

Platformy technologiczne

Java Platform, Standard Edition (JSE) –
powtórzenie najważniejszych informacji

Technologie Java

- Java Platform, Standard Edition (JSE)
 - aktualna wersja: 6.0 (1.6)
 - ciągle szeroko używana jest wersja 5.0
- Java Platform, Enterprise Edition (JEE)
 - aktualna wersja: 6.0 (bardzo „świeża”)
 - ciągle szeroko używana jest wersja 5.0 i 1.4
 - bazuje na J2SE dostarczając usługi przydatne do tworzenia aplikacji serwerowych
- Java Platform, Micro Edition (JME)
 - „uproszczona” wersja Javy na komórki, PDA, BD itp.
 - odpowiednio sprofilowane zestawy API

Gdzie szukać dokumentacji?

- Na stronie <http://java.sun.com> jest bogaty zbiór dokumentacji
- Obowiązkowe pozycje:
 - JSE:
 - JDK 6.0 Documentation (guides, API, tool docs...)
 - The Java Tutorial
 - Code Conventions for the Java Programming Language (<http://java.sun.com/docs/codeconv/index.html>)
 - JEE:
 - The JEE 5/6 Tutorial
 - Java Platform, Enterprise Edition 5/6 Specification
 - i dokumenty powiązane

Gdzie szukać dokumentacji?

- Zalecane pozycje:
 - JSE:
 - Bruce Eckel „Thinking In Java 4th Edition”
 - Core Java Technologies Tech Tips (
<http://java.sun.com/developer/JDCTechTips/>,
<http://blogs.sun.com/CoreJavaTechTips/>)
 - JEE:
 - Java EE Technical Articles & Tips (
<http://java.sun.com/javaee/reference/techart/>)
 - JEE Blueprints (<http://java.sun.com/reference/blueprints/>)

Co potrzebne by zacząć działać?

- Wymagane:
 - JSE Development Kit 6.0
 - dla JEE - serwer aplikacji JEE, np.:
 - Glassfish 3 (<http://java.sun.com/javaee/downloads/index.jsp>)
(dostępny również razem z NetBeansem)
 - Apache Geronimo (<http://geronimo.apache.org/>)
 - JBoss
- Zalecane:
 - dobre środowisko IDE wspierające tworzenie aplikacji JEE
 - na zajęcia zalecany: NetBeans 6.8 (<http://www.netbeans.org>)
 - zalecam 1 GB dla systemów 32 bitowych i co najmniej 2 GB dla systemów 64 bitowych

Java – z czym to się je (przypomnienie)

- Źródła (klasy) Javy umieszczamy w oddzielnych plikach, których nazwy muszą być takie same jak nazwy klas
- Klasy grupujemy w pakietach (katalogi) (obowiązkowo!)
- Biblioteki pakowane są jako archiwa JAR
- Archiwum JAR może służyć jako sposób dystrybucji aplikacji jeśli w pliku manifestu podamy klasę główną
 - `java -jar aplikacja.jar`
- Dokumentację generujemy poleceniem `javadoc`
- Inicjalizacja pól klasy może być wykonana przy deklaracji pól lub:
 - w konstruktorze dla pól obiektu;
 - w sekcji `static {...}` dla pól statycznych

Java – konwencje kodowania

- Najważniejsze zalecenia:
 - używamy javadoc do komentowania kodu i generowania dokumentacji
 - jedna deklaracja lub wyrażenie w linii;
 - klasy i interfejsy nazywamy wielkimi literami, każde kolejne słowo zaczyna się od wielkiej litery
 - MojaKlasa
 - nazwy metod i zmiennych zaczynamy małą literą, kolejne słowa wielkimi literami
 - mojaMetoda(int argumentLiczbowy) { ... }
 - double takaSobieZmienna;
 - nazwy stałych piszemy wyłącznie wielkimi literami, kolejne słowa oddzielamy podkreśleniem
 - public static final int BARDZO_WAZNA_STALA = 5;

