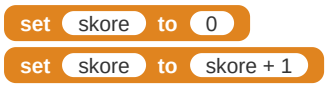
Z bloků do Javy

Cheat sheet: co znáš z code.org a jak to napsat v Javě

1. Proměnná

CODE.ORG



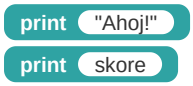
JAVA

```java
int skore = 0;          // vytvoření (celé číslo)
skore = skore + 1;      // změna hodnoty
String jmeno = "Anna";  // text
double cena = 9.5;      // desetinné číslo
boolean hotovo = false; // ano/ne
```

V Javě musíš při vytvoření říct **typ**. Typ píšeš jen jednou, podruhé už jen `skore = ...`

2. Výpis do konzole

CODE.ORG



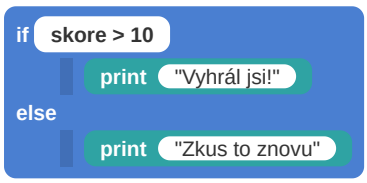
JAVA

```java
System.out.println("Ahoj!");
System.out.println(skore);
System.out.println("Skóre: " + skore); // + spojí text
```

`println` přidá nový řádek, `print` ne.

3. Podmínka (if)

CODE.ORG



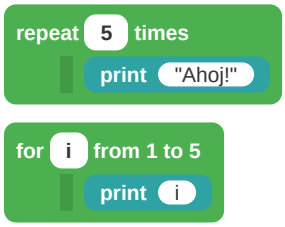
JAVA

```java
if (skore > 10) {
    System.out.println("Vyhrál jsi!");
} else {
    System.out.println("Zkus to znovu");
}
```

Porovnání: `==` rovná se, `!=` nerovná se, `<` `>` `<=` `>=`. Spojení: `&&` (a), `||` (nebo), `!` (ne).

4. Cyklus (for)

CODE.ORG



JAVA

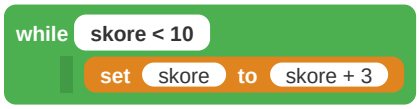
```java
for (int i = 0; i < 5; i++) {
    System.out.println("Ahoj!");
}

for (int i = 1; i <= 5; i++) {
    System.out.println(i);
}
```

`for (začátek; dokud platí; krok)`. `i++` znamená „zvětši i o 1“.

5. Cyklus (while)

CODE.ORG



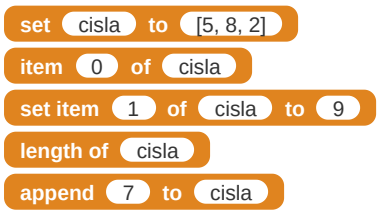
JAVA

```java
while (skore < 10) {
    skore = skore + 3;
}
```

Opakuje, **dokud podmínka platí**. Uvnitř musí být něco, co ji jednou změní na nepravdu, jinak je to nekonečný cyklus.

6. Pole a seznam

CODE.ORG



JAVA

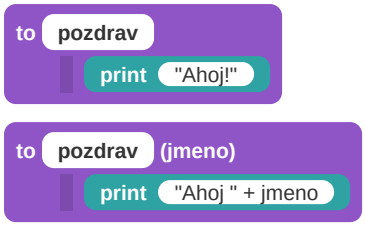
```java
// pole: pevná velikost
int[] cisla = {5, 8, 2};
System.out.println(cisla[0]);   // 5 (index od 0!)
cisla[1] = 9;
System.out.println(cisla.length); // 3
for (int c : cisla) {           // projdi všechny
    System.out.println(c);
}

// seznam: velikost se mění
import java.util.ArrayList;   // úplně nahoře
ArrayList<Integer> seznam = new ArrayList<>();
seznam.add(7);
seznam.get(0);
seznam.size();
seznam.remove(0);
```

Index prvního prvku je **0**, posledního `length - 1`. Mimo pole: `ArrayIndexOutOfBoundsException`. V seznamu místo `int` píšeš `Integer`, místo `double` `Double`.

7. Metoda (vytvoření)

CODE.ORG



JAVA

```java
static void pozdrav() {
    System.out.println("Ahoj!");
}

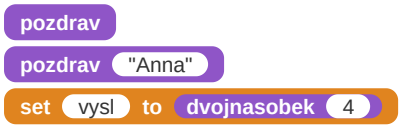
static void pozdrav(String jmeno) {
    System.out.println("Ahoj " + jmeno);
}

static int dvojnasobek(int x) {
    return x * 2;  // vrátí výsledek
}
```

`void` = nic nevrací. Jinak napíšeš typ výsledku (`int`, `String` ...). V závorkách jsou **parametry**.

8. Volání metody

CODE.ORG



JAVA

```java
pozdrav();
pozdrav("Anna");
int vysl = dvojnasobek(4);  // vysl je 8
```

Závorky `()` píšeš vždy, i když je prázdné. Metodu musíš nejdřív vytvořit, pak ji můžeš volat.

Primitivní datové typy v Javě (je jich 8)

Typ	Co ukládá	Velikost	Rozsah	Příklad
`byte`	malé celé číslo	8 bitů	−128 až 127	`byte b = 100;`
`short`	celé číslo	16 bitů	−32 768 až 32 767	`short s = 3000;`
`int`	celé číslo (nejběžnější)	32 bitů	−2 147 483 648 až 2 147 483 647 (cca ±2,1 mld.)	`int x = 42;`
`long`	velké celé číslo	64 bitů	−9 223 372 036 854 775 808 až 9 223 372 036 854 775 807 (cca $\pm 9{,}2 \cdot 10^{18}$)	`long l = 5000000000L;`
`float`	desetinné číslo	32 bitů	cca $\pm 3{,}4 \cdot 10^{38}$, přesnost ~7 číslic	`float f = 1.5f;`
`double`	desetinné číslo (nejběžnější)	64 bitů	cca $\pm 1{,}8 \cdot 10^{308}$, přesnost ~15 číslic	`double d = 9.99;`
`char`	jeden znak	16 bitů	0 až 65 535 (znaky Unicode)	`char c = 'A';`
`boolean`	ano / ne	1 bit (logicky)	`true` nebo `false`	`boolean ok = true;`

`String` **není** primitivní typ (je to třída), proto má velké S a píše se do uvozovek `"..."`, kdežto `char` do apostrofů `'A'`. Typ `long` potřebuje na konci čísla `L`, `float` potřebuje `f`. Při přetečení se číslo „otočí“: `Integer.MAX_VALUE + 1` je nejmenší `int`.

Celý program pohromadě

```java
public class Main {

    static void pozdrav(String jmeno) {          // metoda
        System.out.println("Ahoj " + jmeno);
    }

    public static void main(String[] args) {    // odsud program začíná
        int skore = 0;                           // proměnná
        for (int i = 1; i <= 5; i++) {          // cyklus
            skore = skore + i;
        }
        if (skore > 10) {                          // podmínka
            pozdrav("Anna");                       // volání metody
        }
        System.out.println("Skóre: " + skore);  // výpis (15)
    }
}
```

Nejčastější chyby

- Chybí `;` na konci příkazu.
- Nesedí počet `{` a `}`. Každá otevřená musí mít svou zavírací.
- Java rozlišuje velká a malá písmena: `String` není `string`, `skore` není `Skore`.
- Text patří do uvozovek `"..."`, čísla ne.
- `=` přiřazuje hodnotu, `==` porovnává.