Język JAVA

podstawy programowania

[43] Na ogół łatwiej daje się człowiek przekonać racjom, do których sam doszedł, niż tym, które nastręczyły się komuś innemu.

„Myśli” Blaise Pascal

Wprowadzenie

Język JAVA jest niewątpliwie najbardziej rozwijającym się obecnie środowiskiem tworzenia aplikacji. Czerpie on to co najlepsze z takich języków jak C++ czy Smaltalk przy zdecydowanie prostszej i bardziej czytelnej składni (konstrukcji) programów. Zawiera elementy programowania zarówno strukturalnego jak i obiektowego, zdarzeniowego jak i współbieżnego. Poprzez standardowe jak i rozszerzone biblioteki wkracza w różnorodne rejony zastosowań takie jak np. karty inteligentne i elektronika, systemy zarządzania bazami danych, obsługa multimediów, Internet, grafika 3D, kryptografia, itd. Co więcej JAVA jest niespotykanie bezpiecznym środowiskiem i umożliwia w znaczny sposób kontrolę i sterowanie bezpieczeństwem. Zdecydowanie różni się od innych języków trzeciej generacji tym, że jest językiem interpretowanym a nie kompilowanym. Oznacza to, że powstały w wyniku kompilacji kod wynikowy nie jest programem jaki można niezależnie uruchomić lecz stanowi tzw. Beta-kod, który jest interpretowany przez Maszynę Wirtualną (JavaVM) pracującą w określonym środowisku. Ze względu na kod nie istotne jest na jakim sprzęcie będzie uruchamiana aplikacja. Ważna jest tylko Maszyna Wirtualna. Jest to niezwykle ciekawy pomysł umożliwiający odcięcie się od wszystkich poziomów sprzętowo-programowych będących poniżej Maszyny Wirtualnej. Koncepcja ta jest powszechna również w samym języku JAVA, dzięki czemu poprzez stworzenie abstrakcyjnych klas i metod podstawowe biblioteki Javy nie muszą być nieustannie rozbudowywane.
JAVA jest niewątpliwie językiem najbliższej przyszłości, warto więc poświęcić mu trochę czasu.

Krótka historia Javy

1990 - Bill Joy w raporcie „Further" sugeruje SUNowi stworzenie środowiska obiektowego na bazie C++,

1991 - W ramach projektu „Green" powstaje język OAK - „Object Application Kernel" (James Gosling), przeznaczony dla aplikacji w elektronice powszechnego użytku,

1995 - zmiana nazwy na JAVA ze względu na zastrzeżenie nazwy OAK,

1996 - Pojawia się Netscape zgodny z Javą 1.0, Sun propaguję darmowe środowisko JDK 1.0,

1999 - Java 2 Nowe oblicze Javy.

Główne źródła informacji o Javie w Internecie

http://java.sun.com, SUN, The source of Java Technology (źródło technologii Javy), a w szczególności:

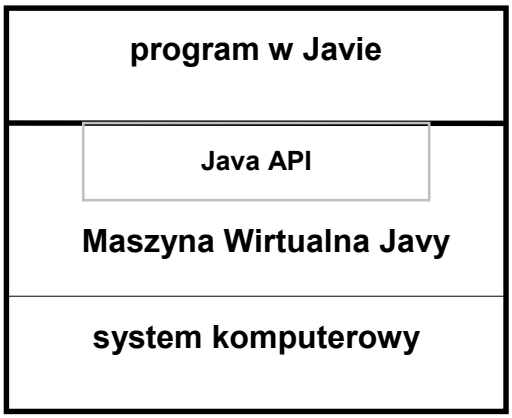
http://java.sun.com/docs/books/jls/, Gosling J., Joy B, Steele G. The Java Language Specification. Addison-Wesley, 1996, (specyfikacja języka !!!),
http://java.sun.com/products/jdk/1.2, SUN, Java 2 SDK software and documentation site (strona źródłowa oprogramowania i dokumentacji Javy),
http://developer.java.sun.com/developer/infodocs/, SUN, On-line books & Tutorials (książki i podręczniki do nauki Javy),
http://java.sun.com/docs/books/tutorial, SUN, The Java Tutorial (Podręcznik Javy).

Forum użytkowników Javy, poradniki, testy, doświadczenia:
http://www.javaworld.com,
http://www.javareport.com,
http://www.jars.com,
http://www.gamelon.com,
http://www.javalobby.com

i wiele innych, których źródła nie można nawet wymienić.

