Objektovo orientované programovenie

2. cvičenie

Vladislav Novák

Dátové typy

- Premenné v Jave majú definované dátový typy
- Rozdelenie
  - primitívne dátové typy
  - referenčné dátové typy
    - pozor na kopírovanie a porovnavanie referenčných typov!

Primitívne typy

- byte – 8 bitový, <-128, 127>
- short – 16 bitov, <-32 768, 32 767>
- int – 32 bitov, $<-2^{31}, 2^{31}-1>$
- long – 64 bitov, $<-2^{63}, 2^{63}-1>$
- float – 32 bitov, plávajúca desatinná čiarka, IEEE 754
- double – 64 bitov, plávajúca desačinná čiarka, IEEE 754
- char – 16 bitov, Unicode
  - unicode môže používať rôzne druhy kódovania
- boolean = {true, false},
  - pamäťová veľkosť nie je presne definovaná

Primitívne číselné typy sú znamienkové.
Celečíselné typy používajú doplnkový kód

https://docs.oracle.com/javase/tutorial/java/nutsandbolts/datatypes.html

Primitívne typy

```
int pocet = 10;
double vyska = 1.5;
```

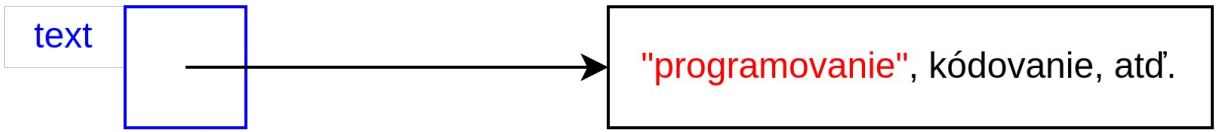
Referenčné typy

- String
  - Špeciálna podpora jazyka
  - inštancie sú nemenné (konštanty), ak chceme zmeniť hodnotu, musíme vytvoriť nový objekt
    - vhodné pre optimalizáciu
  - Ak potrebujeme meniť hodnotu textu, tak môžme použiť StringBuilder
- Color
  - inštancie sú nemenné
- Math
  - pre matematické operácie
- Arrays
  - pre polia
- Obálkové triedy (wrappery) základných typov
- …..

Primitívne vs referenčné typy

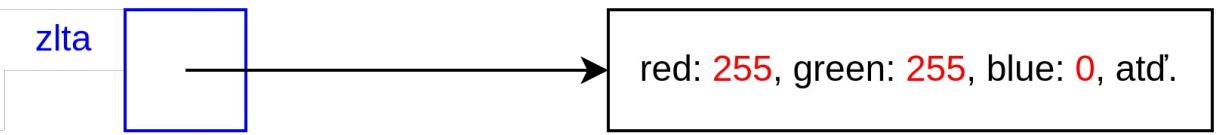
```
int dlzka = 10;
```


```
String text = "programovanie";
```

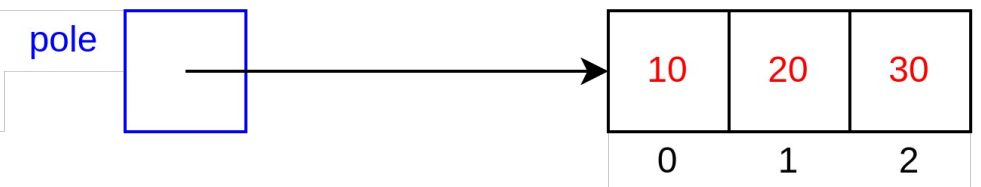



```
Color zlta = new Color(255, 255, 0);
```

https://www.csfieldguide.org.nz/en/interactives/rgb-mixer/




```
int [] pole = {10, 20, 30};
```



