



# NỘI DUNG

## LẬP TRÌNH HĐT

*Chương 1: Tổng quan về lập trình HĐT*

*Chương 2: Giới thiệu về lập trình Java*

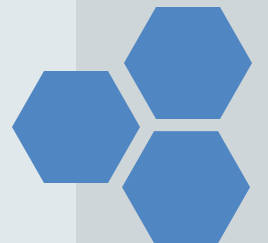
*Chương 3: Lập trình Java cơ sở*

*Chương 4: Lập trình hướng đối tượng với Java*

*Chương 5: Xử lý ngoại lệ*



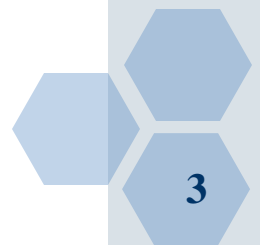
# CHƯƠNG 2: GIỚI THIỆU VỀ LẬP TRÌNH JAVA





# GIỚI THIỆU VỀ LẬP TRÌNH JAVA

- ❖ **Lịch sử phát triển Java**
- ❖ Cách biên dịch và thông dịch trong Java
- ❖ Môi trường lập trình Java
- ❖ Cấu trúc tệp chương trình Java
- ❖ Thực hiện chương trình Java





# Lịch sử phát triển Java

## Java là gì?

- ❖ **Java** là một ngôn ngữ lập trình **HĐT** được phát triển bởi **Sun Microsystems**.
- ❖ Là ngôn ngữ vừa **biên dịch** vừa **thông dịch** vì:
  - Nó có khả năng chạy thống nhất trên nhiều nền tảng.
  - Chỉ cần biên dịch một lần.

WRITE ONE, RUN ANY  
WHERE





# Lịch sử phát triển Java

- ❖ Java là một ngôn ngữ lập trình khá trẻ.
- ❖ Xuất hiện năm 1992, của tập đoàn Sun Microsystems để xây dựng ứng dụng điều khiển các bộ xử lý bên trong điện thoại cầm tay, các thiết bị điện tử dân dụng...
- ❖ 6/1995: Java được giới thiệu chính thức là ngôn ngữ lập trình được xây dựng trên nền tảng C à C++
- ❖ Năm 1996 ban hành bản Java Development Kit 1.0 miễn phí.
- ❖ Ban đầu, Java chủ yếu dùng để phát triển các Applet, các ứng dụng nhúng vào trình duyệt web. Nhưng sau được phát triển rộng rãi





# Lịch sử phát triển Java

- ❖ Ngày nay, nhắc đến Java không còn nhắc đến như một ngôn ngữ lập trình mà còn là một công nghệ, một nền tảng phát triển.
- ❖ Java có một cộng đồng phát triển mạnh mẽ.
  - Một tập hợp các thư viện với số lượng lớn (từ Sun và các nguồn khác).



# JSE, JEE, JME

- ❖ JSE (Java Standard Edition): platform cho phép phát triển và thực thi các ứng dụng Java trên desktop và server, các thiết bị nhúng và môi trường thời gian thực.
- ❖ JEE (Java Enterprise Edition): platform cho phép phát triển các ứng dụng Java chạy trên server.
- ❖ JME (Java Micro Edition): platform cho phép phát triển các ứng dụng Java chạy trên các thiết bị di động

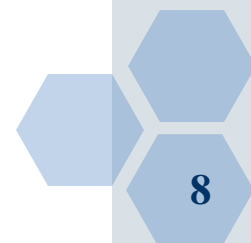




# Lịch sử phát triển Java

## ❖ Các phiên bản của Java:

Năm	Phiên bản	Năm	Phiên bản
1996	JDK 1.0	2004	J2SE5
1997	JDK 1.1	2006	Java SE6
1998	J2SE 1.2	2010	JDK 6.18
2000	J2SE 1.3	2011	Java SE7
2002	J2SE 1.4	2014	JDK 8
		2017	JDK 9

