

GIỚI THIỆU JAVA



VINAENTER

Đã Học Là Làm Được

JAVA RA ĐỜI

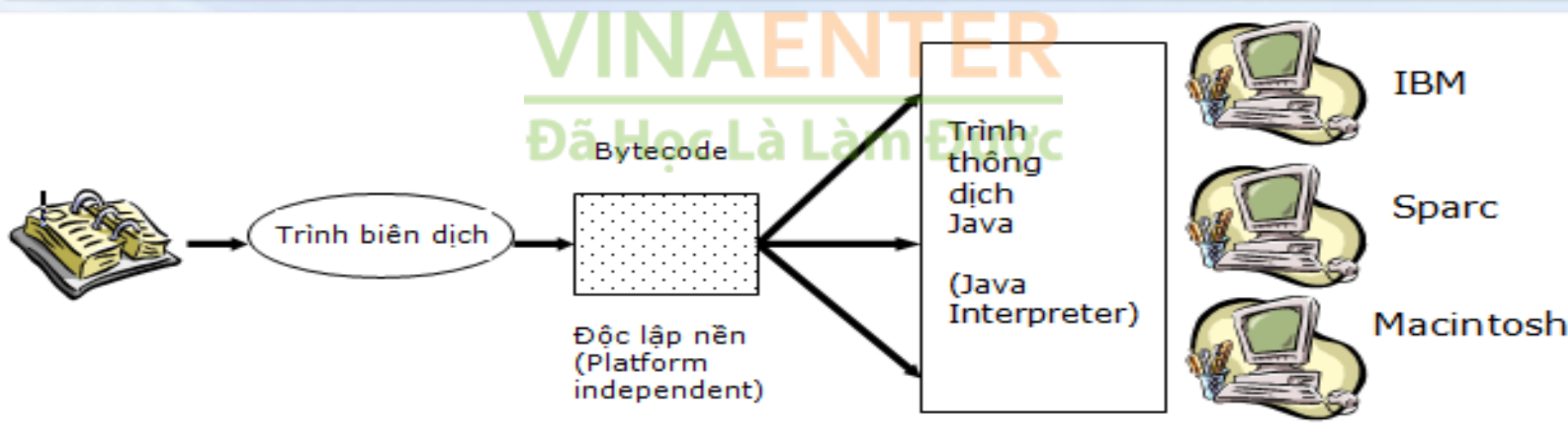
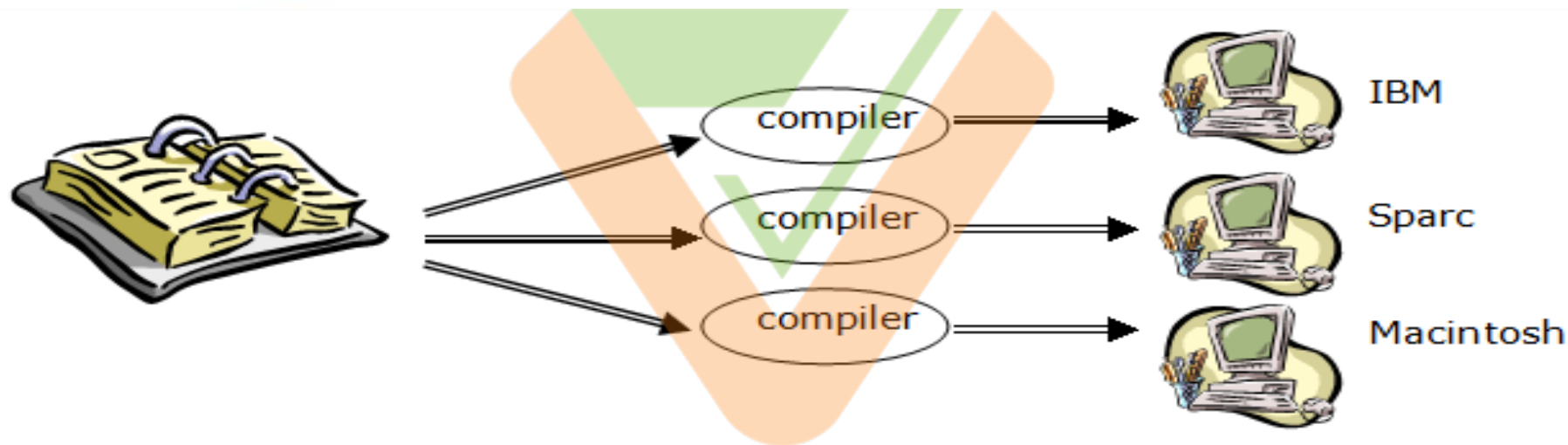
VinaENTER



- Năm 1991, Sun Microsystems muốn thiết kế một ngôn ngữ lập trình để điều khiển các thiết bị điện tử



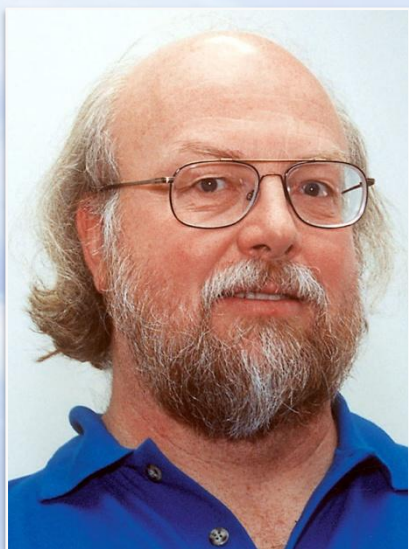
Trình dịch truyền thống & Java





OAK?

