Informatica

Terzo anno

Prof. A. Longheu

INTRODUZIONE AL LINGUAGGIO JAVA

- Originariamente, il progetto che avrebbe portato al linguaggio Java era noto come progetto Green
- Nome del progetto Sun con l'obiettivo di fornire "intelligent consumer-electronic devices".
- Il risultato fu Oak
  - Un linguaggio basato su C++
  - Creato da James Gosling
  - Il nome del nuovo linguaggio fu cambiato in Java
  - E' fortemente ispirato al C++ ma riprogettato senza il requisito della piena compatibilità con il C (a cui però assomiglia)

Green in the red?

- Il progetto Green non andò lontano
  - Il mercato degli "intelligent consumer-electronic devices" crebbe lentamente
  - Sun non si impose in tale settore di mercato
  - Fu sul punto di essere cancellato
- L'esplosione del World Wide Web nel 1993 salvo il progetto Green
- Java fu ripensato come linguaggio per fornire contenuto dinamico alle pagine web
- Java fu formalmente annunciato nel 1995, oggi la versione è la 7.0 (o 1.7 update 51), in previsione la 8.0

Java

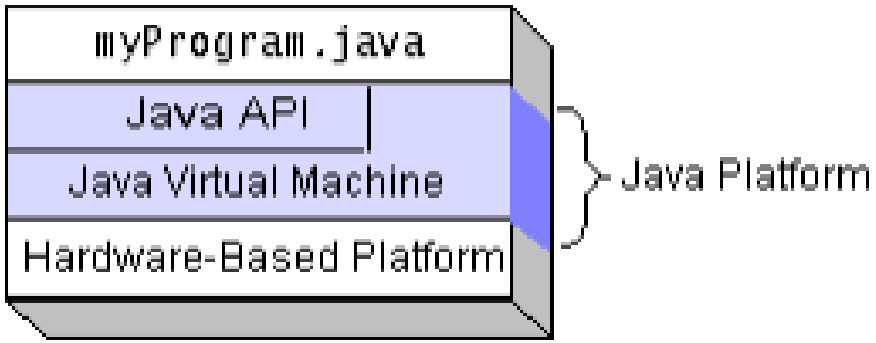
- Java nasce per applicazioni “embedded”
- Si diffonde attraverso il concetto di applet come piccola applicazione da eseguirsi dentro un browser Internet
  - grafica portabile ed eseguibile ovunque
- Può benissimo essere usato come linguaggio per costruire applicazioni anche non per Internet anche non grafiche (rimane un linguaggio general purpose)

La Piattaforma Java

- Piattaforma: ambiente hardware o software dove sono eseguiti i programmi (win-32, win-64, Linux, OSx)
- Una piattaforma in genere può essere descritta come una combinazione di sistema operativo e hardware
- la **Java platform** è solamente software e viene eseguita al di sopra di altre piattaforme basate sull'hardware

La Piattaforma Java

- La piattaforma consiste di due elementi:
  - Java Virtual Machine (**JVM**)
  - Java Application Programming Interface (**Java API**), ovvero una collezione di software pronti per l'uso, ad esempio per gestire Graphical User Interface (GUI), organizzati in librerie di classi e interfacce correlate (**packages**)