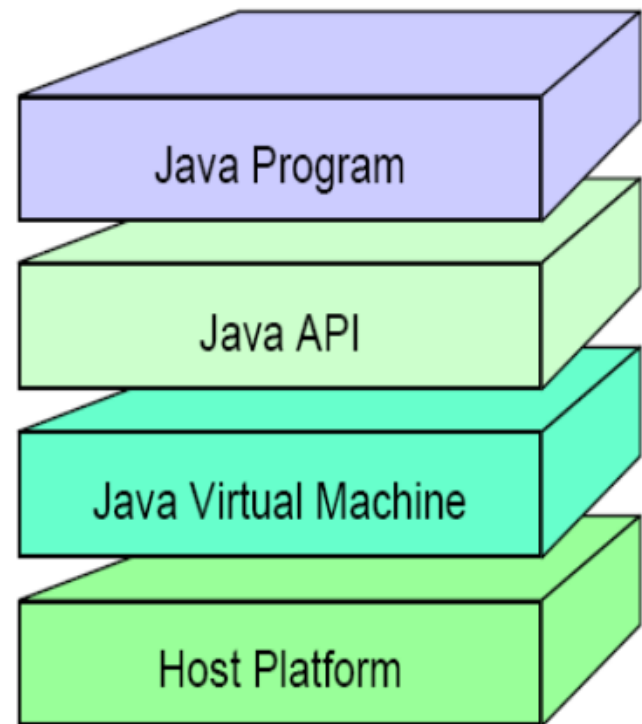






# Nền tảng của Java (Java platform)

- ❖ **Platform:** là môi trường phát triển hoặc triển khai.
- ❖ **Java platform** có thể chạy trên mọi hệ điều hành.
  - Các platform khác phụ thuộc vào phần cứng.
  - Java platform cung cấp:
    - Máy ảo Java (JVM)
    - Giao diện lập trình ứng dụng (API)





# Đặc điểm Java

- ❖ ***Đơn giản, thân thiện:*** cú pháp giống C và hướng đối tượng giống C++ nhưng loại bỏ đi những đặc trưng phức tạp của C++.
- ❖ ***Hướng đối tượng:*** hỗ trợ tất cả các đặc trưng của hướng đối tượng như: class, object, đóng gói, kế thừa, đa hình, nạp chồng, ghi đè...
- ❖ ***Độc lập với nền tảng*** (phần cứng và hệ điều hành): khi biên dịch thì code chuyển sang file \*.class (file chứa mã bytecode để máy ảo Java thực thi). File này có thể chạy ở bất kỳ đâu có máy ảo Java.



# Đặc điểm Java

- ❖ ***Mạnh mẽ, an toàn:*** java kiểm soát chặt chẽ các vấn đề: kiểu dữ liệu phải tường minh, không hỗ trợ con trỏ, đảm bảo không tràn mảng, tự động cấp phát và giải phóng bộ nhớ thông qua bộ thu dọn rác.
- ❖ ***Lập trình mạng, phân tán:*** Java có thể phát triển các ứng dụng Web, phân tán thông qua gói Java.net và công nghệ J2EE.
- ❖ ***Thực hiện đa luồng:*** Java hỗ trợ thực hiện đa luồng và có đầy đủ các cơ chế thực hiện đồng bộ giữa các luồng.



# Ứng dụng của Java

- ❖ **Ứng dụng Applet:** Applet được nhúng bên trong trang Web, khi trang web hiển thị trong trình duyệt, applet sẽ được tải về và thực thi tại trình duyệt.
- ❖ **Ứng dụng dòng lệnh console:** chương trình chạy từ dấu nhắc lệnh và không dùng giao diện đồ hoạ.
- ❖ **Ứng dụng đồ hoạ:** Là chương trình Java chạy độc lập cho phép người dùng tương tác qua giao diện đồ hoạ.
- ❖ **Ứng dụng cơ sở dữ liệu:** các ứng dụng sử dụng JDBC API để kết nối tới CSDL.
- ❖ **Ứng dụng cho mạng, phân tán hay các thiết bị di động...**



# GIỚI THIỆU VỀ LẬP TRÌNH JAVA

- ❖ Lịch sử phát triển Java
- ❖ **Cách biên dịch và thông dịch trong Java**
- ❖ Môi trường lập trình Java
- ❖ Cấu trúc tệp chương trình Java
- ❖ Thực hiện chương trình Java