Literály

```java
int a = 1;
long b = 1L;
int c = 1_000_000;

int d = 0xFF;
System.out.println(d);

int e = 0b1100; // 8 + 4 + 0 + 0 = 12
System.out.println(e);

float  f = 1.2f; // f alebo F
double g = 1.2d; // d alebo D
double h = 1.234e2;
double i = 1.234e2d;
float  j = 1.234e2f;
System.out.println(h);
```

```java
boolean podmienka1 = true;
boolean podmienka2 = false;

char pismeno = 'a';

String text1 = "abcd";

String text2 = null;
```

linka: Escape Sequences

```java
String text3 = "jeden\ndva\n"; // \n \t \b
System.out.println(text3);

// String.class
```

Objekty – vytváranie a používanie

```
String text = "abcd"; // Specialna podpora netreba pouzit klucove slovo new

char [] obsah = {'a', 'b', 'c', 'd'};
String text2 = new String(obsah);

Color zlta = new Color(255, 255, 0); // https://www.csfieldguide.org.nz/en/interactives/rgb-mixer/
System.out.println("zlta: " + zlta.toString());

Color cervena = Color.RED;
System.out.println("cervena: " + cervena.toString());

int [] pole  = {10, 20, 30}; // budeme pouzivat
int pole2 [] = {10, 20, 30};
int [] pole3 = new int[] {10, 20, 30}; // budeme pouzivat
int pole4 [] = new int[] {10, 20, 30};

System.out.println("pole: " + Arrays.toString(pole));
```

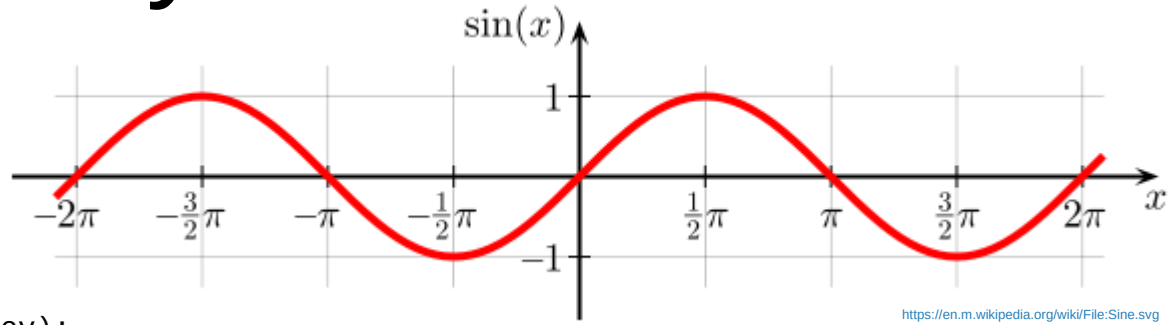
Objekty – vytváranie a používanie

```
String text1 = "abcd";
String text2 = "efgh";
String text3a = "ef";
String text3b = "gh";
String text3 = text3a + text3b; // "efgh"
String text4 = "efgh";
String text5 = text1;

System.out.println(text3);
System.out.println();

System.out.println(text2 == text3); // porovnavanie referencii
System.out.println(text2 == text4); // vysledok zalezi od optimalizacie
System.out.println(text1 == text4);
System.out.println();

System.out.println(text1.equals(text3)); // porovnavanie obsahu
System.out.println(text2.equals(text3));
System.out.println();

System.out.println("abcd".equals(text1));
System.out.println("abcd".equals(text2));
System.out.println();

System.out.println(text1.startsWith("ab"));
System.out.println(text1.startsWith("AB"));
System.out.println();

System.out.println(text1.replace('a', 'A')); // vrati novy objekt typu String
System.out.println(text1.toUpperCase()); // vrati novy objekt typu String
System.out.println(text1.substring(1,3)); // vrati novy objekt typu String
```

Objekty – vytváranie a používanie

```
StringBuilder text = new StringBuilder("jeden");
text.append(" dva");
System.out.println(text);

text.setCharAt(0, 'J');
text.setCharAt(6, 'D');
System.out.println(text);
```