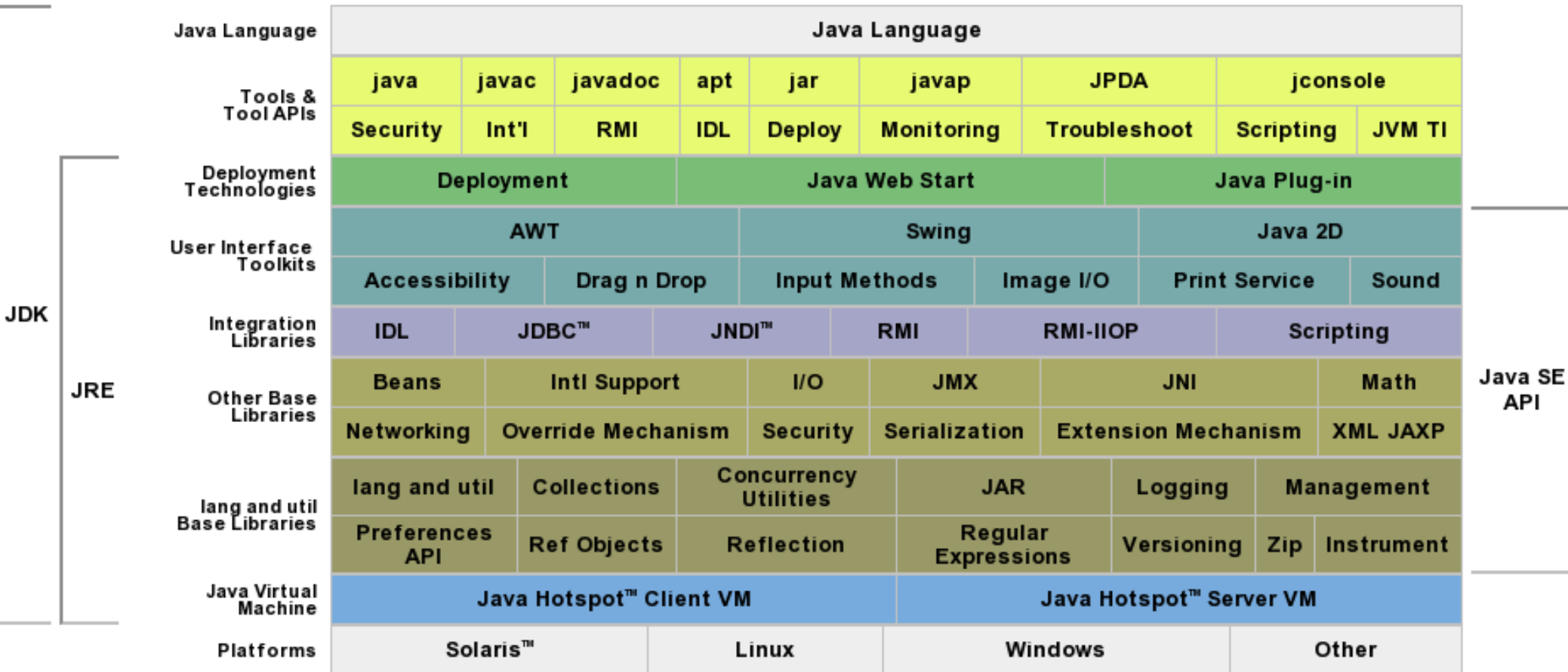
Java – wybrane konstrukcje języka

- Wygodniejsza obsługa kolekcji
 - generics
 - List<String>, Map<Integer, String>
 - autoboxing
 - int → Integer, Integer → int, ...
 - for do przeglądania kolekcji i tablic
 - for (String s : jakasListaStringow) { ... }
- Typ enum
- Zmienna liczba argumentów w metodach
- Adnotacje, np.:
 - @Override
 - @Resource

Java Beans

- Bardzo szeroko wykorzystywane w wielu typach aplikacji, szczególnie w JEE
- Są to komponenty, których można wielokrotnie używać
- Są to klasy Javy o specjalnej charakterystyce:
 - muszą być serializowalne
 - posiadają właściwości
 - jeśli klasa ma metody:
`public int getCecha()`
`public void setCecha(int cecha)`
to znaczy, że ma właściwość cecha typu int dostępną do odczytu i zapisu
- Można oznaczyć pola nie należące do stanu obiektu (transient). Te pola nie będą serializowane.

JSE – najważniejsze pakiety i klasy



JSE – najważniejsze pakiety, klasy i interfejsy

- java.lang
 - String, StringBuilder, StringBuffer
 - Integer, Double, ...
 - System
 - Object
 - ważne metody: equals(), hashCode(), toString()
 - Class
 - Thread, Runnable
 - Math, StrictMath
 - Comparable

JSE – najważniejsze pakiety, klasy i interfejsy

- java.util
 - List, Map, Set, SortedSet, SortedMap, NavigableSet, NavigableMap
 - jeśli zwracamy kolekcję używamy interfejsu nie implementacji!!!
 - (chyba, że niezbędna jest konkretna implementacja kolekcji)
 - Comparator
 - Iterator
 - ArrayList, HashMap, HashSet, TreeSet, TreeMap
 - zalecane jest używanie „nowych kolekcji”, nie są synchronizowane! (do synchronizacji wykorzystujemy klasę narzędziową Collections)
 - Arrays
 - Locale, TimeZone, Date, Calendar
 - Formatter, Scanner
 - Properties

JSE – najważniejsze pakiety, klasy i interfejsy

- java.io
 - ...InputStream, ...OutputStream
 - strumienie wejściowe/wyjściowe binarne;
 - ...Reader, ...Writer
 - strumienie wejściowe/wyjściowe tekstowe;
 - File
 - RandomAccessFile
 - Serializable, ObjectInputStream, ObjectOutputStream
 - serialVersionUID
 - Console
 - dostęp do konsoli tekstowej
- java.util.zip
 - strumienie do czytania/pisania plików ZIP i Gzip

JSE – najważniejsze pakiety, klasy i interfejsy

- java.text
 - DateFormat, NumberFormat
 - SimpleDateFormat, DecimalFormat
- java.util.regex
 - wyrażenia regularne
- java.math
 - BigDecimal, BigInteger
- java.net
 - URI, URL
 - Socket, ServerSocket
- java.util.logging
 - Logger, Level

JSE – najważniejsze pakiety, klasy i interfejsy

- `java.util.concurrent`
 - klasy wspierające programowanie wielowątkowe
- `javax.imageio`
 - klasy do czytania/pisania plików obrazków (JPEG, PNG)
- `java.sql`
 - JDBC - Java Database Connectivity
 - klasy pozwalające korzystać z baz danych
- `javax.xml`
 - klasy używane do przetwarzania XML-a
- `javax.swing`, `java.awt`
 - interfejsy użytkownika
 - Java2D