- 
- ```
graph LR; A[Tập cái  
(Header File)] --> D[Tập mã đích  
(Object File)]; B[Tập thư viện  
(Object File)] --> E[Tập chương trình thực hiện  
(Executable File)]; C[Tập chứa chương trình chính  
main()  
{ } ] --> D; D -- "Dịch" --> E; E -- "Liên kết" --> F[Tập chương trình thực hiện  
(Executable File)]; F -- "Kiểm tra (Test)" --> G[Chương trình ứng dụng  
hiện thời]; G -- "Gỡ lỗi (Debug)" --> C;
```
- The flowchart illustrates the C programming compilation process. It starts with three input files: 'Tập cái (Header File)', 'Tập thư viện (Object File)', and 'Tập chứa chương trình chính main() { }'. The 'Tập cái (Header File)' and 'Tập chứa chương trình chính main() { }' are grouped together in a dashed oval. Arrows from these two files point to 'Tập mã đích (Object File)'. An arrow labeled 'Dịch' (Compile) points from 'Tập mã đích (Object File)' to 'Tập chương trình thực hiện (Executable File)'. An arrow labeled 'Liên kết' (Link) points from 'Tập thư viện (Object File)' to 'Tập chương trình thực hiện (Executable File)'. From 'Tập chương trình thực hiện (Executable File)', an arrow labeled 'Kiểm tra (Test)' points to 'Chương trình ứng dụng hiện thời' (Current application program). Finally, an arrow labeled 'Gỡ lỗi (Debug)' points from 'Chương trình ứng dụng hiện thời' back to 'Tập chứa chương trình chính main() { }'.

