Java 12 - 17

Records

```
record Point(int x, int y) {
}

var point = new Point(1, 2);
point.x(); // returns 1
point.y(); // returns 2
```

- Immutable schránka na data. Časem se podle mě bude používat pro DTO. Record implicitně obsahuje: private final atributy, gettery, all-args public konstruktor, equals, hashCode & toString
- Record může obsahovat custom instanční metody, statické metody a statické atributy.
- https://www.baeldung.com/java-record-keyword

Text Blocks

```
String html = """
    <html>
        <body>
            <p>Hello, world</p>
        </body>
    </html>
    """;
```

- Po prvním `"""` musí být newline!!!
- https://openjdk.java.net/jeps/355
- Bohužel neumí interpolaci jako je například v JavaScriptu, řešení viz. další snímek
- Super ve spojení s stripIndent() na konci.

Text Blocks & interpolace

V současné době jsou dva způsoby jak implementovat interpolaci v text blocks:

```
String html = """
    <html>
        <body>
            <p>Hello, ${name}</p>
        </body>
    </html>
""".replace("${name}", "Jirka");
```

```
String html = """
    <html>
        <body>
            <p>Hello, %s</p>
        </body>
    </html>
""".formatted("Jirka");
```

Instance of pattern matching

```
Object obj = "stuff 1";
if (obj instanceof String s && s.length() > 5) {
    System.out.println("obj is a String with more than 5 characters: "
                       + s.toUpperCase());
}
```

- Když se zjistí, že objekt je nějakého typu (pomocí instanceof), tak se rovnou uloží do nové proměnné bez nutnosti přetypování.

Switch expressions

```
enum Day {
    SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY
}

Day day = Day.MONDAY;

int numLetters = switch (day) {
    case MONDAY, FRIDAY, SUNDAY -> 6;
    case TUESDAY -> 7;
    default -> {
        String s = day.toString();
        int result = s.length();
        yield result;
    }
};
System.out.println(numLetters);
```

Switch nově může vrátit výsledek

Ve switch expression se nepoužívá break; … a chová se to, jako kdyby tam byl … konečně!

yield vrací hodnotu

https://nipafx.dev/java-13-switch-expressions/
https://www.baeldung.com/java-switch

Helpful NullPointerExceptions

```
A a = new A();
D d = a.b.c.d;
```

```
Exception in thread "main" java.lang.NullPointerException:
Cannot read field "c" because "a.b" is null
    at com.example.demo.Main.main(Main.java:11)
```

```
class A {
    B b;
}

class B {
    C c;
}

class C {
    D d;
}

class D {

}
```

Stream.toList()

```
List<String> results =
        Stream.of("one", "two", "three")
                .filter(s -> s.length() == 3)
                .toList();
System.out.println(results);
```

- Konečně! Místo .collect(Collectors.toList()) je možné použít .toList()
  - Pozor: toList() je unmodifiable! Takže to není 1:1 stejné!!!

Další vylepšení Streamů

- Stream::mapMulti
  - https://nipafx.dev/java-16-stream-mapmulti/
- Collectors.teeing
  - https://marco.dev/teeing

Vylepšení String

- indent(x), stripIndent()
  - Přidá / odstraní indentaci textu
- transform(Function)
  - Metoda, do které vstupuje Function a slouží pro transformaci aktuálního Stringu

Sealed classes

```
public abstract sealed class Shape
        permits Circle, Rectangle {...}

public final class Circle extends Shape {...} // OK
public final class Rectangle extends Shape {...} // OK
public final class Triangle extends Shape {...} // Compile error
```

- Sealed classes mohou omezit kdo je bude moci extendovat

Elastic Metaspace

- Metaspace paměť v Javě se od Java 16 rychleji uvolňuje operačnímu systému a díky tomu se sníží použití metaspace paměti:
  - https://openjdk.java.net/jeps/387