- Năm 1991: Java được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun Microsystems.
- Ban đầu ngôn ngữ này được gọi là GreenTalk
- 1995: Tên chính thức là Java



VINA ENTER

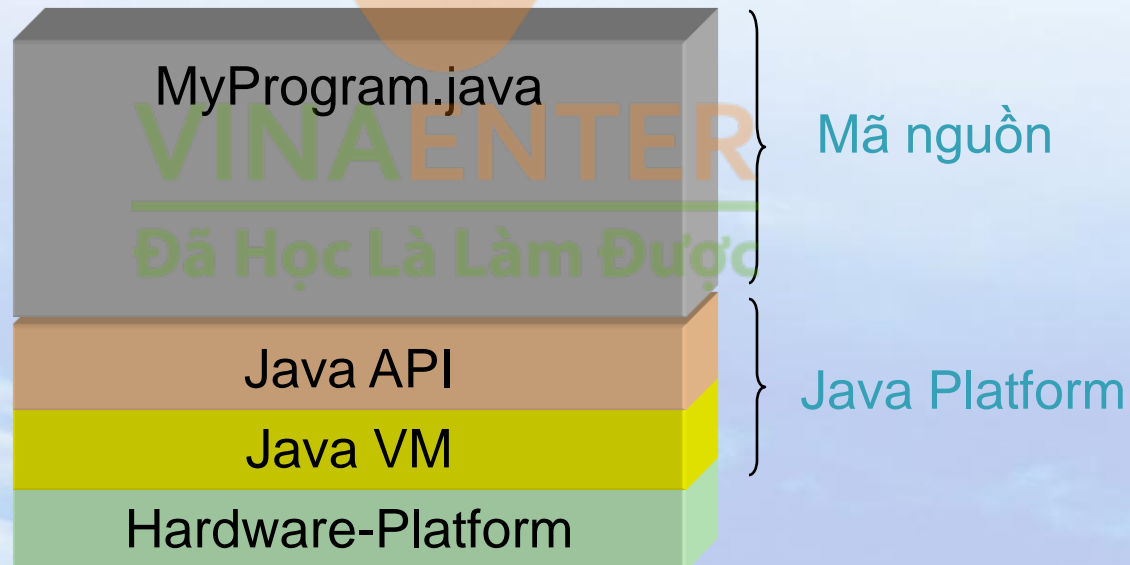
Đã Học Là Làm Được



Java Platform

VinaENTER

- Platform là một môi trường phần cứng hoặc phần mềm để chạy chương trình
 - Java Virtual Machine (JVM)
 - Java Application Programming Interface (Java API)