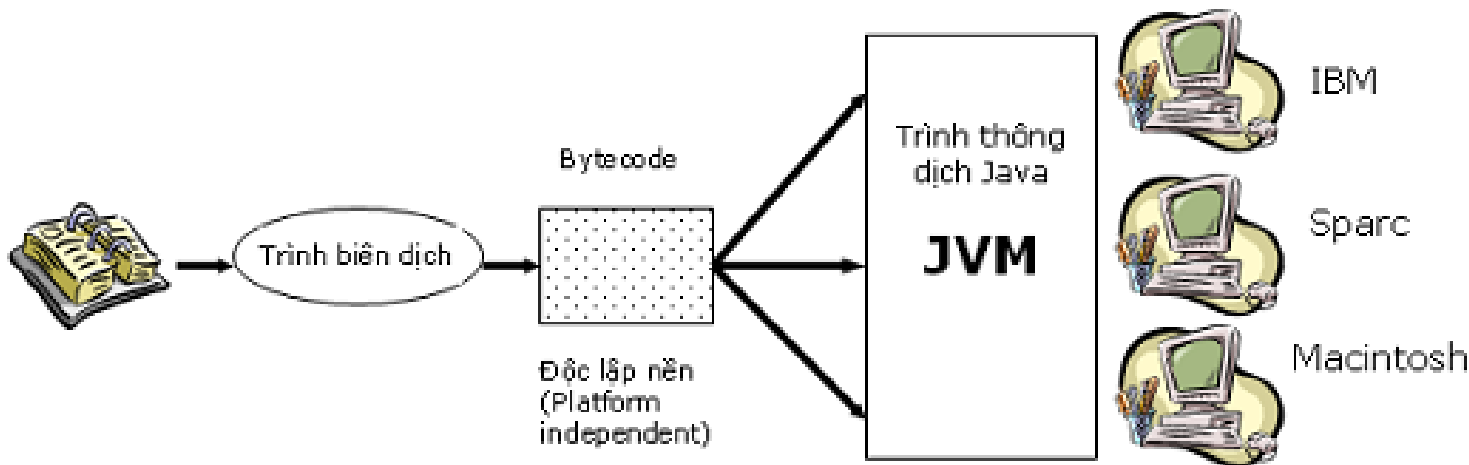


# Mô hình dịch và thông dịch của Java

- ❖ Mã nguồn được biên dịch thành bytecode rồi được thông dịch bởi JVM

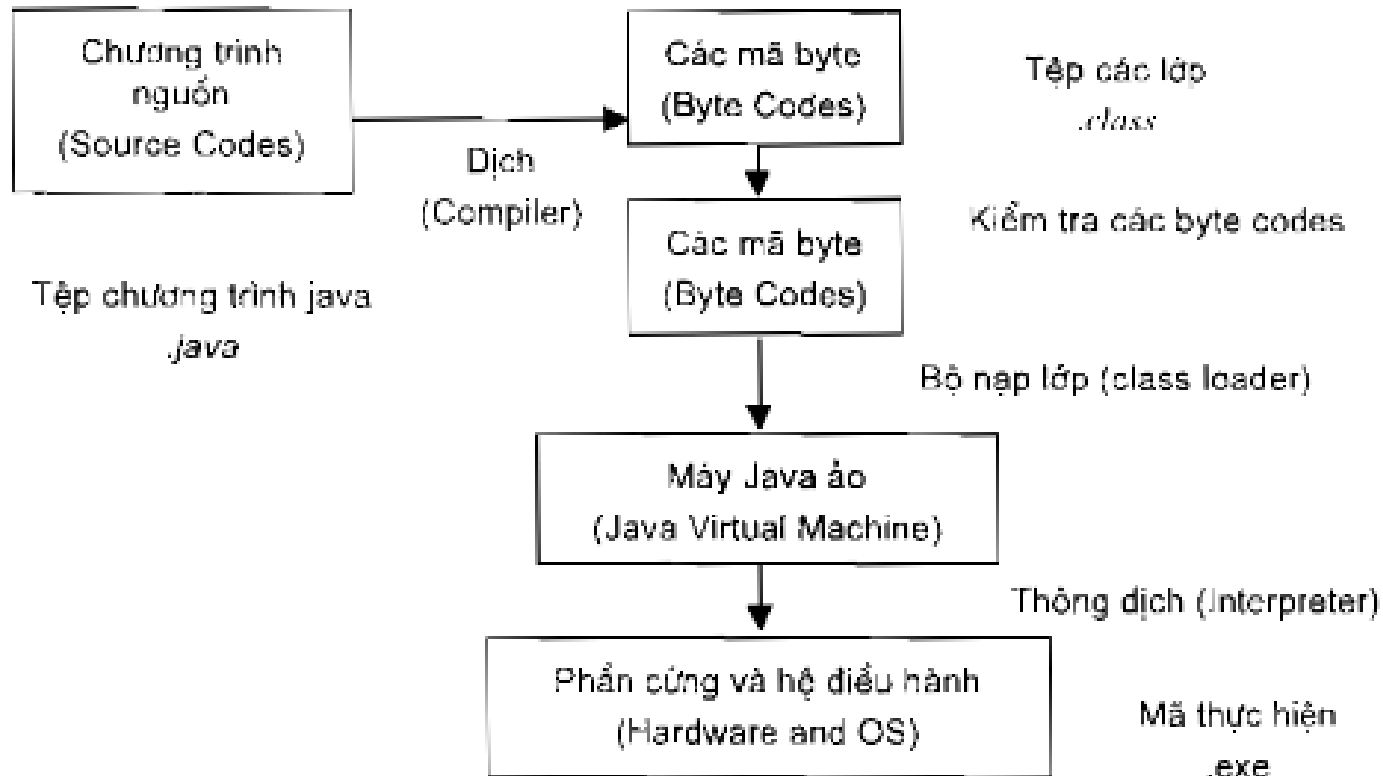
```
%javac Hello.java
Hello.class created
% java Hello
```

