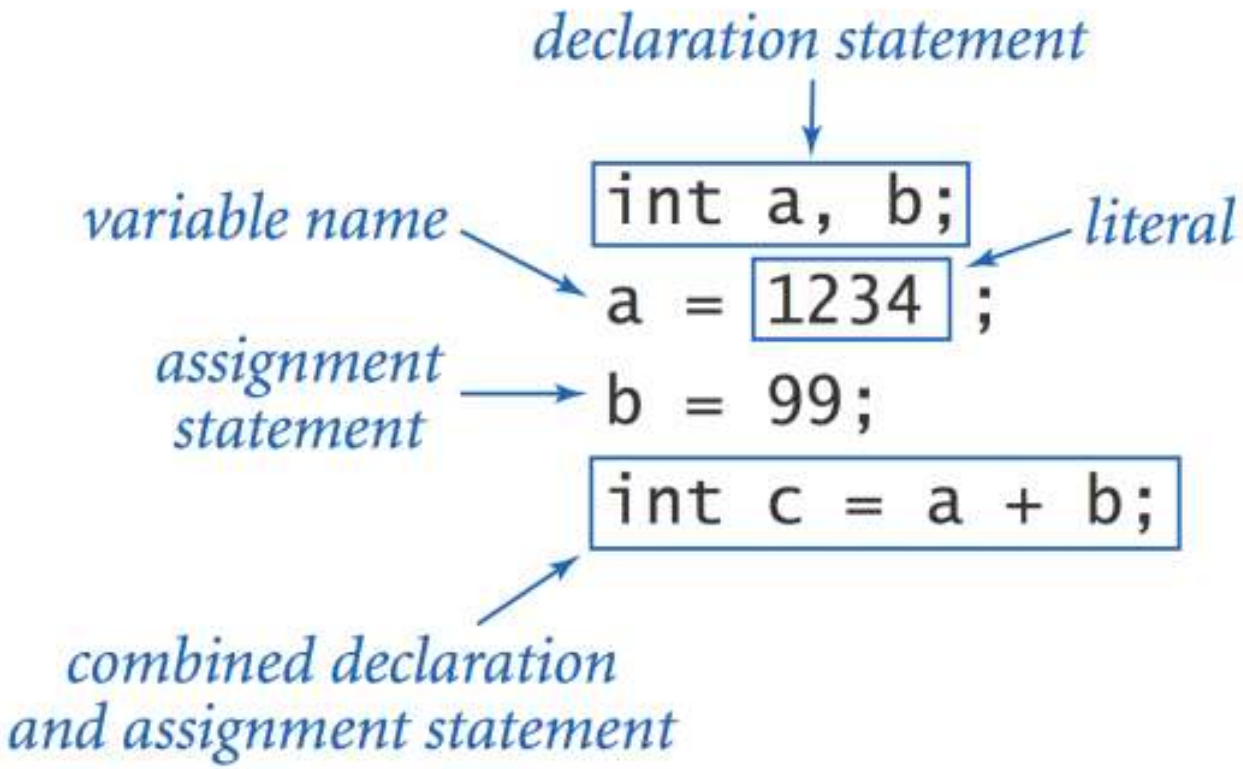
```
int gear = 1;
```

Тип данни == множество стойности + операции върху тях

*


```
var s = "FMI rulez"; // local variable type inference
```

Променливи


declaration statement
int a, b;
variable name
literal
a = 1234 ;
assignment statement
b = 99;
int c = a + b;
combined declaration and assignment statement

Примитивни типове

Keyword	Type	Example
boolean	true or false	true
byte	8-bit integral value	123
short	16-bit integral value	123
int	32-bit integral value	123
long	64-bit integral value	123
float	32-bit floating-point value	123.45f
double	64-bit floating-point value	123.456
char	16-bit Unicode value	'a'

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 notation, 1x10^-1

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;
```

Числови литерали с подчертавка

