Java 19 update

Spis treści

Java 19 update

Java 19 została wydana we wrześniu 2022 i jest wersją **non-LTS**. Poniżej omówimy niektóre funkcjonalności udostępnione w tym wydaniu. Przy aktualizacji wersji Javy często poprawianych jest o wiele więcej funkcjonalności i dodawanych o wiele więcej klas lub metod niż te, które wymieniamy tutaj. W obrębie tych materiałów poruszamy tylko te kwestie, które są adekwatne do naszego poziomu zaawansowania jako Java developerów.

W tej wersji zostało dodanych kilka zmian w trybie preview, poniżej omówimy zmiany istotne z naszego punktu widzenia.

Swoją drogą, Java 19 jest dziesiątym wydaniem, które podąża za trybem wydań co 6 miesięcy.

Chcemy przyzwyczajać Cię, do tego, że niektóre zajavkowe materiały będą Ci udostępnione tylko w formie pisemnej i tak samo będzie w tym przypadku. Zmian dodanych w Java 19 (oczywiście z perspektywy naszej ścieżki), nie ma aż tak wiele, dlatego ten materiał będzie omówiony w postaci artykułu / notatki.

Virtual threads (preview)

Halo, przecież jeszcze nie było mówione o wielowątkowości, to jakie wątki i jeszcze do tego wirtualne. Tak wiem, że nie było, ale warto wspomnieć, że prace w tym obszarze cały czas trwają i obszar concurrency jest rozwijany na bieżąco. Virtual threads są preview feature w Java 19. Nie będziemy natomiast poruszać tego tematu w tym warsztacie. Temat wirtualnych wątków zostanie poruszony w dedykowanym warsztacie, gdy pojawią się one w Java jako standard feature. Jeżeli natomiast chcesz już teraz poczytać, o co w tym chodzi, możesz przeczytać np. ten artykuł.

Pattern Matching for switch (Third preview)

Mechanizm **Pattern Matching for switch** doczekał się kolejnych zmian i usprawnień, natomiast nadal został utrzymany jako **preview feature**. Mechanizm ten zostanie omówiony w pełni, gdy zostanie udostępniony jako **standard feature**.

Record Patterns (Preview)

Mechanizm *Record Patterns* również został wprowadzony w formie preview. Jest to swojego rodzaju rozszerzenie mechanizmu *Pattern Matching instanceof*. Mechanizm ten pozwoli na dopasowanie wartości do rekordu (`record`) i powiązać zmienne z odpowiednimi składowymi tego rekordu. Możemy również takiemu rekordowi nadać opcjonalny identyfikator, dlatego pojawia się tutaj słowo wzorzec. Następnie możemy się za pomocą identyfikatora do tego rekordu odwoływać.

Chyba w kodzie będzie łatwiej to zobrazować. Wyobraź sobie, że masz taki rekord:

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

Do tego, gdzieś w kodzie została zdefiniowana metoda `print()`:

```
class SomeClass {
    private void print(Object object) {
        if (object instanceof Point point) {
            System.out.printf("Object is a point, x = %s, y = %s", point.x(), point.y());
        }
        // else ...
    }
}
```

Powyższa notacja została poruszona już wcześniej i nosi nazwę *Pattern Matching instanceof*. To gdzie ten *Record Pattern*?

Mechanizm *Record Pattern* pozwala na taki zapis:

```
class SomeClass {
    private void print(Object object) {
        if (object instanceof Point(int x, int y)) {
            System.out.printf("object is a point, x = %s, y = %s", x, y);
        }
        // else ...
    }
}
```

Czyli zamiast przypisywać zmienną `object` do typu `Point`, możemy stworzyć definicję tego typu `Point` w locie, a następnie odwołać się do zmiennych `x` oraz `y` bezpośrednio, bez wywoływania metod dostępowych.

Z racji, że ten mechanizm jest w fazie *preview*, omówimy go dokładnie, gdy wejdzie już on do Javy w trybie *standard*.

Locale

W Java 19 została wprowadzona drobna modyfikacja w klasie `Locale`. Jeżeli napiszesz kod w ten sposób:

```
Locale localePL = new Locale("pl", "PL");
```

To okaże się, że: *'Locale(java.lang.String, java.lang.String)' is deprecated.*

Od Java 19, `Locale` tworzymy w ten sposób:

```
Locale localePL = Locale.of("pl", "PL");
```

Podsumowanie

Dlaczego tak szybko skończyliśmy? Java 19 wprowadziła relatywnie mało zmian. Dużo zmian wprowadzonych w Javie 19 dotyczyła mechanizmów, których albo nie umiemy, albo są zbyt skomplikowane na tym poziomie nauki, albo twórcy Zajavki uznali, że z naszego punktu widzenia zmiany te nie są aż tak istotne i lepiej poświęcić ten sam czas na skupienie się na dalszych zagadnieniach. Dlatego właśnie zostało omówione tak mało zagadnień. ☺

Przypomnę, że przy aktualizacji wersji Javy często poprawianych jest o wiele więcej funkcjonalności i dodawanych o wiele więcej klas lub metod niż te, które wymieniamy tutaj. Z kolejnymi wersjami wprowadzane są również rozmaite poprawki lub usprawnienia w samym działaniu JVM albo przykładowo Garbage Collectora (w tym przypadku mogą to być, chociażby różne algorytmy, o których działanie oparty jest GC). Zmianom mogą ulegać również kwestie dotyczące zarządzania pamięcią. Oprócz tego kolejne wersje Javy mogą również wprowadzać dodatkowe narzędzia, które programista może wykorzystywać w swojej pracy. Do tego poprawkom mogą podlegać istniejące implementacje metod. W obrębie tych materiałów poruszamy tylko te kwestie, które są adekwatne do naszego poziomu zaawansowania jako Java developerów.

Jeżeli natomiast interesuje Cię, jakie jeszcze zmiany są wprowadzane z każdą wersją — wystarczy, że wpiszesz w Google np. "Java 19 features" i znajdziesz dużo artykułów opisujących wprowadzone zmiany. Możesz również zerknąć na tę stronę JDK 19. Zaznaczam jednak, że wiele funkcjonalności będzie niezrozumiałych. ☺