Run JVM Byte Code





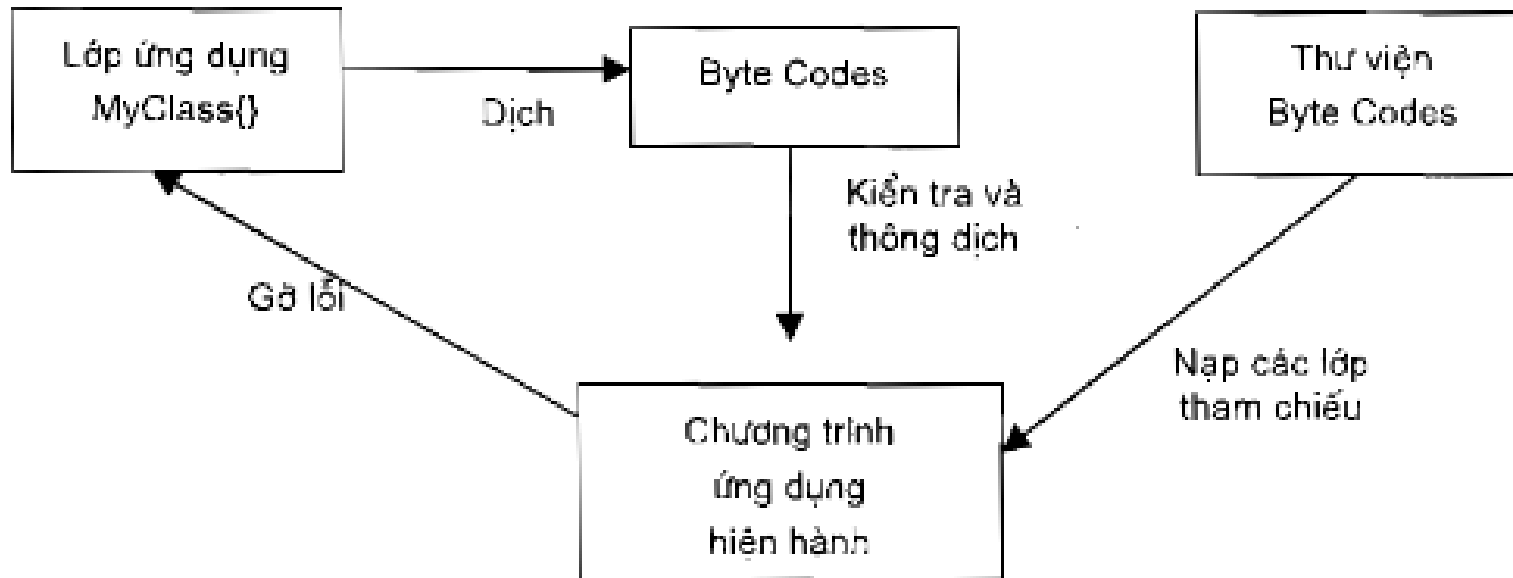
# Mô hình dịch và thông dịch của Java







# Quá trình phát triển chương trình Java





# GIỚI THIỆU VỀ LẬP TRÌNH JAVA

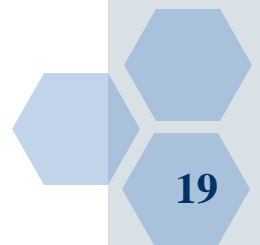
- ❖ Lịch sử phát triển Java
- ❖ Cách biên dịch và thông dịch trong Java
- ❖ **Môi trường lập trình Java**
- ❖ Cấu trúc tệp chương trình Java
- ❖ Thực hiện chương trình Java



# Môi trường lập trình Java

## ❖ **Môi trường Java bao gồm 5 phần tử sau:**

- Ngôn ngữ
- Định nghĩa byte code
- Các thư viện lớp Java/Sun
- Máy ảo java (JVM)
- Cấu trúc của file \*.class





# Môi trường lập trình Java

- ❖ **Java** có thể dùng được trong nhiều hệ điều hành như **Windows, Linux, MAC OSX...**
- ❖ Để làm việc được với **Java** thì ta cần thiết lập môi trường làm việc cho máy tính bao gồm:
  - Cài đặt **JDK** (Java Development Kit): bộ công cụ phát triển ứng dụng bằng ngôn ngữ lập trình **Java**.
  - Cài đặt **IDE** (*Integrated Development Enviroment*: Môi trường phát triển tích hợp.p) như Netbean, Eclipse



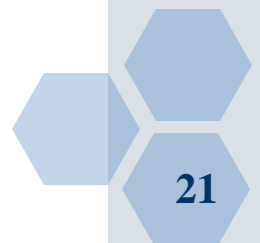
# Môi trường lập trình Java

## Cài đặt JDK trong Windows:

**B1:** Download JDK tại:

**<http://www.oracle.com/technetwork/java/javase/downloads/index.html>**

**B2:** Tiến hành cài đặt.





# JDK, JRE là gì

- ❖ **JDK:** là bộ công cụ phát triển ứng dụng bằng ngôn ngữ lập trình Java. Trong JDK gồm:
  - **Javac:** trình biên dịch mã nguồn sang byte code.
    - Cú pháp: `javac sourcecodename.java`
  - **Java:** trình thông dịch: nó thực thi các ứng dụng độc lập và các file \*.class
    - Cú pháp: `java classname`
  - **Javadoc:** bộ sinh tài liệu
    - Cú pháp: `javadoc sourcecodename.java`



# JDK, JRE là gì

- ***Jdb (Java debugger)***: chương trình tìm lỗi
    - Cú pháp: `jdb sourcecodename.java`
    - Hoặc `jdb-host-password sourcecodename.java`
  - ***Javap***: trình dịch ngược
    - Cú pháp: `javap classname`
  - ***Appetviewer***: chương trình xem applet
    - Cú pháp: `appletviewer sourcecodename.java/url`
- ❖ **JRE**: là ứng dụng nền giúp thực thi các file mã máy byte code sau khi đã được JDK biên dịch.