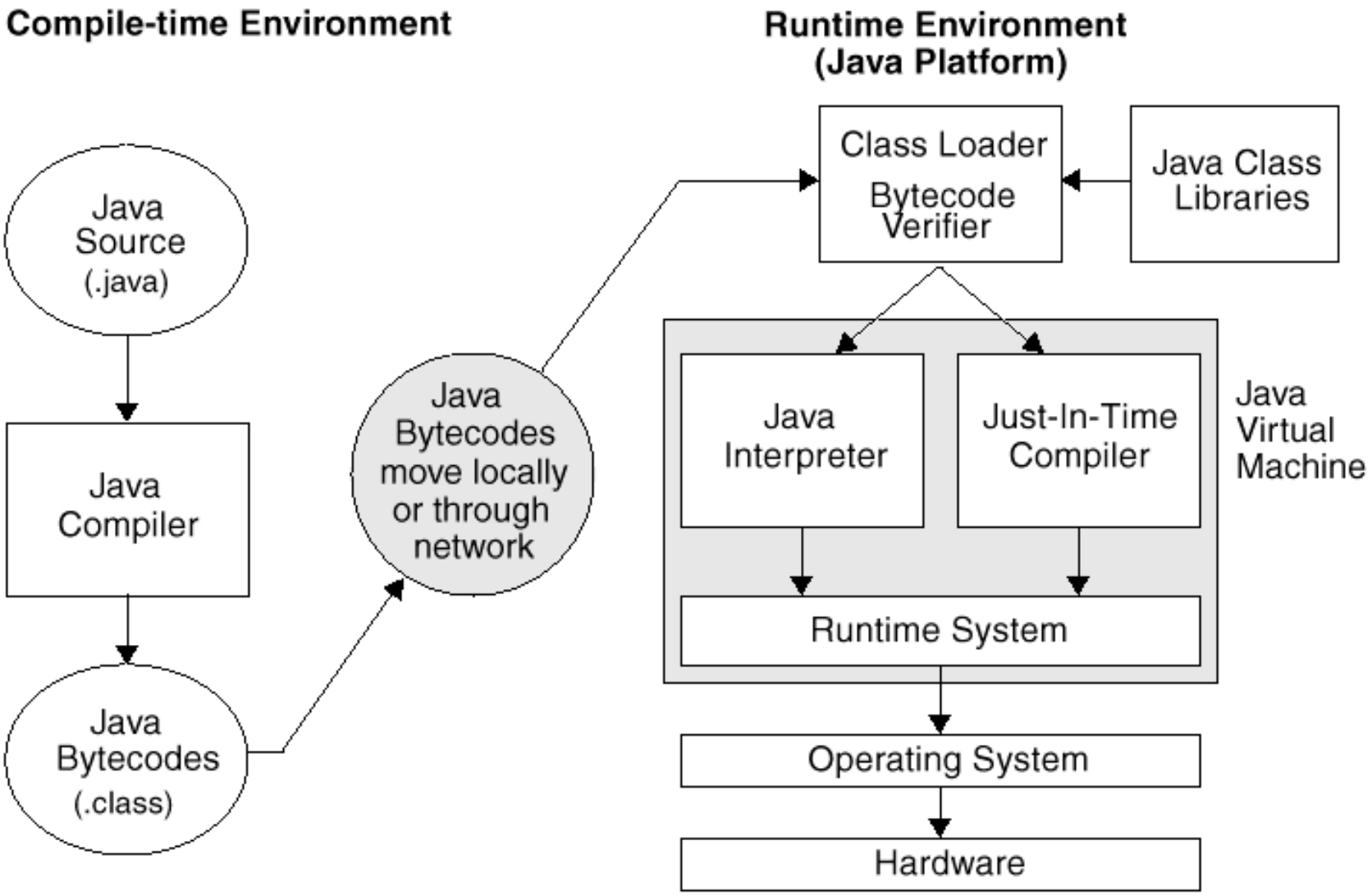


Java API e Java VM isolano il programma dall'hardware

La Piattaforma Java

- Il codice Java viene **compilato** in un linguaggio intermedio chiamato **bytecode**
- Il bytecode è **interpretato** a run-time dalla JVM e convertito nel linguaggio macchina del calcolatore su cui è richiesta l'esecuzione;
- la JVM è in quindi un computer virtuale sviluppato per rendere indipendente dalla macchina il codice Java: paradigma *"write once, run anywhere"*, talvolta mutato dai detrattori in *"write once, debug anywhere"*
- La **portabilità** del codice è assicurata dalla garanzia progettuale che i tipi di data abbiano comportamento standard al variare della piattaforma (ad esempio, i reali sono *IEEE-compliant*);
- in alternativa, il bytecode può essere compilato tramite un ***just in time*** (***JIT***) ***compiler***, qualora occorrano prestazioni maggiori.