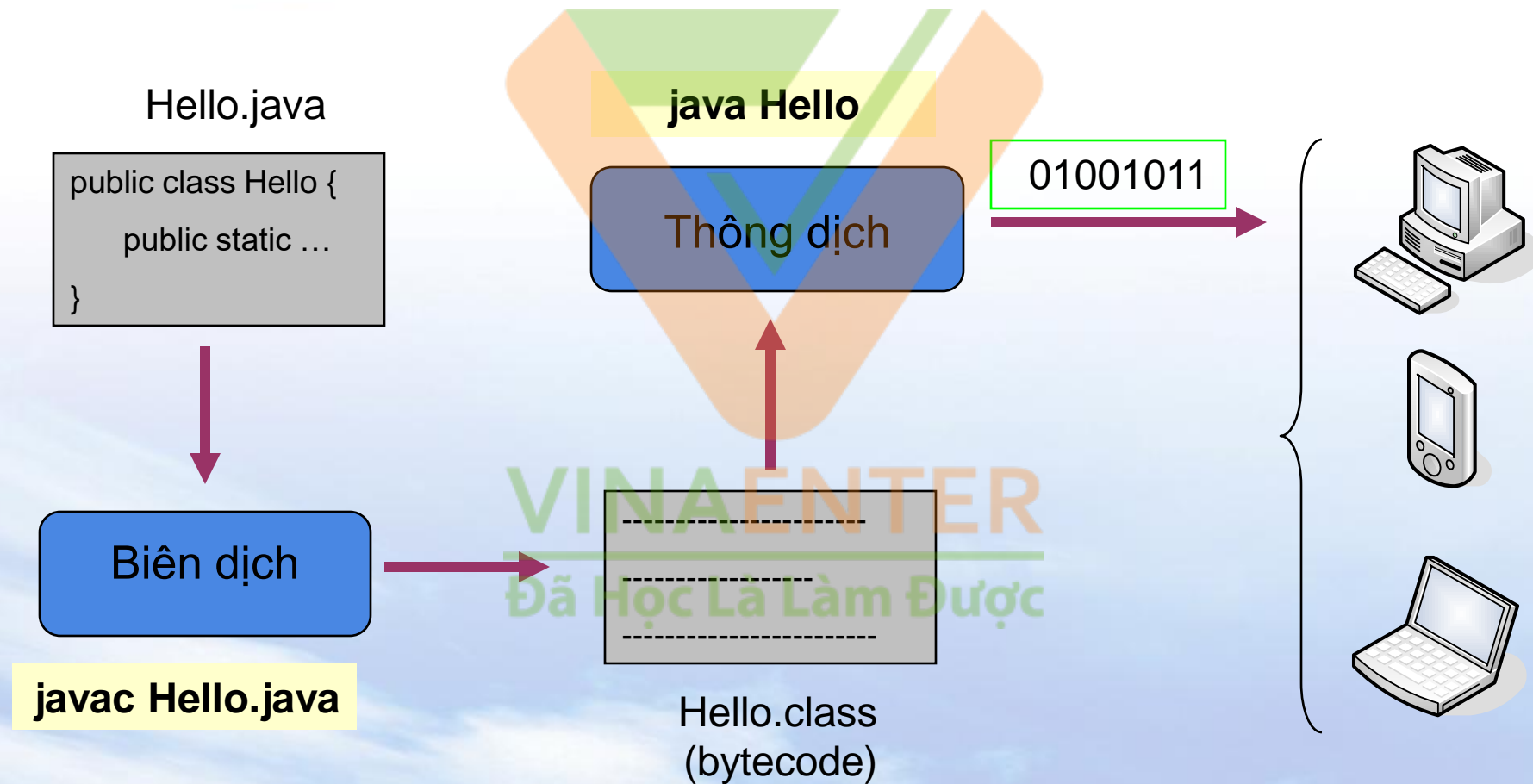
Kiến trúc của Java

- Java Development Kit – JDK
 - Bộ công cụ phát triển Java (jdk) gồm trình biên dịch, thông dịch, trợ giúp, soạn tài liệu... và các thư viện chuẩn
 - Ngoài ra còn một số thư viện khác như JSP, JavaMail, ...

Đã Học Là Làm Được



Phát triển ứng dụng Java





Quá trình dịch chương trình Java

- **Trình biên dịch (javac)** chuyển mã nguồn thành tập các lệnh không phụ thuộc vào phần cứng cụ thể
- **Trình thông dịch (java)** trên mỗi máy chuyển tập lệnh này thành chương trình thực thi
- **Máy ảo (JVM)** tạo ra một môi trường để thực thi các lệnh bằng cách:
 - Nạp các file `.class`
 - Quản lý bộ nhớ
 - Dọn “rác”

Cài đặt JDK, thiết lập môi trường JAVA

VinaENTER

- Cài đặt JDK
 - <http://www.oracle.com/technetwork/java/javase/downloads/index.html>
 - Thiết lập môi trường JAVA
- Các IDE: Notepad, Jbuilder, EditPlus, JCreator
- Cài đặt IDE Eclipse
 - <https://eclipse.org/downloads/packages/eclipse-ide-java-developers/keplersr2>
- Cài đặt IDE NetBeans
 - <https://netbeans.org/downloads/>



Java là gì?

- Là một ngôn ngữ OOP đầy đủ, không thể viết 1 ứng dụng hướng thủ tục trong Java.
- Có thể giải các họ bài toán như những ngôn ngữ lập trình khác.
- Cho phép tạo Application hoặc Applet.
- Applet là những chương trình nhỏ chạy trong tài liệu HTML với điều kiện trình duyệt có hỗ trợ Java (như IE, Netscape Navigator, HotJava,...)
- Sử dụng 2 cơ chế: Interpreter | Compiler
- Write code one, run it anywhere, anytime, forever



Đặc điểm của Java

- **Đơn giản(simple).** Tương tự như C++ nhưng bỏ bớt các đặc tính phức tạp của C++ như: quản lý bộ nhớ, pointer, overload toán tử, không dùng include, bỏ struct, union
- **Hướng đối tượng (OO).** Mọi thứ trong Java là đối tượng
- **Phân tán (Distributed).** Nhắm đến phân bố ứng dụng trên mạng, ứng dụng độc lập platform.
- **Mạnh (Robust).** Định kiểu mạnh, tường minh, kiểm tra lúc biên dịch và kiểm tra khi thông dịch trước khi thực thi → Giới hạn được lỗi; kiểm tra truy xuất phần tử của mảng, chuỗi lúc thực thi, kiểm tra ép kiểu run-time. Có trình *gom rác – garbage collection*- programmer không cần phải lo toan đến việc hủy đối tượng.

Đặc điểm của Java (tt)

- **Bảo mật (Secure):** Kiểm tra an toàn code trước khi thực thi, có nhiều mức kiểm tra bảo mật → Môi trường thực thi an toàn

Mức 1: Mức ngôn ngữ: Nhờ tính bao gói dữ liệu của OOP, không cho phép truy cập trực tiếp bộ nhớ mà phải thông qua method.