Java platformą tworzenia i wykonywania aplikacji

Platformą nazywa się przeważnie pewną kombinację sprzętu i oprogramowania umożliwiającą tworzenie i wykonywanie programów. Przyjęło się powszechnie mówić o tzw. platformie sprzętowo-programowej. Platformę stanowi więc komputer o danej konfiguracji oraz system operacyjny w środowisku którego uruchamiana jest dowolna aplikacja. Przykładowe platformy to Intel PC + Windows NT; Sun Ultra + Solaris; SGI O2 + Irix 6.4, itp. Konstrukcja platformy Javy jest podobna, niemniej nie odnosi się bezpośrednio do sprzętu i systemu operacyjnego, które stanowią dla niej pewną abstrakcję. Istotą platformy Javy jest zbiór dwóch elementów: Java API (Application Programming Interfaces) - interfejsy tworzenia aplikcaji oraz JavaVM (Virtual Machine) - maszyna wirtualna. Maszyna wirtualna Javy jest rozwinięciem dotychczas używanego pojęcia platformy, stanowiąc pewną nadbudowę. Maszyna wirtualna interpretuje kod wynikowy (Beta-kod) Javy do kodu wykonywalnego danego systemu operacyjnego i komputera, którego jest nadbudową. Oznacza to, że Maszyna Wirtualna jest interfejsem pomiędzy uniwersalnym kodem Javy, a różnymi konfiguracjami komputerów. Ta różnorodność systemów komputerowych wymaga różnorodności Maszyn Wirtualnych. Firma Sun dostarcza obecnie Maszynę Wirtualną wersji Java 2 dla systemów operacyjnych Windows95/98/NT oraz Solaris. Oczywiście interpretacja kodu właściwa dla danego systemu operacyjnego (konwersja w locie Beta-kodu do kodu wykonywalnego) wymaga odpowiednich bibliotek. Biblioteki klas, metod, pól, itp., zarówno te zależne sprzętowo jak i te niezależne sprzętowo stworzone już w Javie znajdują się w postaci skompilowanej w Java API. Podsumowując Java VM oraz Java API tworzą platformę Javy zwane często środowiskiem uruchomieniowym aplikacji - Java Runtime Engine (JRE).



Rysunek 1. Środowisko uruchomieniowe programu w Javie - platforma Javy.

Java - środowisko tworzenia aplikacji

Aby można było uruchomić aplikację na platformie Javy trzeba ją najpierw stworzyć, po czym skompilować do Beta-kodu. Posługując się regułami języka Java oraz zapleczem klas i metod powstaje kod źródłowy programu. W celu generacji Beta-kodu program ten podaje się następnie kompilacji (np. kompilatorem „java" firmy Sun w pełni stworzonego za pomocą języka Java). Dla potrzeb tworzenia aplikacji SUN oferuje pakiet Java Development Kit (JDK lub Java SDK - Software Development Kit), który składa się z JRE oraz narzędzi kompilacji i bibliotek.

Java - język programowania

Stworzenie programu w Javie polega na umiejętnym wykorzystaniu znajomości reguł języka oraz bibliotek. Konstrukcja programu i reguły języka przypominają znacznie język C. Programiści znający ten język z łatwością i przyjemnością rozpoczną pracę z Javą. Dla znawców języka C miła będzie informacja, że w Javie nie używa się w ogóle wskaźników, statycznego rezerwowania i zwalniania pamięci itp. Program w Javie nie zawiesi się więc z uwagi na „Null Pointer Assigment". Bardzo istotne w konstrukcji programu jest znajomość obsługiwanych typów. Ponieważ kod Javy jest niezależny od sprzętu, to również typy danych są niezależne od sprzętu (platformy). Java jest językiem obiektowym, dzięki czemu kod jest uniwersalny i bardzo czytelny. Nowością jaką niesie ze sobą Java jest również tworzenie tzw. apletów. Aplet jest programem wykonywanym w określonych ramach (nadbudowie Maszyny Wirtualnej). Aplet ma więc takie możliwości jakie nadaje mu program uruchomieniowy. Przykładowe programy uruchomieniowe to przeglądarki WWW np. Netscape, Internet Explorer. Kolejnym nowym elementem konstrukcyjnym Javy jest to, że można kod grupować w liczne wątki, które w wyniku interpretacji tworzą niezależnie wykonywane procesy współdzielące czas procesora.

Opis środowiska Java 2 SDK

Proces tworzenia aplikacji Javy z pomocą dostarczanego przez Suna środowiska Java 2 SDK można przedstawić następująco:

1. Napisanie z pomocą dowolnego edytora tekstu kodu źródłowego programu zawierającego klasę publiczną o nazwie takiej samej (dokładnie takiej samej z uwzględnieniem wielkości znaków) jak docelowa nazwa programu np. RycerzJedi.