La Piattaforma Java



La Piattaforma Java

Esistono **edizioni** differenti della piattaforma Java

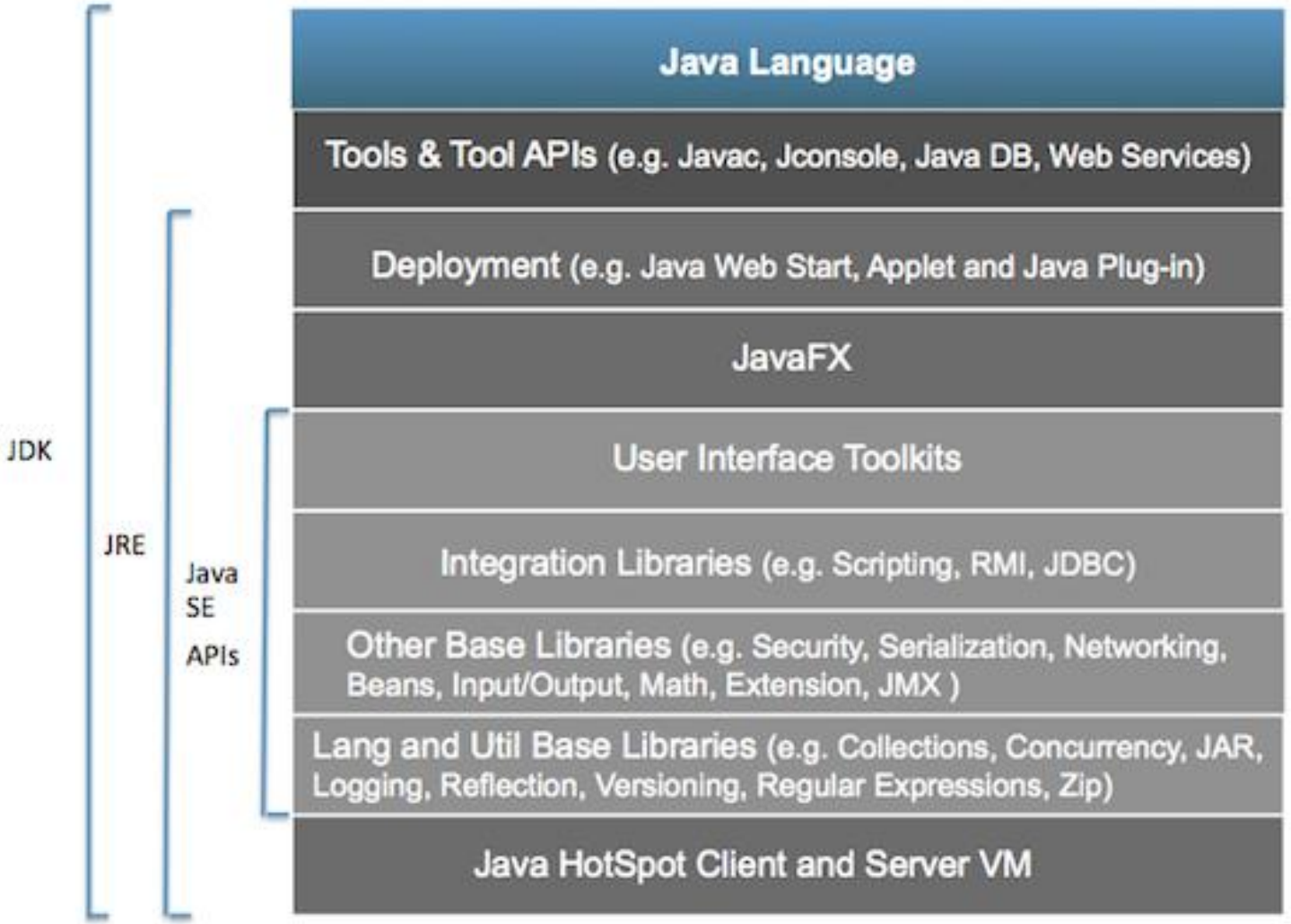
- **Standard Edition (J2SE)**
  - Librerie di base per lo sviluppo di applicazioni desktop (client applications) incluso AWT e Swing
  - Consente di eseguire applicazioni e applet
- **Enterprise Edition (J2EE)**
  - per lo sviluppo di applicazioni lato server
    - Per sviluppatori Web (EJB's, Servlets e JSPs)
    - richiede J2SE
- **Micro Edition (J2ME)**
  - per lo sviluppo di applicazioni mobili
    - librerie ridotte e più piccole
    - Implementa un subset delle funzionalita di Java

La Piattaforma Java

Ogni edizione si compone di:

- una serie di specifiche (documenti)
- una serie di strumenti (es: compilatore, debugger ecc.)
- una serie di librerie o API (Application Program Interfaces)
- schematizzando (eccessivamente) è possibile dire che i livelli più complessi includono i più semplici