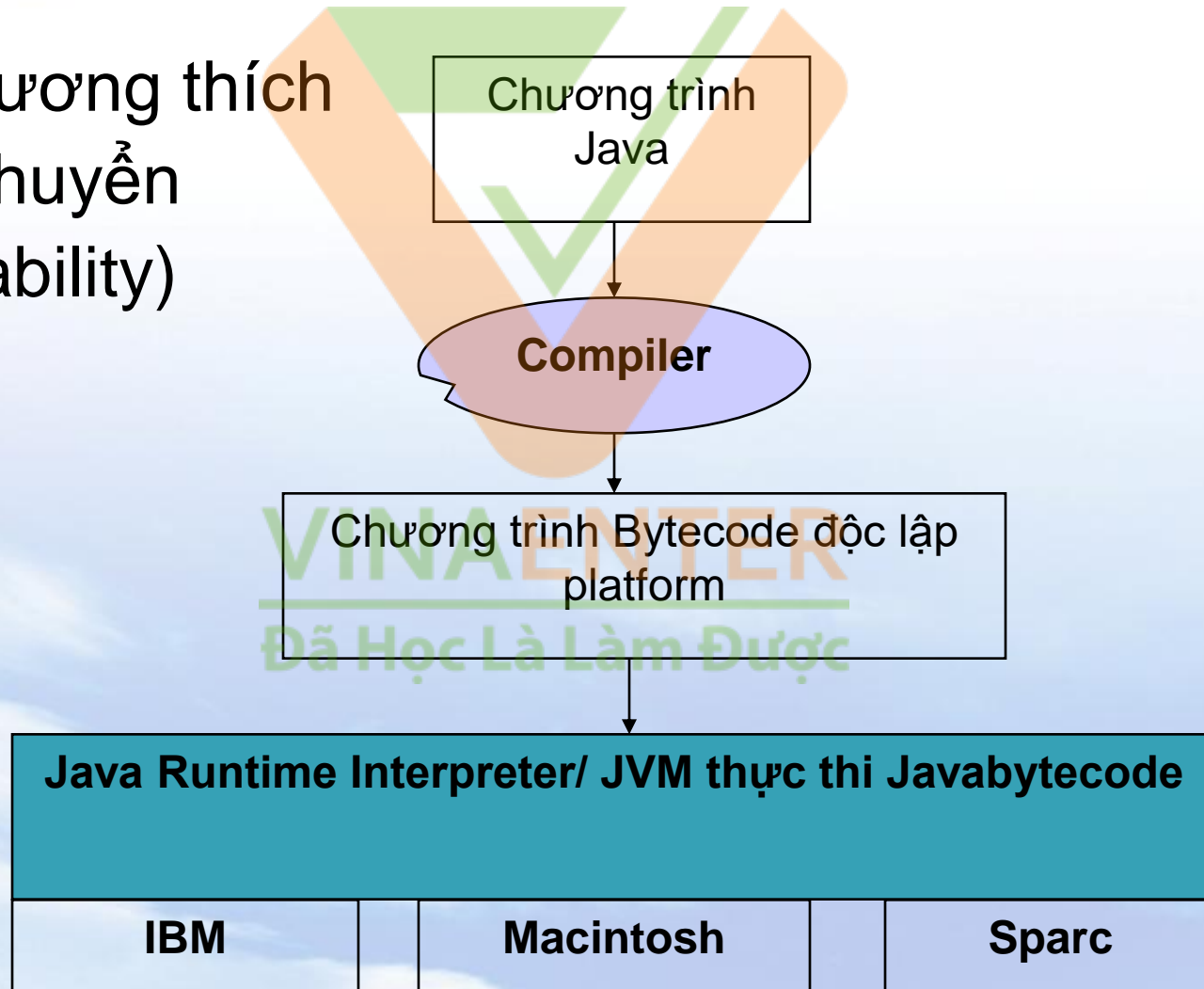
Mức 2: Mức Compiler, kiểm tra an toàn cho code trước khi biên dịch.

Mức 3: Mức Interpreter, trước khi bytecode được thực thi, được kiểm tra an toàn.

Mức 4: Mức Class, các class trước khi nạp được kiểm tra an toàn.

Đặc điểm của Java (tt)

- Tính tương thích
khả chuyển
(Portability)



Đặc điểm của Java (tt)

VinaENTER

- **Thực thi dạng thông dịch: (Interpretive execution)** Chỉ thị chỉ được dịch sang lệnh máy lúc thực thi
Chương trình độc lập platform → Write Once Run Anywhere (WORA)
 - (1) Các file tài nguyên → trình biên dịch **javac** → class file độc lập thiết bị
 - (2) Class file → trình thông dịch **java** → mã máy thực thi, không cần liên kết (link)
- ➔ **Lợi ích**
 - (1) Java class file có thể được dùng ở bất kỳ platform nào.
 - (2) Tính module hóa cao, dùng bộ nhớ tốt hơn với class file hơn là file thực thi vì class file cần một bước dịch nữa mới được CPU thực thi.



Đặc điểm của Java (tt)

- **Hiệu suất cao** (high performance):

bytecode → native machine code dễ dàng nhờ Just-in-time compiler.