```
int thousand = 1_000;

int million = 1_000_000;

long magic = 0xCAFE_BABE;

int mask = 0b1010_1010_1010;
```

Конвертиране на типовете

- Имплицитно: „разширяване“ към по-голям тип; към низ

byte → short → int → long → float → double

от int или long към float, или от long към double
може да доведе до загуба на точност поради закръгляване

- Експлицитно: „свиване“, чрез cast

expression	*expression type*	*expression value*
"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	Precedence
postfix	*expr*++ *expr*--
unary	++*expr* --*expr* +*expr* -*expr* ~ !
multiplicative	* / %
additive	+ -
shift	<< >> >>>
relational	< > <= >= instanceof
equality	== !=
bitwise AND	&
bitwise exclusive OR	^
bitwise inclusive OR	|
logical AND	&&
logical OR	||
ternary	? :
assignment	= += -= *= /= %= &= ^= |= <<= >>= >>>=

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.length; i++) { }
for (char c : ca) { } // enhanced for-loop, a.k.a. for-each

Arrays.sort(ca);
var sorted = String.valueOf(ca); // “Fbdeiirr”
```

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 ” + 2018;
```

Разбиване на низ на поднизове по разделител

```
String str1 = “Current year is 2018”;

String[] sa = str1.split(“ “); // разделител – интервал

// sa[0] е „Current“, sa[1] е “year”, sa[2] е “is”, sa[3]
е „2018“

int year = Integer.parseInt(sa[3]); // year == 2018
```

Вход / изход

Писане на стандартния изход

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

Четене от стандартния вход

```
import java.util.Scanner; // това се слага над 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 изпълнението

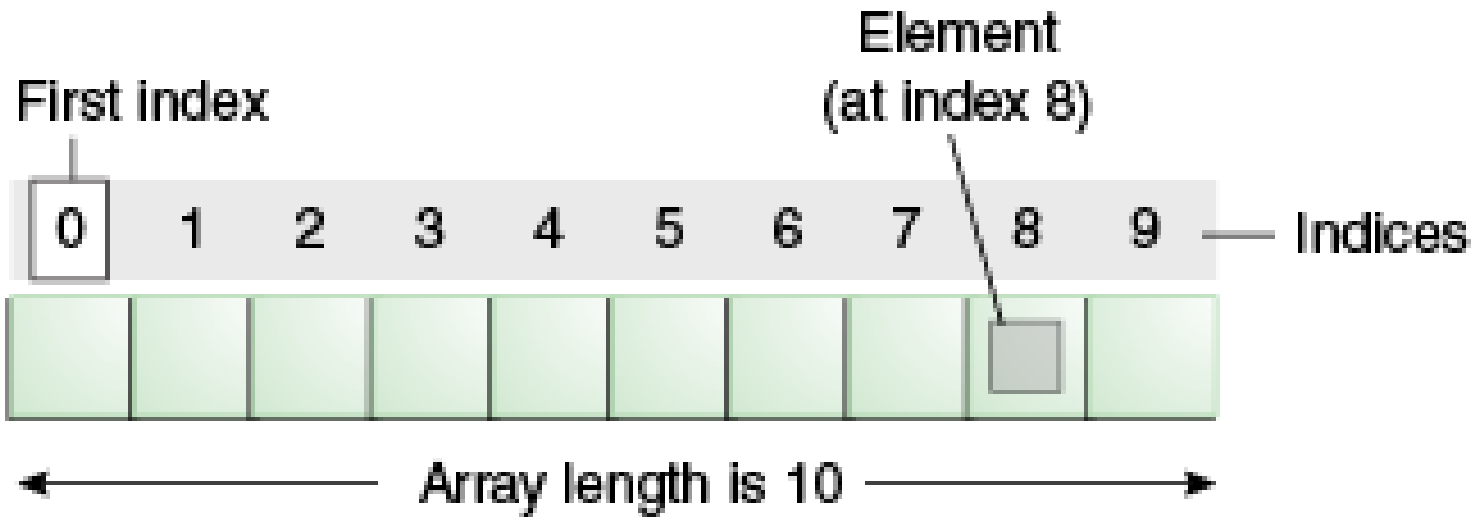
```
return [value]

break [label]

continue [label]
```