# Thiết lập biến MT chạy Java trong command line

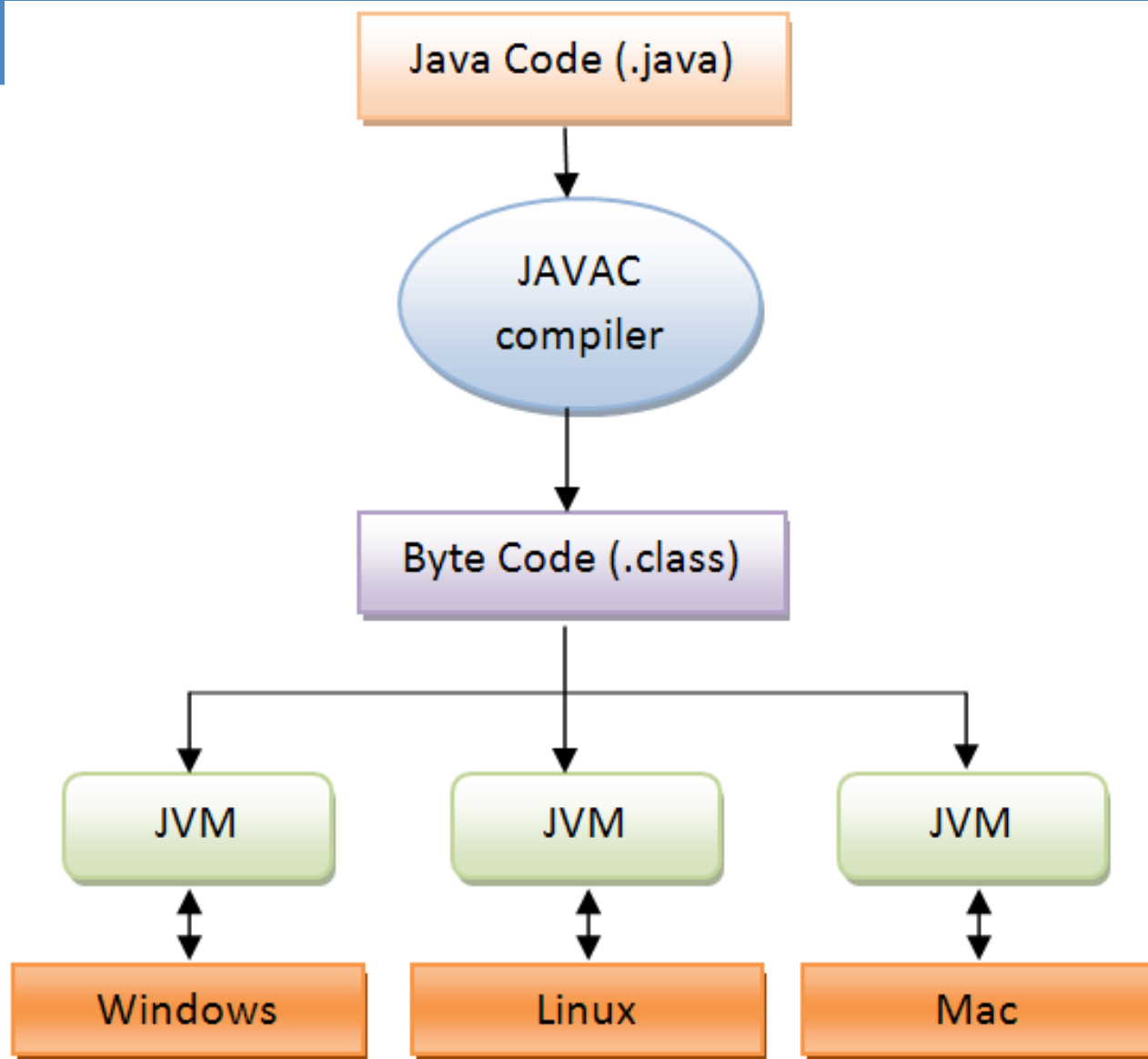
- ❖ Lập trình trên notepad hoặc notepad++ và chạy bằng cmd.
  - B1: click chuột phải vào my computer => Properties sẽ xuất hiện cửa sổ system.
  - B2: Chọn Advanced system settings sẽ xuất hiện cửa sổ System properties.
  - B3: Trong tab Advanced chọn Environment Variables sẽ xuất hiện cửa sổ Environment Variables.
  - B4: Thêm giá trị với biến path trong mục System variable chọn Edit





# Thiết lập biến MT chạy Java trong command line

- ❖ Path: Chứa danh sách các thư mục mà chương trình sẽ tìm kiếm cho file thực thi tương ứng với tên lệnh được đưa ra bởi người dùng.
- ❖ Xuất hiện cửa sổ edit, trong mục Variable value, di chuyển tới cuối và nhập vào đường dẫn tới JDK như:  
C:\Program Files\Java\jdk1.7.0\_25\bin =>OK.
- ❖ Kiểm tra lại cấu hình.