2. Nagranie kodu źródłowego jako pliku o danej nazwie z rozszerzeniem .java, np. RycerzJedi.java

3. Kompilacja kodu źródłowego zawartego w pliku z rozszerzeniem .java do pliku docelowego o rozszerzeniu .class zawierającego Beta-kod np.
c:\ javac RycerzJedi.java
gdzie:

javac - nazwa komilatora programów Javy stworzonego przez Suna (kompilator napisany w Javie),
RycerzJedi.java - kod źródłowy programu do kompilacji (WAŻNE: podana nazwa pliku musi zawierać rozszerzenie .java).

W Wyniku kompilacji powstanie plik lub zestaw plików z tym samym trzonem nazwy o rozszerzeniu .class, np. RycerzJedi.class.

4. Uruchomienie w środowisku interpretatora Beta-kodu, np.
c:\ java RycerzJedi
gdzie:

java - nazwa interpretatora Javy stworzonego przez Suna, inaczej uruchomienie Maszyny Wirtualnej,
RycerzJedi - nazwa pliku z Beta-kodem programu w Javie kompilacji (WAŻNE: podana nazwa pliku nie może zawierać rozszerzenia .class).

W celu kompilacji i uruchomienia programu napisanego w języku Java użyto w powyższym przykładzie dwóch podstawowych narzędzi pakietu Java 2 SDK: javac oraz java. Kompilator „javac" (często nazywany „Jawak") jest nieodzowną częścią pakietu SDK, podczas gdy interpretator „java" stanowi specyficzną dl adanej platformy część pakietu środowiska uruchomieniowego Java Runtime Engine. Wynika stąd, że po instalacji pakietu Java SDK interpretator „java" będzie znajdował się zarówno w części JRE (niezależnej od tworzenia aplikacji) jak i w zbiorze narzędzi tworzenia aplikacji. Przykładowo katalog zawierający Java 2 SDK wygląda następująco:

<DIR> bin
<DIR> demo
<DIR> include
<DIR> include-old
<DIR> jre
<DIR> lib
935 COPYRIGHT

8˙762 LICENSE
6˙010 README
9˙431 README.html
16˙715˙279 src.jar
313˙746 Uninst.isu

W katalogu „bin” znajdują się liczne narzędzia obsługi aplikacji np:

javac - kompilator,
java - interpretator z konsolą ,
javaw - interpretator bez konsoli,
javadoc - generator dokumentacji API,
appletviewer - interpretator apletów,
jar - zarządzanie plikami archiwów (JAR),
jdb - debager,

Ze względu na to, że pisząc programy w Javie często korzysta się z narzędzi znajdujących się w katalogu „bin”, warto ustawić w środowisku ścieżkę dostępu do tego katalogu. Narzędzia dostępne w tym katalogu można wywoływać z licznymi opcjami. Praktycznie jednak najbardziej przydatne opcje to:

***javac:**
-g ->wyświetl pełną informację debagera,
- verbose ->wyświetl wszystkie komunikaty w czasie kompilacji,
np. javac -g -verbose RycerzJedi.java
***java:**
-cp -classpath -> gdzie -classpath katalog zawierający wykorzystywane klasy użytkownika (lepiej ustawić zmienną środowiska CLASSPATH),
-version ->wyświetl wersję platformy Javy.

Drugim ważnym katalogiem jest katalog „jre”. Jak łatwo się domyślić w katalogu tym znajduje się Java Runtime Environment JRE - platforma Javy. Zgodnie z tym co powiedziano na początku platforma Javy składa się z Maszyny Wirtualnej oraz bibliotek API. Dlatego katalog „jre” podzielony jest na dwa podkatalogi: „bin” - w którym znajduje się interpretator „java” (ten sam co wcześniej) oraz : „lib” gdzie znajdują się spakowane biblioteki API oraz pliki konfiguracyjne i środowiskowe platformy (np. określające poziom bezpieczeństwa, czcionki, fonty, itp.).

Uwaga praktyczna:

W czasie nauki języka Java lepiej unikać wszelkiego rodzaju programów typu szybkiego tworzenia aplikacji, gdyż traci się często kontrolę nad zrozumieniem treści tworzonego programu.

Integralną częścią środowiska Javy są biblioteki. Korzystanie z bibliotek jest znacznie prostsze jeśli rozumie się jak z nich korzystać. Nieodzownym jest więc korzystanie z dokumentacji bibliotek API. Opis bibliotek jest dostępny oddzielnie względem środowiska JDK i stworzony jest jako serwis WWW, który można przeglądać on-line lub off-line. Dokumentacja zawiera spis wszystkich pakietów, klas, ich pól i metod oraz wyjątków wraz z odpowiednimi opisami.