No “goto” ← keyword reserved but not used

Switch

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

Масиви


Element
(at index 8)
First index
0
1
2
3
4
5
6
7
8
9
Indices
Array length is 10

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
```

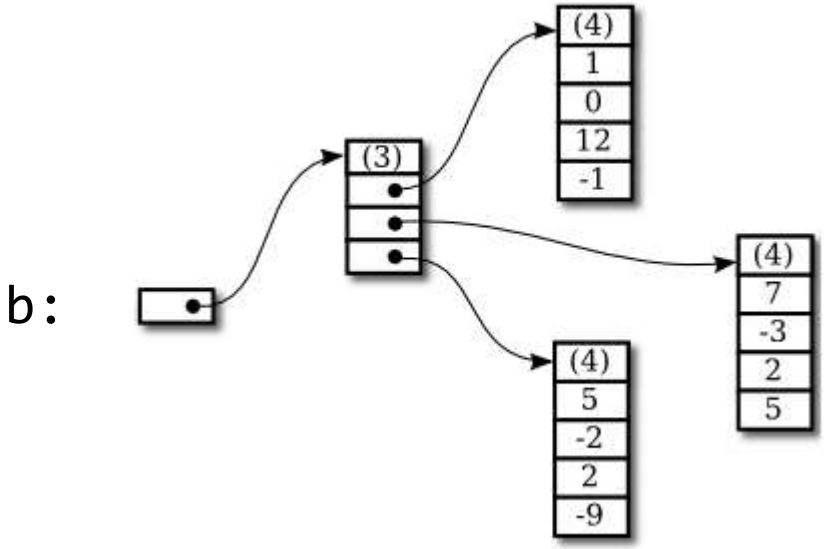
Инициализация – заделя се памет за елементите на масива

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

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); // копиране


Arrays.equals(arr1, arr2); // проверка за равенство
Arrays.fill(arr, value); // запълване с константна стойност
Arrays.toString(arr); // конвертиране на масив към низ
```

Стандартни операции с масиви

```
Arrays.sort(arr); // сортиране
Arrays.sort(a, Collections.reverseOrder()); // в обратен ред
```

Функции

signature
return type
method name
argument type
argument variable

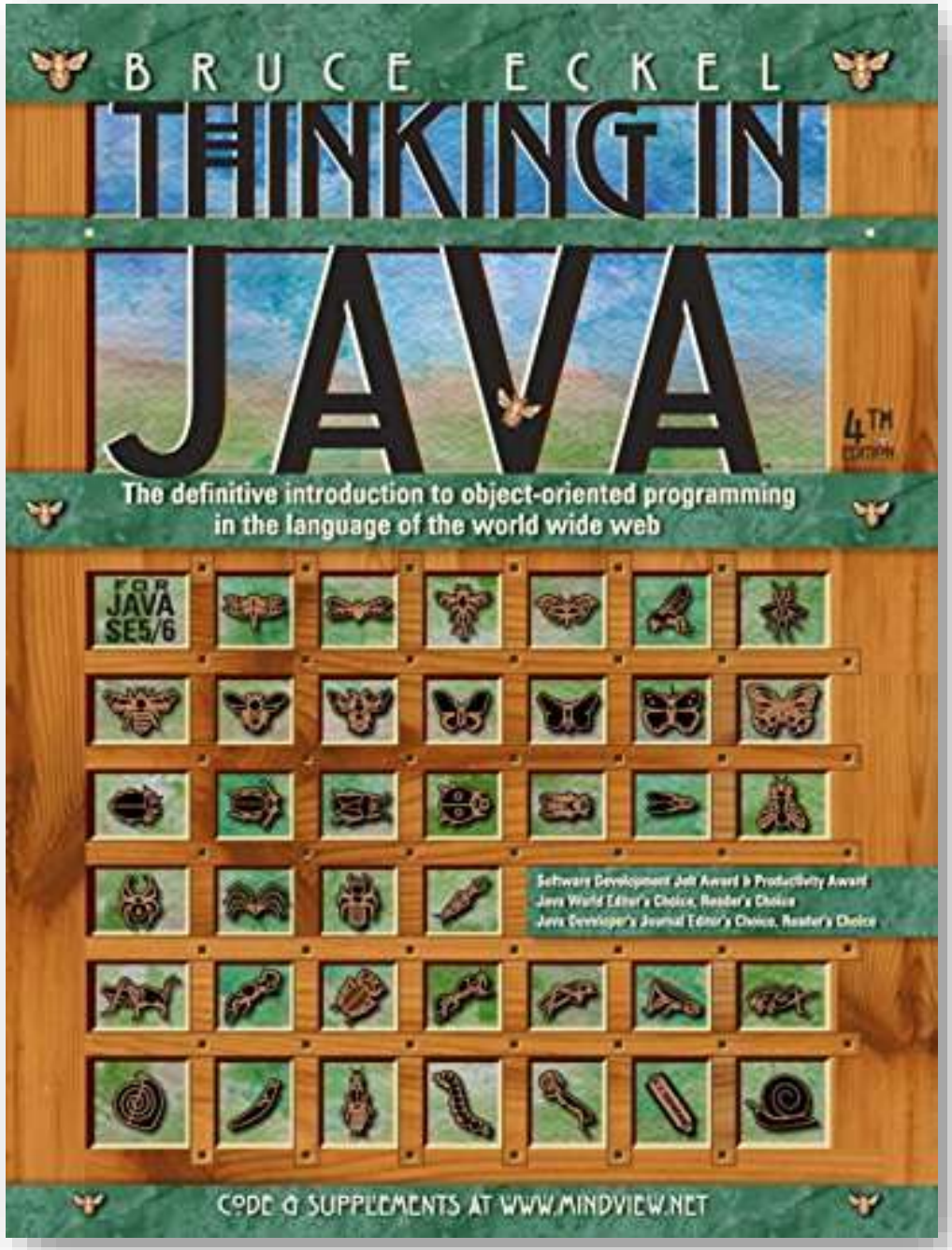
```
public static double sqrt ( double c )
{
    if (c < 0) return Double.NaN;
    double err = 1e-15;
    double t = c;
    while (Math.abs(t - c/t) > err * t)
        t = (c/t + t) / 2.0;
    return t;
}
```