# Cài đặt IDE (Eclipse)

- ❖ **Eclipse** là một công cụ dành cho lập trình viên để viết, biên dịch, gỡ lỗi và triển khai chương trình.
  - B1: Download bộ cài **Eclipse**:  
<http://www.eclipse.org/downloads/>
  - B2: Tiến hành cài đặt



# Máy ảo Java JVM (Java Virtual Machine)

## ❖ Máy ảo Java là trái tim của ngôn ngữ Java

- Đem đến cho các chương trình Java khả năng viết một lần nhưng chạy được ở mọi nơi.

## ❖ Tạo ra môi trường bên trong để thực thi lệnh:

- Nạp các file \*.class
- Quản lý bộ nhớ
- Dọn rác

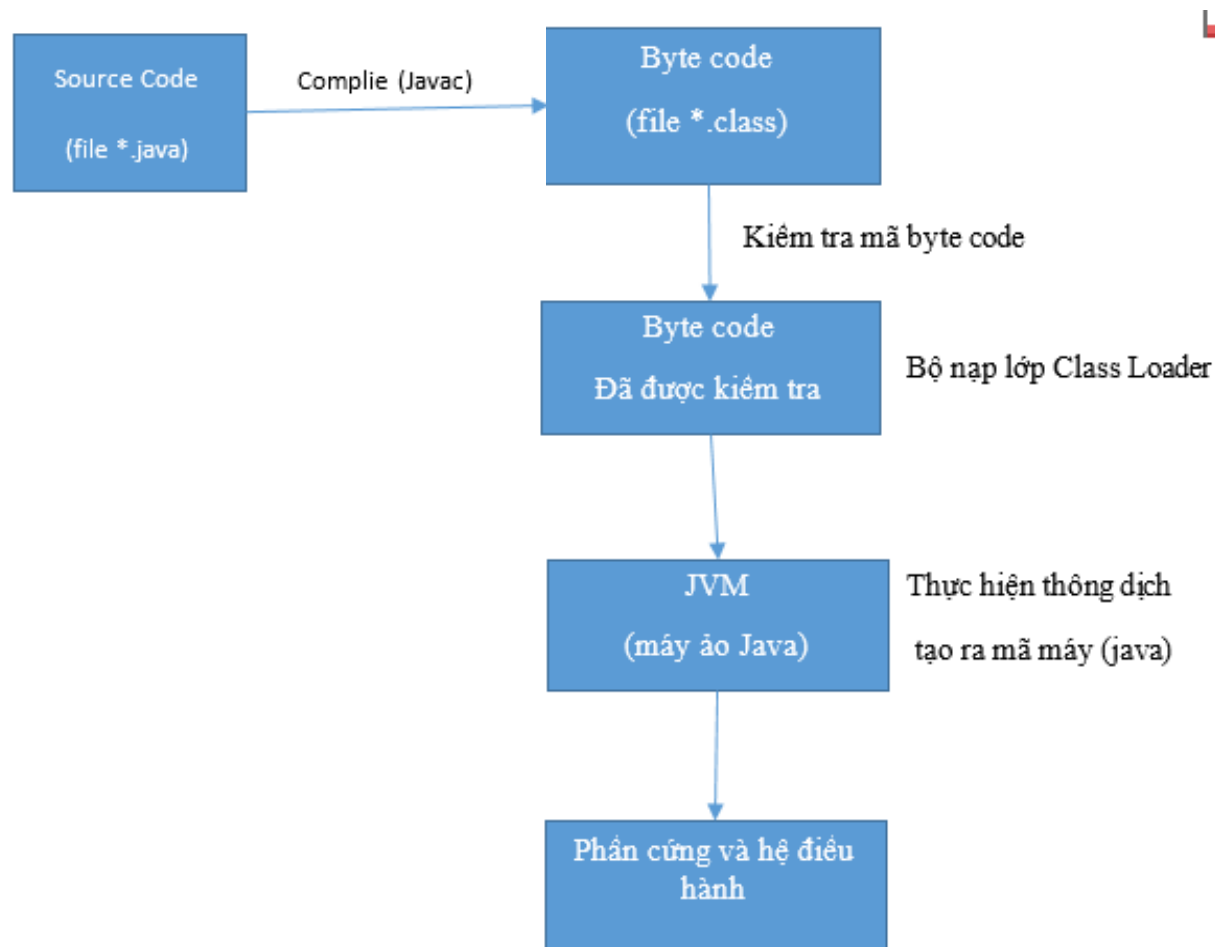
Trình thông dịch “Just In Time-JIT”