- **Đa luồng** (multithreading)

Cho phép lập trình đa luồng (nhiều chương trình đồng hành nhờ lớp Thread : khởi tạo, ngưng 1 luồng, kiểm tra trạng thái của luồng)

thread: một luồng thực thi của CPU → là 1 chương trình

- **Linh động** (dynamic): Cho phép tương thích với sự thay đổi của môi trường, Trong CT java có các thông tin run-time → Kiểm tra truy xuất lớp an toàn, → an toàn để liên kết caùc lớp vào CT → dynamic



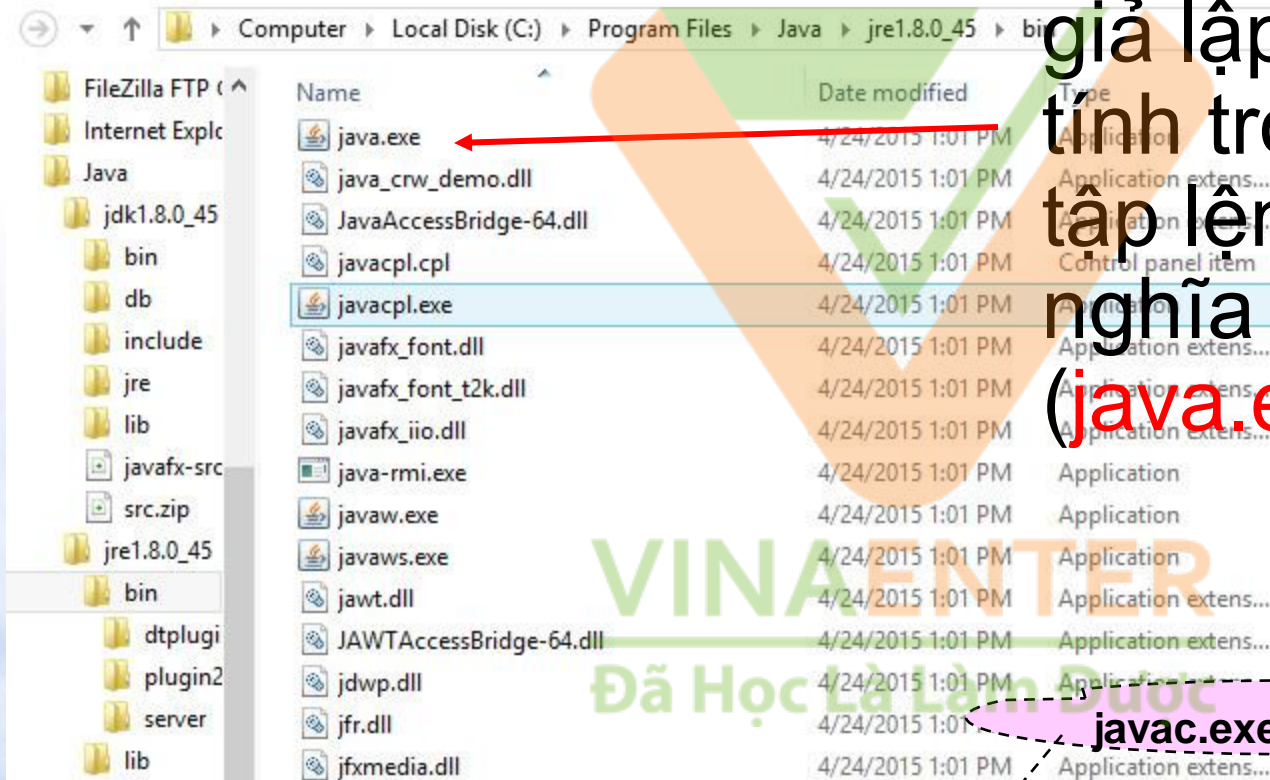
JVM- Java Virtual Machine

- 5 thành phần của môi trường Java
 - (1) Java language
 - (2) Bytecode definitions
 - (3) Java/ Sun Class libraries
 - (4) The Java Virtual Machine
 - (5) The structure of .class file
- JVM là trung tâm của Java
- Các thành phần dẫn đến sự thành công của Java: Bytecode definitions, the structure of .class file, JVM.

JVM là gì?

VinaENTER

- Là một phần mềm giả lập một máy tính trong đó : có tập lệnh định nghĩa các tác vụ (**java.exe**)



Chương trình
Java (file.java)

javac.exe : compiler

Dùng cơ chế Just-In-Time thông dịch
bytecode thành lệnh
máy cụ thể

platform

Java Bytecode (file.class)

JVM (java.exe)

