‘Zapiski uczestnika Bootcampu Zajavka’ by Bartek Borowczyk aka Samuraj Programowania
***Dopiski na czerwono** od Karola Rogowskiego*

Dzień 1 Bootcampu

Miłe wprowadzenie :)
Ten bootcamp to był dobry wybór ;)

Dzień 2 Bootcampu

Java jest językiem

-- **przenośnym** - a więc działającym niezależnie od sprzętu (procesora) i systemu, w oparciu o maszynę wirtualną - czym jest maszyna wirtualna (Java Virtual Machine) w dalszej części notatek :). Java w wyniku kompilacji kodu źródłowego napisanego w Javie tworzy kod pośredni (inaczej zwany **kodem bajtowym**), który może być wykonany na dowolnym procesorze/systemie, na którym jest zainstalowana **wirtualna maszyna Javy**. Nie trzeba tworzyć osobnego kodu dla różnych systemów/procesorów. Pamiętajmy na tym etapie, że przenośność oznacza niezależność od systemu czy procesora.

-- **szybkim** - kod pośredni (bajtowy) generowany przez kompilator Javy jest zoptymalizowany, podobnie zresztą jak maszyna wirtualna. Dlatego kod może wykonywać się szybko, więc mimo że mamy tutaj potrzebę interpretacji przez maszynę wirtualną kodu bajtowego, to działa to szybko jak błyskawica.

-- **bezpiecznym** - za sprawą m.in. ograniczania programu do środowiska uruchomieniowego (inaczej zwanego wykonawczym). Program nie może zrobić nic więcej niż pozwala mu właśnie maszyna wirtualna. Tak więc nie może np. uzyskać nieograniczonego dostępu do systemu. Liczę, że w przyszłości dowiemy się o bezpieczeństwie w Javie więcej :). Inaczej taką sytuację można nazwać sandboxem (dla dociekliwych Link).

A ponadto możemy o Javie powiedzieć, że jest językiem:
-- **ogólnego przeznaczenia** - to znaczy, że sprawdza się w licznych, zróżnicowanych zastosowaniach, nie tylko w serwisach webowych.

-- **językiem wysokiego poziomu** (w przeciwieństwie do języków niskopoziomowych) - nie chodzi tu o poziom rozwoju języka (choć rzeczywiście tu poziom Javy jest wysoki ;), tylko poziom abstrakcji względem kodu maszynowego (czyli 0 i 1, które rozumie procesor) Wszystkie współczesne języki (Java, PHP, Python, C++, Ruby, JavaScript, a nawet C) są językami wysokiego poziomu.
-- **językiem zorientowanym obiektowo** - program w Javie jest tworzony i reprezentowany za pomocą obiektów tworzony na podstawie klas. Obiekty składają się z danych i metod do obsługi tych danych. Tym zagadnieniem w bootcampie poświęcimy sporo czasu w 4. i 5. tygodniu nauki.

Co warto wiedzieć z historii Javy

- Opracowana przez Sun Microsystem w 1991 r. Nazwa Java funkcjonuje jednak od 1995 r.
- Bezpośredni przodkowie Javy (z których czerpie ona wiele rozwiązań) to języki C i C++. Składnia oraz model programowania obiektowego są oparte właśnie o C (składnia) i C++ (obiektowość). Natomiast językiem, który czerpał wiele z Javy był (i jest) C# (czyli Csharp).
- W 2010 r. firma Oracle kupiła Sun Microsystem i została właścicielem Javy.
- Java obecnie jest dostarczana przez Oracle w dwóch wersjach: wersji płatnej (SE - Standard Edition) i bezpłatnej (OpenJDK). Są to najbardziej popularne, ale nie jedyne dostępne dystrybucje Javy (specyfikacja Javy może być wdrażana także przez innych dostawców (vendorów - nawiązuję do nomenklatury w odcinku)).
- Java jest językiem mocno i systematycznie rozwijanym. Dwa razy w roku pojawia się nowa wersja. Obecnie najpopularniejszą wersją jest Java 11 (z oznaczeniem LTS - czyli długotrwałe wsparcie - Long Term Support). Najnowszą wersją Javy jest natomiast wydana w marcu 2021 wersja z numer 16.

Java jest językiem kompilowanym, ale także interpretowanym.

Kod napisany w Javie (a więc kod źródłowy) jest kompilowany przez kompilator (o tym w dalszej częsci) do formatu zrozumiałego dla maszyny wirtualnej Javy (zapamiętać skrót: **JVM**). Taki kod nazywamy **kodem pośrednim** lub **kodem bajtowym** (w odróżnieniu od kodu binarnego czy kodu maszynowego, który jest wersją już zrozumiałą dla procesora).

Ten wygenerowany w wyniku kompilacji kod jest zoptymalizowany do użycia przez JVM.