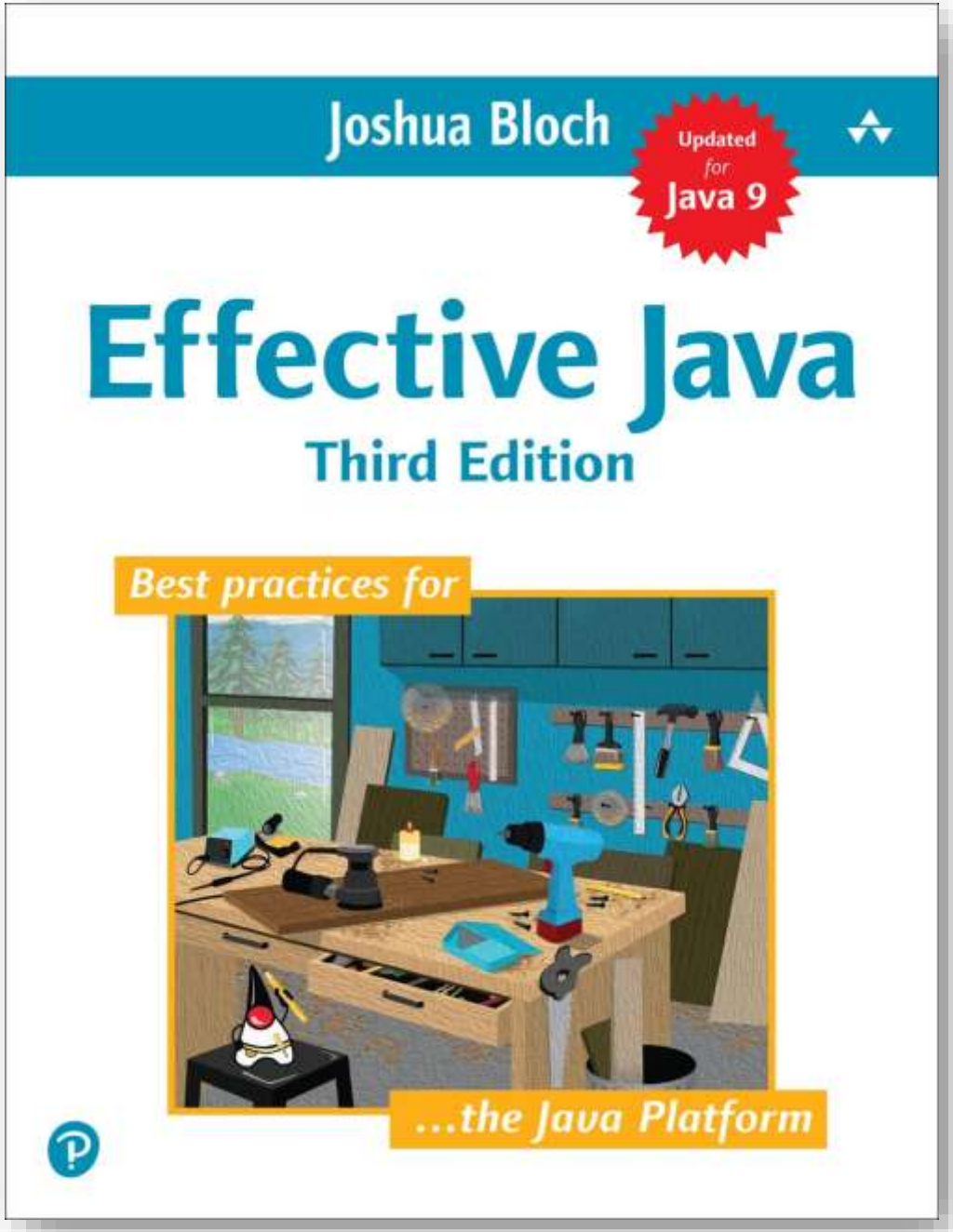
local variables
method body
return statement
call on another method