OS

Hardware




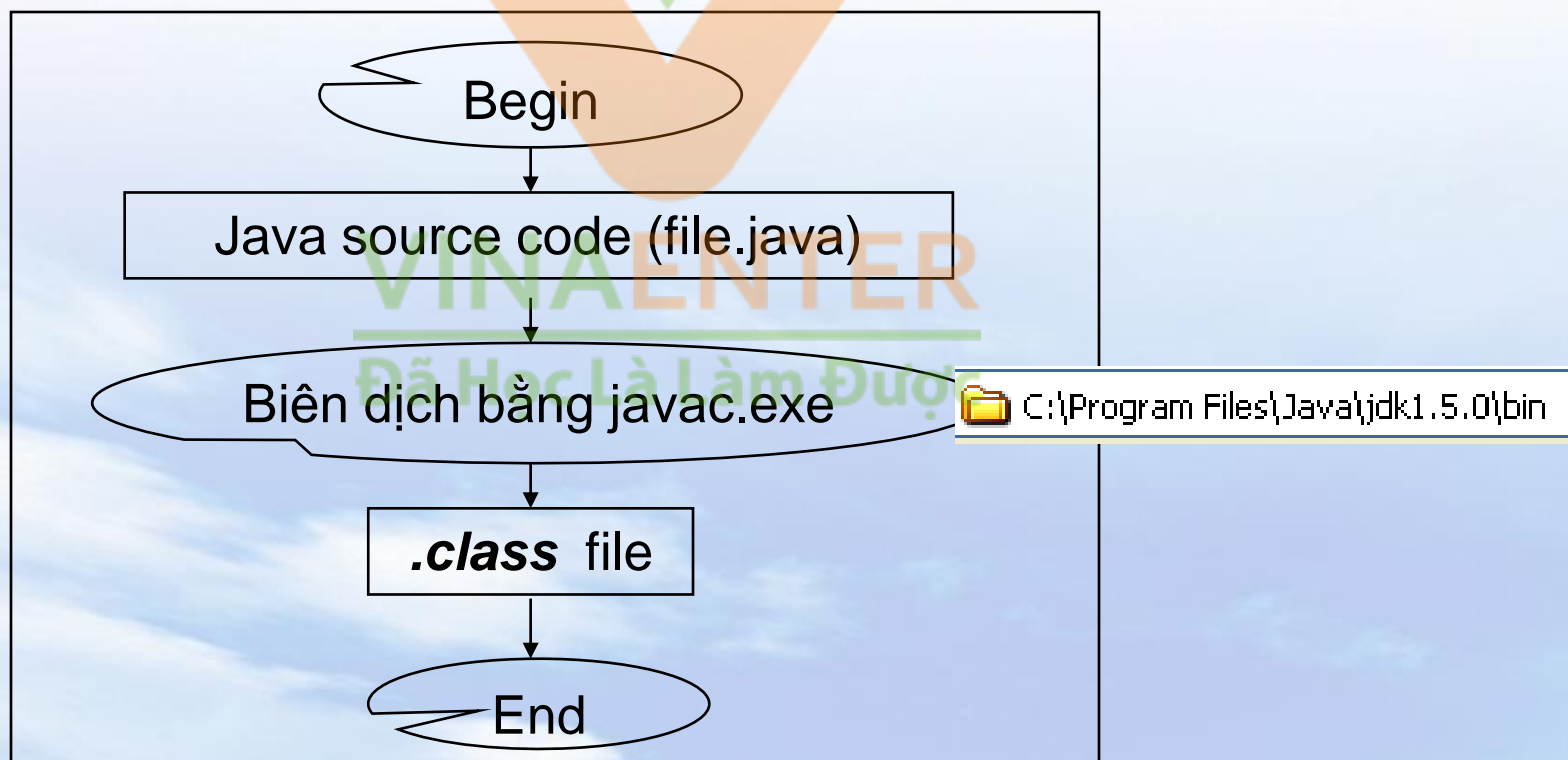
JVM là gì? (tt)

- JVM tạo ra 1 hệ thực thi phụ thuộc platform bao gồm các tác vụ:
 - (1) Nạp *.class* file
 - (2) Quản lý bộ nhớ
 - (3) Thực thi gom rác
- Vì sự không tương thích phần cứng, JVM dùng cơ chế máy-stack chứa các thông tin sau:
 - (1) Các frame (khung thông tin) biểu diễn trạng thái các method.
 - (2) Các toán hạng bytecode
 - (3) Các tham số được truyền cho các method
 - (4) Các biến cục bộ
- Dùng thanh ghi lệnh ghi nhớ địa chỉ lệnh đang thực thi.



JRE-Môi trường run-time

- JRE: Java Run-time Environment.
- Các công cụ được chứa trong thư mục  C:\Program Files\Java\jre1.5.0\bin
- Hai giai đoạn của 1 Java application: Compile-time, Run-time
- Compile-time phase: Viết và biên dịch chương trình