Metódy tried



https://en.m.wikipedia.org/wiki/File:Sine.svg

```
// Math

double dvaPi = 2 * Math.PI;
System.out.println("dvaPi: " + dvaPi);
double piPol = Math.PI / 2;
System.out.println("piPol: " + piPol);
double piZoStupnov = Math.toRadians(180);
System.out.println("piZoStupnov: " + piZoStupnov);

System.out.println("sin(PI): " + Math.sin(Math.PI));
System.out.println("sin(PI/2): " + Math.sin(Math.PI / 2));
System.out.println();

// Arrays

int [] pole1 = {10, 20, 30, 40};
int [] pole2 = {10, 20, 30, 40};
System.out.println("toString: " + Arrays.toString(pole1));
System.out.println("equals: " + Arrays.equals(pole1, pole2));
System.out.println();

// System

String userDir = System.getProperty("user.dir");
System.out.println("userDir: " + userDir);
```

Štandardná dokumentácia

- https://docs.oracle.com/en/java/javase/21/
- https://docs.oracle.com/en/java/javase/21/docs/api/index.html
- vo vývojovom prostredí
  - myšou nad triedu alebo metódu
  - ctrl + click

Obálkové triedy (wrappery)

- pre primitívne dátové typy
- použitie:
  - tam kde treba referenčný typ
  - rozšírenie možností
- inštancie sú …..
- auto-boxing (automatické zabalenie)
  - napríklad `int` → `Integer`
- auto-unboxing (automatické rozbalenie)
  - napríklad `Integer` → `int`

Obálkové triedy (wrappery)

```
Integer i1 = 10; // auto-boxing
int i2 = Integer.parseInt("256");
int i3 = Integer.parseInt("100", 16); // rovnaka hodnota, len napisana v 16-ovej
sustave

System.out.println(i1);
System.out.println(i2);
System.out.println(i3);

int i4 = i1; // auto-unboxing

int i5 = Integer.max(10, 20);
System.out.println(i5);

System.out.println(Integer.toBinaryString(i5)); // 20 = 16 + 0 + 4 + 0 + 0
```

Operátory && ||

skrátené vykonávanie - optimalizácia, prípadne zabránenie nezmyselnej operácie

```
int delenec = 12;
int delitel = 4;
// int delitel = 0;
// int podiel= delitel / delitel;
boolean podielViacAkoDva = delitel != 0 && delenec / delitel > 2;
System.out.println(podielViacAkoDva);
```

```
int cena = 200;
int vPenazenke = 500;
int naUcte = 1000;
boolean mozemKupitIhned = vPenazenke >= cena || naUcte >= cena;
System.out.println(mozemKupitIhned);
```

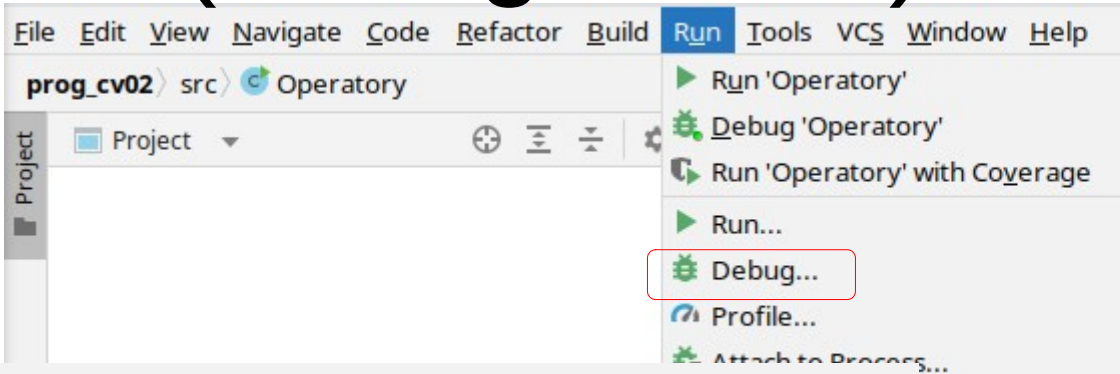
final (premenná)