Java2 – SE Conceptual diagram



Tipi di Programmi

Java permette di 4 tipi di programmi:

- Application
- Applet
- Servlet
- Beans

Vediamo solo i primi 2, e di questi nel seguito considereremo solo il primo (application)

Java Application

- Standalone program nel senso che richiede solo la JVM per essere eseguito
- Non richiede un programma host (come un browser) per l'esecuzione.
- Il metodo "main" è utilizzato come entry point e deve quindi essere presente
- Il metodo main deve avere la signature
  public static void main(String args[ ])

Java Applet

- Piccoli programmi tipicamente scaricati da un server su una macchina client
- La JVM è costruita all'interno del browser o di un apposito programma (appletviewer), che agiscono come programma host per l'*applet*.
- Un applet è tipicamente lanciato dentro un file HTML
- Un applet può essere inserito in un documento HTML con l'apposito tag <applet>
- Un *applet* generalmente opera sotto una gestione sicura che impone un *sandbox security*.
- Ciò previene che un applet possa eseguire operazioni potenzialmente pericolose come leggere e scrivere su un disco.
- Un applet è derivato dalle classi **`Applet`** o **`JApplet`**.

Esempio applet

esempio applet - Cerca co × | Corso Applets- Java Apple ×

www.echoecho.com/it/applets02.htm

Applicazioni | Importati da Firefox | Mail Ingegneria | OpenERP | La Repubblica.it - H...

- INTRODUZIONE
- MENU'
- BOTTONI
- SCORRIMENTO DI TESTO
- IMMAGINI IN SUCCESSIONE
- EFFETTI DI IMMAGINE
- BANDIERE ANIMATE
- OROLOGI
- ABBELLIMENTI

Uno degli applet più conosciuti è il **"Lake" applet**.

L'applet:

Il codice per inserire l'applet:

```
<html>
<head>
<title>Lake Applet</title>
</head>

<body>
L'Applet Lake<br><br>
<applet CODE="lake.class" width="370" height="200">
<param name="image" value="sunset.gif">
</applet>
</body>
</html>
```

Il Java Development Kit (JDK)

- Il JDK è l'insieme di strumenti di sviluppo che funge da "riferimento ufficiale" del linguaggio Java:
  - non è un ambiente grafico integrato: è solo un insieme di strumenti da usare dalla linea di comando
  - non è particolarmente veloce ed efficiente (non sostituisce strumenti commerciali)
  - però funziona, è gratuito ed esiste per tutte le piattaforme (Win32, Linux, Solaris, Mac..)
  - riferimento: http://www.java.com/it/ sito ufficiale per scaricare tutto il software e la documentazione

Documentazione

- La documentazione ufficiale è scaricabile dal sito http://docs.oracle.com/javase/7/docs/api/index.html
- Terminato il download, si decomprime il file in una cartella del computer e per consultare le Api basta eseguire file "index.htm" presente nella cartella di estrazione.
- Esiste anche il tutorial, ossia un percorso di apprendimento guidato dalle prime nozioni alle più avanzate, anche questo scaricabile dal sito ufficiale
- Su Internet esistono numerosi siti ricchi di esempi e guide. Qualche esempio:
  - http://examples.oreilly.com/jenut/
  - http://www.exampledepot.com/
  - http://www.java2s.com/Code/Java/JDK-6/CatalogJDK-6.htm
  - http://www.idevelopment.info/data/Programming/java/PROGRAMMING_Java_Programming.shtml

Ambiente di sviluppo

Esistono molti strumenti tesi a migliorare il JDK, e/o a renderne più semplice l'uso
- editor con "syntax highlightling", ad esempio Scite http://www.scintilla.org/SciTE.html
- ambienti integrati freeware o shareware che, pur sfruttando il JDK, ne consentono l'uso in modo interattivo e in ambiente grafico
  - JCreator LE, EditPlus, Forte, JaSupremo, etc...
  - Eclipse http://www.eclipse.org
- ambienti integrati commerciali, dotati di compilatori propri e debugger