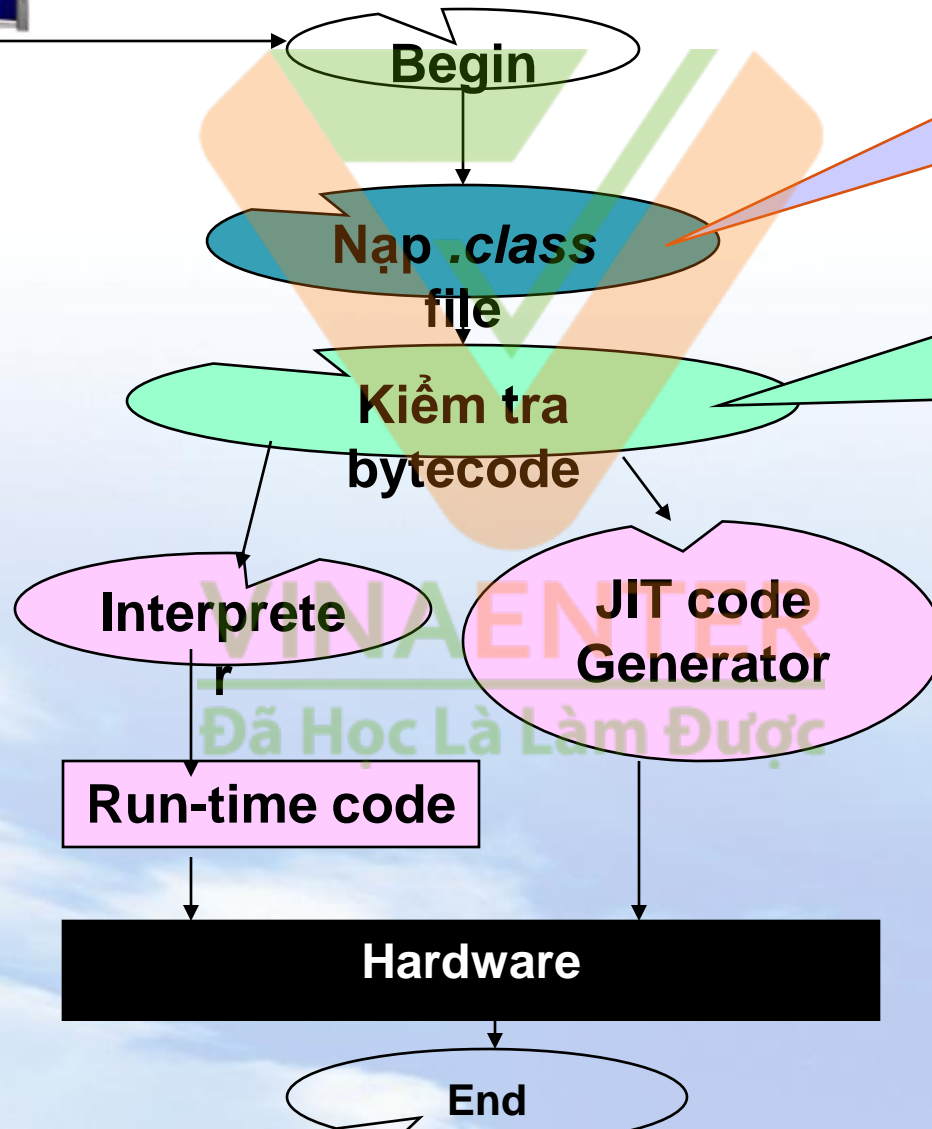


JRE- Run-time phase

VinaENTER

.class
file

Net
work



Nhờ class
Loader, kiểm tra
an toàn

Nhờ chức năng
bytecode
verifier, kiểm tra
code format và
quyền truy xuất

Interprete

JIT code
Generator

Run-time code

Hardware

End



Trình gom rác- Garbage Collection

- **Heap**: Vùng nhớ chia sẻ thông tin giữa các quá trình. Với C, C++, Pascal, programmer phải tự quản lý vùng nhớ cấp phát động này bằng các hàm cơ bản.
- **Cơ chế quản lý heap**

Heap được quản lý bằng 2 danh sách:

Free block list và **Allocated Block List**.

- Cách cấp phát: **“first-fit block”**
- Khi khối bộ nhớ được yêu cầu lớn hơn khả năng của các khối tự do: **Compaction** - dồn vùng nhớ để tạo ra vùng lớn hơn.
- **Heap trong Java** : 2 heap
heap cấp phát tĩnh và heap cấp phát động.

Trình gom rác (tt)

Dynamic heap: có gom rác

Static Heap

-Class definitions

-Các hằng

-CMT1

(class1, method1, Add1)

-CMT2

(class2, method2, Add2)

.....

Static heap: không gom rác

CMT: class method table

Dynamic heap Section 1

Biến đối tượng O2

Biến đối tượng O1

Dynamic heap Section 2

(Các entry: 2 pointers)

(O1, CMT1)

(O2, CMT2)

Section 2: Theo dõi hoạt động của các đối tượng



Cơ chế gom rác

Cơ chế cấp bộ nhớ

- 1/ Nhận yêu cầu cấp bộ nhớ
- 2/ **if** (*Free-Block list đủ*) cấp bộ nhớ cho yêu cầu (First-fit)
- 3/ **else if** (*máy rảnh*) thực thi gom rác
- 4/ **else** ứng dụng phải gọi tường minh tác vụ gom rác: `System.gc()`;
Trình gom rác được ấn định độ ưu tiên rất thấp → Gọi tường minh có ý nghĩa chấp nhận ứng dụng này tạm dừng để chờ gom rác.

Cơ chế gom rác (chỉ gom rác ở Dynamic heap)

- 1/ Xem đối tượng nào không có entry trong section2 → Không còn dùng đối tượng này nữa.
- 2/ Garbage Collector sẽ gọi method `finalize()` để thu tài nguyên của đối tượng (file, stream kết hợp, bộ nhớ)

Môi trường lập trình Java

A blue rectangular sign with a white arrow pointing right. The word "VinaENTER" is written in orange and blue text on the sign.