Kod bajtowy (pośredni) jest następnie wykonywany przez interpreter. Interpreter dostarcza właśnie JVM (tak, tak, ta maszyna wirtualna właśnie pełni rolę interpretera). Interpretowanie i wykonanie kodu ma miejsce w wyizolowanym środowisku (a więc jest 'całkiem' bezpiecznie - o czym już wspominałem wskazując cechy Javy). Kod bajtowy (przy okazji, po angielsku to bytecode) zapewnia także przenośność, wystarczy bowiem zainstalowana maszyna wirtualna na danej platformie i można uruchomić program.

Wróćmy jeszcze do szybkości - kod w Javie jest interpretowany w chwili wykonania (kod bajtowy nie jest zrozumiały bez JVM dla procesora), co oznacza, że Java jest potencjalnie językiem trochę wolniejszym niż języki, których kod jest od razu kompilowany do kodu wykonywalnego (maszynowego). Jednak w tym przypadku sposób optymalizacji zarówno kodu bajtowego jak i samej JVM zapewnia, że Java jest uważana za język szybki. (Kiedyś była uznawana za wolny, natomiast wszystko idzie do przodu)

Uwaga: Temat kompilacji i optymalizacji jest ciekawy ale zaawansowany. Pojawia się tutaj wiele pojęć takich jak kompilacja just-in-time czy ahead-of-time, ale na tym etapie (i często późniejszym) nie jest nam ta wiedza do niczego potrzebna ;) (Chyba, że będziesz potrzebował zejść naprawdę głęboko, ale w pracy na co dzień raczej się nad tymi pojęciami nie będziesz zastanawiać)

Kompilator - kompiluje kod Javy na kod pośredni (kod bajtowy). Kod bajtowy nie jest zrozumiały dla człowieka. Nie jest też zrozumiały dla procesora, dlatego określamy go kodem pośrednim. Dla kogo jest więc zrozumiały? Dokładnie! Dla wirtualnej maszyny Javy :) Kompilator jest dostarczony razem z JDK. No więc przejdźmy do JDK.

JDK - Java Development Kit - narzędzia developerskie Javy / środowisko programowania w Javie

Javę najczęściej utożsamia się z językiem programowania. Potocznie oczywiście Java to język programowania, ale, patrząc całościowo, Java to całe JDK. Dobrze to widać, gdy mówimy o wersji Javy. Numer (wersja) Javy odnosi się właśnie do JDK.

JDK należy traktować jako pakiet narzędzi dla programistów Javy, który udostępnia przede wszystkim dwa programy:

- ***kompilator*** - nazywa się on **javac** - zamienia kod napisany w Javie na kod bajtowy

- ***interpreter*** - o nazwie **java** - który interpretuje kod bajtowy, a więc w praktyce uruchamia aplikacje

Kod napisany w Javie umieszczamy w plikach z rozszerzeniem **.java** (pliki źródłowe)**.** Po kompilacji (pliki wynikowe) mają już rozszerzenie **.class.**

krok 1 - uruchomienie w konsoli kompilatora

```
C:\Users\48512>javac app.java
```

krok 2 - uruchomienie w konsoli interpretera (tutaj już bez rozszerzenia .class wpisujemy

```
C:\Users\48512>java app
```

Zanim przejdziemy dalej, zastanówmy się, co obejmuje rdzeń zasad (specyfikacji) Javy. Są to przede wszystkim:
-- maszyna wirtualna
-- zdefiniowany język programowania (czyli m.in. składnia)
I właśnie te rzeczy są wdrażane potem w konkretnym JDK. JDK z danym numerem jest zgodne ze specyfikacją dla danego numeru.

Występują dwie wersje Javy dostarczane przez Oracle:

- Java z literkami SE oznaczającymi Standard Edition np. Java SE 11 (11 oznacza, że oparta o specyfikację JDK 11)
- Java OpenJDK, która również ma swoje numeryczne oznaczenie, a od wersji SE różni się tym, że jest open source i można jej używać za darmo.

JRE - Java Runtime Environment - środowisko uruchomieniowe Javy
Odpowiedzialne za uruchomienie programu. Zawiera zarówno maszynę wirtualną Javy, jak i zestaw użytecznych klas (klas podstawowych).

JVM - Java Virtual Machine - maszyna wirtualna Javy
Jest elementem JRE. Interpretuje i wykonuje kod bajtowy.

Jak wygląda cały proces od kodu źródłowego do wykonania programu:

1. kod źródłowy w Javie (rozszerzenie plików .java)
2. proces kompilacji kodu źródłowego [kompilator]
3. wygenerowanie kodu bajtowego (rozszerzenie plików .class)
4. proces interpretacji kodu bajtowego [interpreter - maszyna wirtualna]
5. wyjście - działanie programu

Dzień 3 Bootcampu

Skąd pobrać Javę?