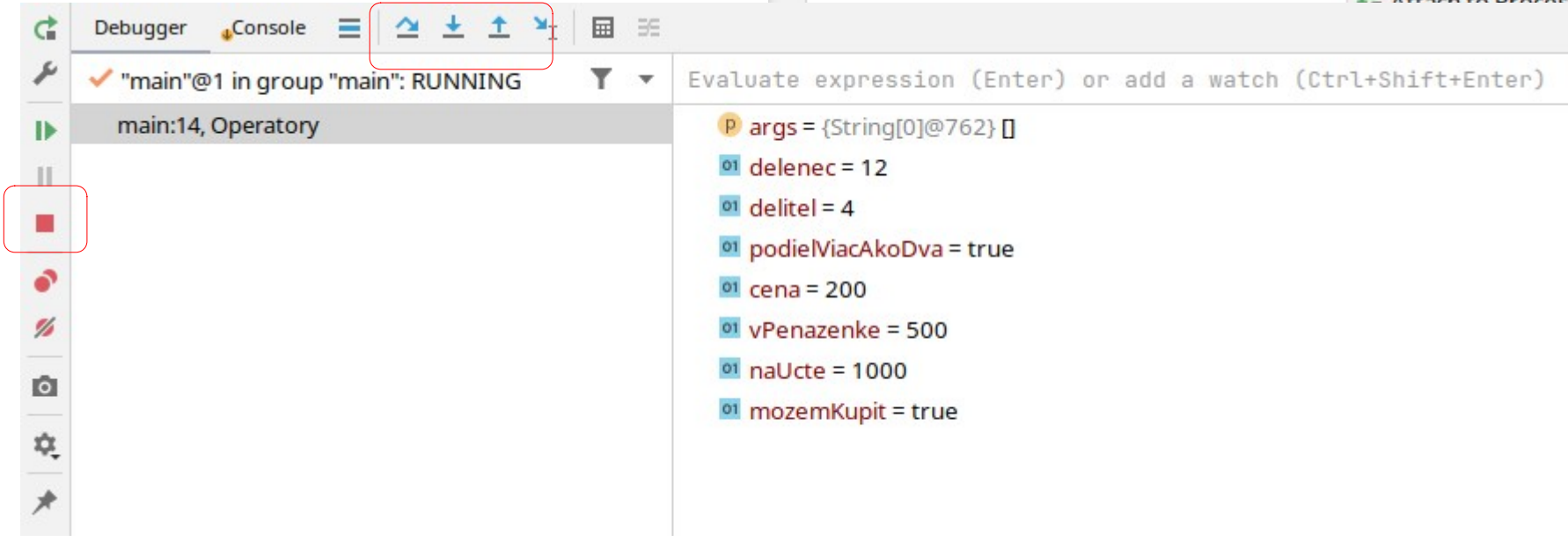
```
final int RAZ_NASTAVITELNA_A = 10;
// RAZ_NASTAVITELNA_A = 20; CHYBA

final int RAZ_NASTAVITELNA_B;
RAZ_NASTAVITELNA_B = 10;
// RAZ_NASTAVITELNA_B = 20; CHYBA
```

Kľúčové slovo `final` má aj iné použitie

Ladenie programu (debugovanie)

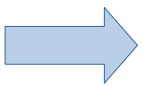
- V IntelliJ Idea
  - Run → Debug





Dátový typ uvádzať explicitne

```
var users = getUsers();
var address = getAddress();
var height = 1.2;
var count = 100L;
```



```
String[] users = getUsers();
String address = getAddress();
double height = 1.2;
long count = 100L;
```