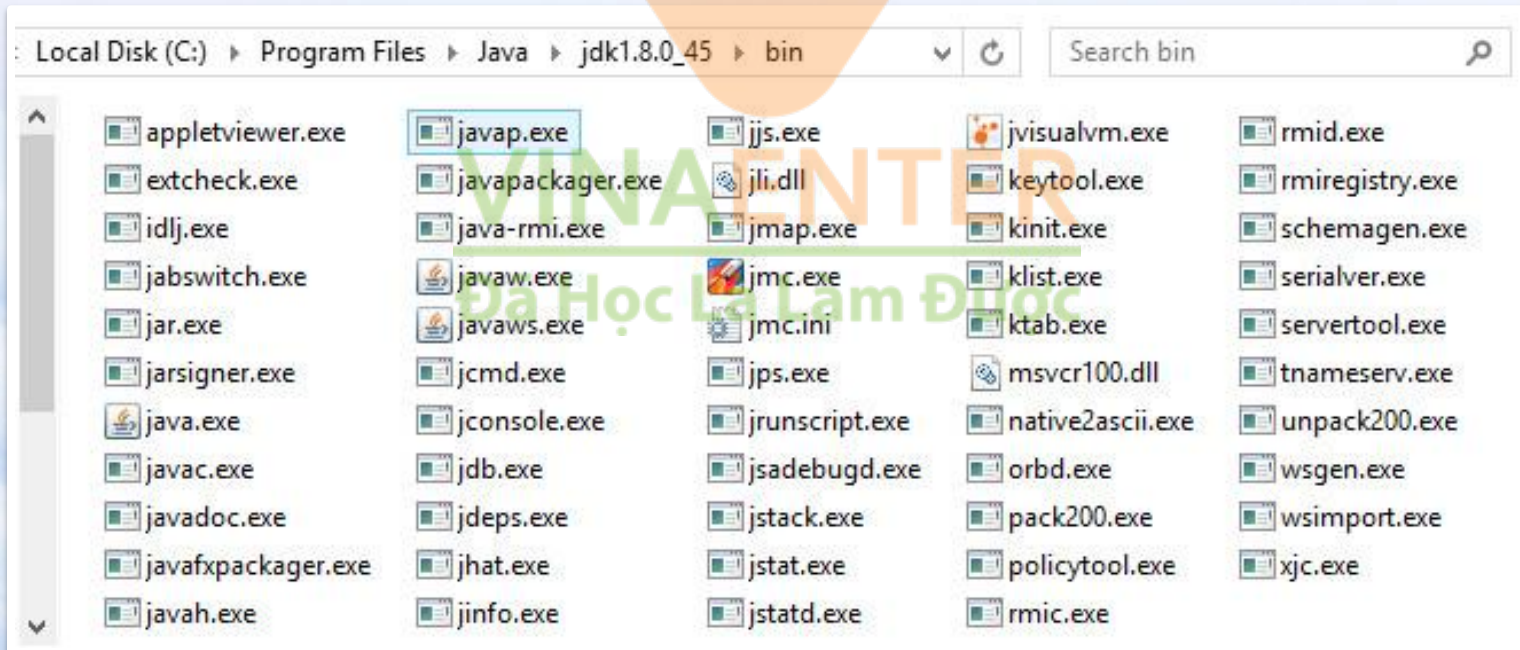
VinaENTER

- **JDK**- Java Development Kit- Bộ công cụ phát triển ứng dụng Java bao gồm 4 thành phần:
 - (1) Classes
 - (2) Compiler
 - (3) Debugger
 - (4) Java Runtime Environment

Các công cụ chính của môi trường Java

Trong thư mục BIN của JDK (sau khi cài đặt) có:

- **javac** : Java Compiler:
Dịch source code → Independent Bytecode
- **java** : Thực thi class file trong JVM
- **appletviewer** : cho phép chạy applet mà không cần Browser.



Chương trình JAVA đầu tiên

```
1  /* Tên file : Hello.java
2   Tác giả : VinaENTER */
3
4  class Hello
5  {
6      // Phương thức main, điểm bắt đầu của chương trình
7      public static void main( String args[ ] )
8      {
9          System.out.println( "Hello World" );
10     } // Kết thúc phương thức main
11 } //
```

Tên lớp chứa hàm main phải giống tên file

Điểm bắt đầu và kết thúc của lớp

Dấu hiệu chú thích =>

Làm cho chương trình dễ hiểu hơn. Trình biên dịch sẽ bỏ qua những dòng có dấu chú thích

Khai báo lớp

Mỗi CT phải có ít nhất một khai báo lớp

Hiển thị dãy ký tự ra màn hình

Phương thức main() sẽ được gọi đầu tiên. Mỗi CT thực thi phải có một phương thức main()

Các câu lệnh phải kết thúc bằng dấu chấm phẩy

Phương thức main()

```
3 package vinaenter.edu;
4 public class VinaenterEdu {
5     public static void main(String[] args) {
6         System.out.println("Hello World");
7     }
8 }
9 //package: dùng để chỉ class mà chúng ta đang
10 ///////////////có nằm trong package đó
11 //println(): phương thức của class System.out
```



Lệnh in chuỗi

- Để in chuỗi hoặc biến trong java, dùng lệnh:

System.out.print("Chuỗi cần in");

hoặc:

System.out.println("Chuỗi cần in");

Đã Học Là Làm Được