-- bez rejestracji dostępna wersja 16

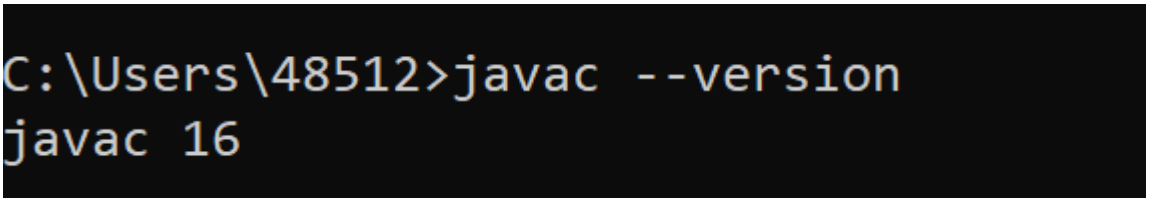
-- wersja 11 Javy wymaga rejestracji.

https://openjdk.java.net/

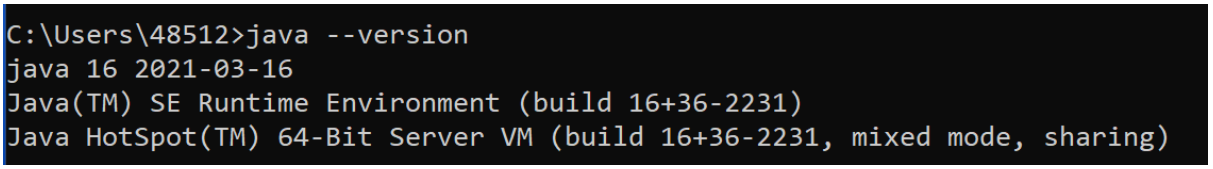
Jak sprawdzić, czy wszystko poszło dobrze?

Uruchomić konsolę/terminal (np. cmd albo powershell, jeśli chodzi o Windowsa)

Kompilator:

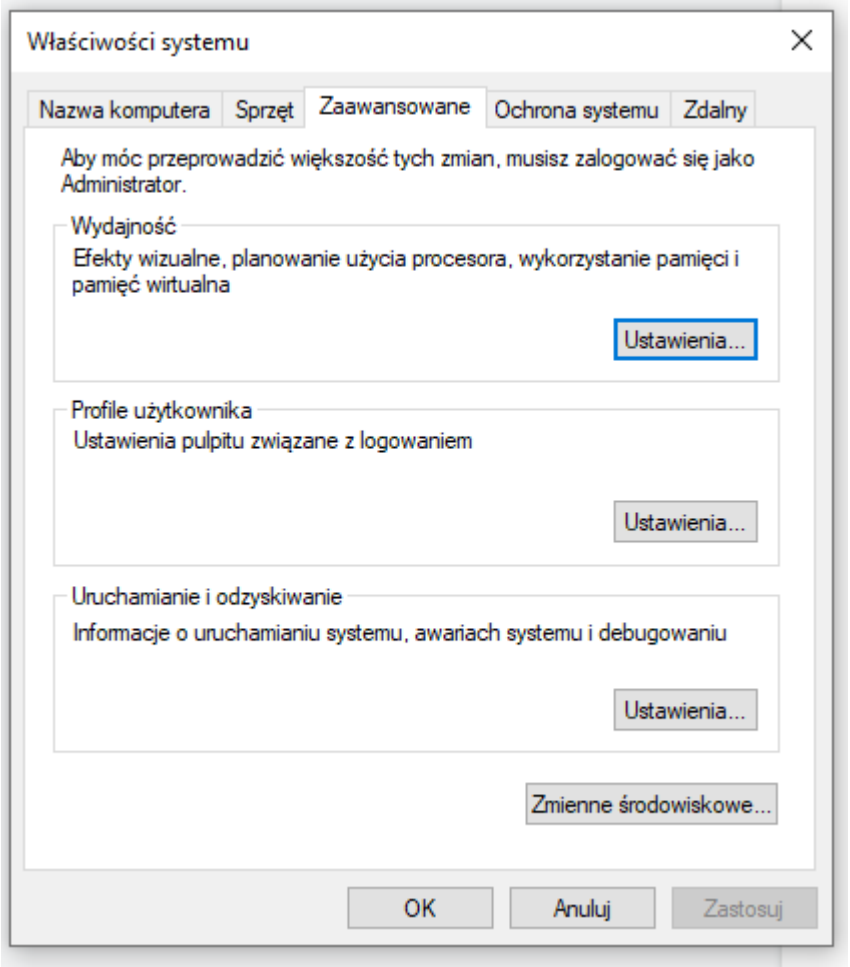


Interpreter:



Zmienne środowiskowe

Jeśli nie chodzą polecenia javac i java w konsoli, to należy dodać zmienne środowiskowe wg przykładu poniżej.



Umieszczamy ścieżkę do miejsca, w którym zainstalowaliśmy (rozpakowaliśmy) Javę. W przykładzie poniżej ta ścieżka już jest (a Java znajduje się akurat w folderze C:\jdk-11). Jeśli brakuje ścieżki u Ciebie, użyj przycisku ‘Nowy’ i wprowadź ścieżkę do folderu bin.