Daniel Górski
Korporacyjna Java
Wykład 4: Java

Java

- Język obiektowy ogólnego zastosowania
- Język wysokiego poziomu
- Kompilowany do kodu pośredniego
  - Możliwość wywołania tego samego kodu na wielu platformach
- Język silnie typowany

Java: historia głównych wersji

- JDK 1.0 – 1996
- JDK 1.2 - 1998
- Java 6 – 2006
- Java 8(LTS) – 2014
- Java 11(LTS) – 2018
- Java 17(LTS) – 2021
- Java 21(LTS) – 2023

Java: jaki to język?

- Został stworzony jako język obiektowy ogólnego zastosowania
- Początkowo był rozwijany jako język, który pomoże rozwiązać największe bolączki C++
- Począwszy od wersji 8 jest obserwowane duże wzbogacenie języka elementami funkcyjnymi
- Począwszy od wersji 8 jest możliwe wielodziedziczenie kodu (domyślne implementacje w interfejsach)

Java: jak to działa

- Programista nie musi zwalniać pamięci: Garbage Collector się tym zajmuje
  - Jest opcja aby zajmować się samodzielnie
- Kompilacja polega na wytworzeniu bytecode'u z kodu aplikacji
- Bytecode początkowo jest interpretowany na JVM
- JIT (Just In Time) kompiluje bytecode do kodu natywnego

Inne języki na JVM

- Kotlin
- Scala
- Groovy
- Clojure

Najpopularniejsze IDE

- IntelliJ IDEA
- Eclipse
- VSCode
- Other
- Browser-IDE

Biblioteki

- Ilość i różnorodność bibliotek jest jedną z największych zalet Javy
- Z jednej strony ogromne "kombajny" typu Spring Framework, Hibernate
- Z drugiej jest to mnóstwo specjalistycznych biblioteczek do bardzo szczegółowych zadań
- Jest również sporo "wspomagaczy" kodowania jak Lombok

Java: pojęcia języka

- Klasa abstrakcyjna
- Interfejs
- Interfejs funkcyjny
- Metody domyślne
- Typ prymitywny
- Referencje Strong / Soft / Weak / Phantom

Java: na co uważać

- == oznacza tę samą wartość dla typów prymitywnych i tę samą instancję dla typów obiektowych
  - == może zadziałać np dla String'ów z przypisanymi wartościami i małych Integerów (domyślnie od -128 do 127)
- Uważnie z == dla typów zmiennoprzecinkowych
  - Klasa BigDecimal

Java: kontrakty

- hashCode() i equals()
- compareTo() / klasa Comparator
- Podklasy powinny realizować kontrakt nadklasy (Liskov substitution: SOLID)

Zajęcia: organizacja

- 7:30 – 8:15: Prezentacje
- 8:15 – 9:15: Wykład
- 9:15 – 9:45: Sesja pytań i odpowiedzi
- 9:45 – 10:00: Przerwa
- 10:00 – 11:30: Pracownia
  - Prezentacje
  - Pytania i odpowiedzi
- Ostatni wykład i pracownia najprawdopodobnbiej na żywo na uczelni

Pracownia: projekt

- Początkowo interfejs tekstowy z poziomu konsoli, docelowo będzie rozszerzany o możliwości przetwarzania plików i różne formaty wyjściowe
  - Program ma być konsolą: ma przyjmować w pętli polecenia i je wykonywać
- Poziom trudności zadań geometrycznych do rozwiązywania: maksimum trudne liceum

Pracownia: co wygląda dobrze

- Iteracja po dostępnych poleceniach w poleceniu help
- toLowerCase / toUpperCase wejścia – nie było tego w wymaganiach, ale pojawi się

Pracownia: co wygląda na ciężkie w utrzymaniu

- Teksty w różnych miejscach programu: przy obecnej wielkości nie jest to żaden problem, ale warto założyć, że z czasem będzie dużo poleceń i komunikatów

Pracownia

- Program ma umożliwić rozwiązywanie podstawowych zadań geometrycznych dla kwadratu i trójkąta równobocznego
- Ma wyświetlić pełne informacje o danej figurze, na podstawie zadanego wejścia
  - Polecenie z jednym parametrem

Pracownia: Kwadrat

- Charakterystyka figury
  - Długość boku
  - Długość przekątnej
  - Pole powierzchni
- Możliwe wejście: (1 cecha)
  - Długość boku
  - Długość przekątnej
  - Pole powierzchni

Trójkąt równoboczny

- Charakterystyka figury
  - Długość boku
  - Pole powierzchni
  - Wysokość
- Możliwe wejście: (1 cecha)
  - Długość boku
  - Pole powierzchni
  - Wysokość

Pracownia: ważne

- Obsługa błędów wejścia jest istotną częścią zadania