Ambiente di sviluppo

- Eclipse è un IDE (ambiente di sviluppo integrato)
- progetto open source legato alla creazione e allo sviluppo di una piattaforma di sviluppo ideata da un consorzio di grandi società quali Ericsson, HP, IBM, Intel, MontaVista Software, QNX, SAP e Serena Software
- Usato per la produzione di software di vario genere
- Fornisce
  - Un completo IDE per il linguaggio Java (JDT, "Java Development Tools")
  - un ambiente di sviluppo per il linguaggio C++ (CDT, "C/C++ Development Tools")
  - plug-in che permettono di gestire XML, PHP
  - Plug-in per progettare graficamente una GUI per un applicazione JAVA (Eclipse VE, "Visual Editor"),

Java Programming

- Il più semplice Programma Java è costituito da una singola classe e da un singolo metodo, il main:

```
public class Programma {
        public static void main(String args[]){
         System.out.println("ciao");
        }
}
```

- Le parole chiave public e class devono essere presenti.
- Il nome della classe è arbitrario (purchè diverso da parole chiave), ma deve coincidere con il nome del file dove viene salvato il codice
- Dentro la coppia di parentesi graffe della classe viene inserito il metodo main, che deve essere dichiarato public, static, void
- Il metodo è un contenitore di un insieme di istruzioni (qui solo println)
- La classe è un contenitore di metodi; ogni metodo è quindi un "pezzo" del programma. In questo caso c'è solo il main
- Anche il main ha le sue {}, al cui interno si mettono le istruzioni del programma

Java Programming

- Convenzioni:
  - il nome di una classe ha sempre l'iniziale maiuscola (es. Esempio)
  - se il nome è composto di più parole concatenate, ognuna ha l'iniziale maiuscola (es. DispositivoCheConta)
  - non si usano trattini di sottolineatura
  - i nomi dei singoli campi (dati e funzioni) iniziano invece per minuscola
  - Tutte queste non sono regole; infrangerle non crea errori, ma viola la convenzione e quindi rende meno leggibile il codice, specie agli altri

Java Programming

- Altro esempio:

```
import java.io.*;
public class Esempio {
    public static void main (String args []) throws IOException {
            int PriNum;         int SecNum;         int Somma;

            BufferedReader tastiera= new BufferedReader(new
    InputStreamReader(System.in));

            System.out.println("inserisci il primo numero");
            PriNum=Integer.parseInt(tastiera.readLine());

            System.out.println("inserisci il secondo numero");
            SecNum=Integer.parseInt(tastiera.readLine());

            somma=PriNum+SecNum;
            System.out.println("la somma è "+somma);

    }
}
```

Java Programming

- Ogni volta che nel main c'è almeno una istruzione di I/O si deve mettere throws IOException nell'intestazione
- Tutti i programmi in genere prevedono una fase di input (prendere due numeri da tastiera), una di elaborazione (sommarli) ed una di output (stampare il risultato)
- Per fare i calcoli vengono utilizzate delle caselle di memoria chiamate variabili (qui PriNum, SecNum e Somma)
- Ogni variabile va "prenotata" prima del suo utilizzo; l'operazione viene chiamata dichiarazione e richiede di specificare il tipo e il nome della variabile, ad esempio int Somma significa che si desidera utilizzare una variabile chiamata Somma e destinata a contenere numeri interi
- BufferedReader tastiera= new BufferedReader(new InputStreamReader(System.in)); è l'istruzione che prepara la tastiera per potere permettere all'utente l'inserimento dei numeri
- La tastiera "grezza" è System.in e lavora con i byte; l'istruzione InputStreamReader() prende tale tastiera e ne converte i byte in caratteri; l'istruzione BufferedReader() prende a sua volta il risultato e permetterà di "bufferizzare", ossia di inserire tutti i caratteri desiderati fino alla pressione del tasto invio; senza la bufferizzazione il computer prenderebbe per buona già la pressione di un singolo tasto
- L'inserimento dei numeri avviene con il comando readLine(); poichè tale comando legge caratteri, occorre convertirli in numeri, ecco perchè si usa Integer.parseInt()
- I println che precedono ogni readLine() non sono obbligatori ma permettono di visualizzare un messaggio all'utente, che altrimenti si troverebbe un cursore lampeggiante in attesa senza sapere cosa scrivere
- System.out.println("la somma è "+somma); stampa la scritta "la somma è" a cui attacca il valore contenuto nella variabile somma; il + concatena le due parti