Datové typy v Javě

```
<link rel="stylesheet"  href="http://cdnjs.cloudflare.com/ajax/libs/font-
awesome/3.1.0/css/font-awesome.min.css">
```

Úvod k datovým typům v Javě

- Existují dvě základní kategorie datových typů: **primitivní** a **objektové**

Primitivní

- v proměnné je uložena přímo hodnota
- např. `int`, `long`, `double`, `boolean`, ...

Objektové

- musí se nejdřív zkonstruovat (použitím `new`)
- do proměnné se uloží pouze odkaz
- např. `String`, `Person`, ...

Datové typy primitivní

integrální typy
: zahrnují typy *celočíselné* (byte , short , int a long) a typ char

čísel s pohyblivou řádovou čárkou
: float a double

logických hodnot
: boolean

Výchozí (default) hodnoty

- Každý typ má svou výchozí (default) hodnotu, na kterou je nastaven, není-li hned přiřazena jiná.
- Dle Java Language Specification: Each *class variable*, *instance variable*, or array component is initialized with a default value when it is created (§15.9, §15.10):

Type	Default value
`byte`	`(byte)0`
`short`	`(short)0`
`int`	`0`
`long`	`0L`

Type	Default value
`float`	`0.0f`
`double`	`0.0d`
`char`	`'\u0000'`
`boolean`	`false`
reference types	`null`

Na co se vztahují výchozí hodnoty

- Automatické nastavení proměnných na výchozí hodnoty se tedy vztahuje na proměnné objektů a tříd (atributy) a prvky polí.
- Nevztahuje se na lokální proměnné a parametry, ty musejí být před prvním použitím nastaveny, inicializovány.

Příklad výchozí hodnoty

```
int i; // automatically i = 0
```

Více informací najdete na: The Java Tutorials: Primitive Data Types.

Repl.it demo k primitivním typům a objektům

- https://repl.it/@tpitner/PB162-Java-Lecture-02-primitive-types

Zajímavosti a odlišnosti

- V Javě neexistuje možnost typy rozsahově či interpretačně modifikovat (žádné unsigned int apod.)
- Pro velká čísla lze v nových verzích Javy použít notaci s podtržítkem k oddělení řádů po tisících:

```
private int bigNumber = 123_456_789;
```

Datové typy objektové

- objektovými typy v Javě jsou všechna ostatní typy
- **třídy**
- **rozhraní** ("téměř totéž, co třídy")

- **pole** (ano, v Javě jsou samotná pole objekty)

Výchozí hodnota objektového typu je `null` — tzv. "ukazatel na nic".

Příklad použití objektového typu

```
Person p; // p is null automatically
p = new Person(); // now p references to an object
```

- Objektový typ je všechno, kde se používá operátor `new`.

Shrnutí

Základní rozdíl je v práci s proměnnými.

primitivní typy
: přímo obsahují danou *hodnotu*

objektové typy
: obsahují pouze *odkaz* na příslušný objekt

  - Důsledek: dvě objektové proměnné mohou nést odkaz na **tentýž objekt**

Přiřazení primitivní proměnné

- Hodnota proměnné se nakopíruje:

```
double a = 1.23456;
double b = a;
a += 2;
// a is 3.23456
// b is 1.23456
```

Přiřazení objektové proměnné

- Objektové proměnné ukazují na stejný objekt:

```
public class Counter {
   private double value;
   public Counter(double v) {
      value = v;
   }
   public void add(double v) {
      value += v;
   }
}
...
Counter c1 = new Counter(1.23456);
Counter c2 = c1;
c1.add(2);
// c1 has value 3.23456
// c2 has value 3.23456
```

Operátor ==

- Pro primitivní typy porovnává hodnoty:

```
1 == 1 // true
1 == 2 // false
```

- Pro objektové typy porovnává odkazy:

```
Counter c1 = new Counter(1.23456);
Counter c2 = c1;
c1 == c2 // true
c1 == new Counter(1.23456) // false
```

- Na porovnání *hodnot* objektových typů se používá `equals`, probereme později.

Použití u metod — primitivní typy

- Java funguje na principu "pass-by-value", tj. změna v metodě se neprojeví:

```
public static void main(String[] args) {
  int i = 1;
  passByValue(i);
  System.out.println(i); // 1
}

private static void passByValue(int i) {
  i = 4;
}
```

Použití u metod — objektové typy

- Odkazy na objekty fungují obdobně — změna objektu na jiný se neprojeví.
- Modifikace téhož objektu však ano!

```
public static void main(String[] args) {
  Dog d = new Dog("Max");
  passByValue2(d);
  System.out.println(d); // Charlie
}

private static void passByValue2(Dog d) {
  d.setName("Charlie");
  d = new Dog("Alex");
}
```

Pole v zkratce

- Vytvoření, naplnění a získání hodnot vypadá následovně:

```
int[] array = new int[2];
array[0] = 1;
array[1] = 4;
System.out.println("First element is: " + array[0]);
```

- Deklarace: *typ*[] *jméno* = `new` *typ* [*velikost*];
- *typ* může být i objektový: `Person[] p = new Person[3];`