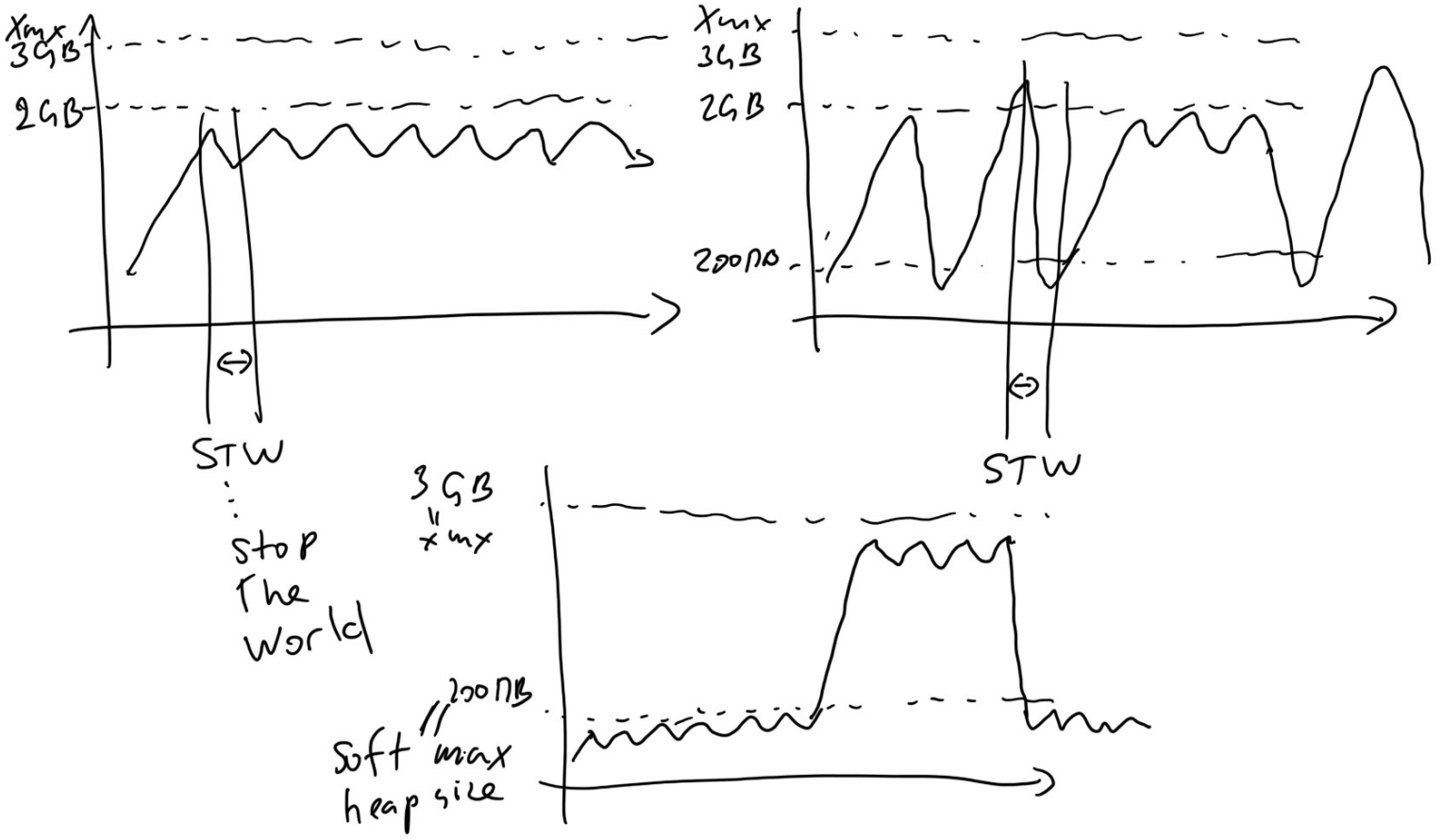
Shenandoah GC

- Další Garbage Collector, tentokrát od RedHatu, který se snaží o redukci GC pauz. Něco mezi G1 a ZGC:
  - https://wiki.openjdk.java.net/display/shenandoah/Main

Vylepšení ZGC

- Od Java 15 je ZGC production-ready, maximum heap size je 16TB a umí -XX:SoftMaxHeapSize (a přemýšlí se, že se tato funkcionalita v budoucnosti přidá i do G1)
  - https://malloc.se/blog/zgc-softmaxheapsize

Xmx vs. SoftMaxHeapSize



Všechny změny v API

- https://javaalmanac.io/