Въпроси?

ИЗПОЛЗВАНА (И ПРЕПОРЪЧВАНА) ЛИТЕРАТУРА

- Thinking in Java
- Effective Java
- Oracle OCA Java SE 8 Programmer I Study Guide
- Learning the Java Language


BRUCE ECKEL
THINKING IN
JAVA
4TH EDITION
The definitive introduction to object-oriented programming
in the language of the world wide web
FOR JAVA SE5/6
Software Development Jolt Award & Productivity Award
Java World Editor's Choice, Reader's Choice
Java Developer's Journal Editor's Choice, Reader's Choice
CODE & SUPPLEMENTS AT WWW.MINDVIEW.NET


Joshua Bloch
Updated for Java 9
Effective Java
Third Edition
Best practices for
...the Java Platform

Jeanne Boyarsky and Scott Selikoff

OCA

Oracle Certified Associate Java® SE 8 Programmer I

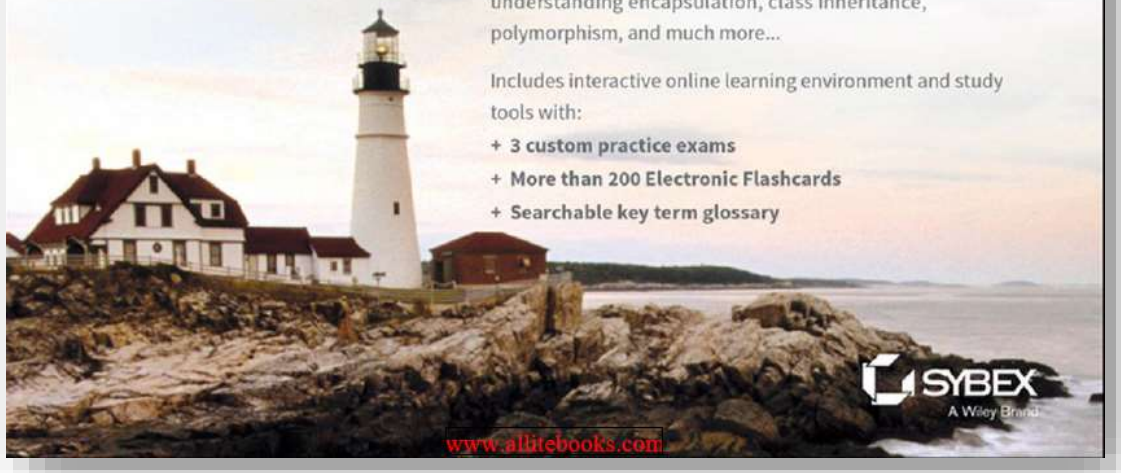
STUDY GUIDE

EXAM 1Z0-808

Covers 100% of exam objectives, including developing Java applications, becoming proficient in Java data types, mastering operators and decision control structures, understanding encapsulation, class inheritance, polymorphism, and much more...

Includes interactive online learning environment and study tools with:

+ **3 custom practice exams**
+ **More than 200 Electronic Flashcards**
+ **Searchable key term glossary**



SYBEX
A Wiley Brand

www.allitebooks.com

Сега да пробваме!