- Chuyển tập lệnh bytecode thành mã máy cụ thể cho từng loại CPU.





# Máy ảo Java JVM





# GIỚI THIỆU VỀ LẬP TRÌNH JAVA

- ❖ Lịch sử phát triển Java
- ❖ Cách biên dịch và thông dịch trong Java
- ❖ Môi trường lập trình Java
- ❖ **Cấu trúc tệp chương trình Java**
- ❖ Thực hiện chương trình Java



# Cấu trúc tệp chương trình Java

## //Filename: NewApp.java

```
//Phần 1: Tùy chọn  
//Định nghĩa gói  
Package GoiNhaTruong
```

```
//Phần 2: (0 hoặc nhiều hơn)  
//các gói cần sử dụng  
Import java.io.*
```

```
//Phần 3: (0 hoặc nhiều hơn)  
//Định nghĩa các lớp và các interface  
Public class NewApp{...}  
Class C1{...}  
interface I1{...}  
//  
Class Cn{...}  
Interface Im{...}
```



# Cấu trúc tệp chương trình Java

## ❖ *Lưu ý:*

- Tệp chương trình Java luôn có tên trùng với tên của một lớp công khai (lớp chứa hàm *main()* nếu là ứng dụng độc lập) và có đuôi là *.java*.
- Tệp *NewApp.java* nếu là chương trình ứng dụng độc lập thì có một lớp có tên là *NewApp* và lớp này phải có phương thức *main()*. PT này luôn có dạng

```
Public static void main(String args[]){  
//Nội dung cần thực hiện của chương trình ứng dụng  
}
```
- Mỗi lớp trong tệp chương trình sẽ được dịch thành byte code và được ghi thành tệp riêng có tên trùng với tên của lớp và có đuôi *.class*. Các lớp sẽ được nạp vào chương trình lúc thông dịch và thực hiện theo yêu cầu.
- Ko có các khai báo biến, hàm tách biệt khỏi lớp và chỉ có các khai báo và định nghĩa các lớp, interface







# Các dạng chương trình ứng dụng của Java

## ❖ Có ba loại chương trình có thể phát triển với Java:

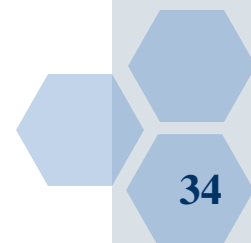
- Các chương trình ứng dụng độc lập
- Các chương trình ứng dụng nhúng (applet)
- Các chương trình kết hợp cả 2 loại trên



# Chương trình ứng dụng độc lập

## ❖ Có ba loại chương trình có thể phát triển với Java:

- Phải định nghĩa một lớp có chứa phương thức main ()
- Khi xây dựng cần:
  - Tạo lớp (lớp chính) có phương thức main()
  - Kiểm tra xem tệp chương trình có tên trùng với tên của lớp chính và đuôi “.java”?
  - Dịch tệp chương trình để tạo ra các tệp mã byte code với các đuôi “.class” tương ứng.
  - Sử dụng thông dịch để chạy chương trình đã dịch





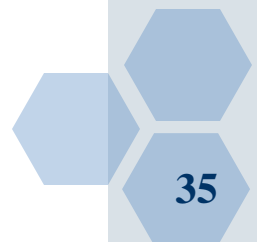
# Chương trình ứng dụng nhúng

- ❖ Được đưa vào trang tư liệu theo dạng HTML để sau đó nạp được xuống thông qua Web Browser hoặc appletviewer của JDK
- ❖ Thông tin cần phải có đối với applet:

- Tên của tệp lớp applet
- Kích cỡ của applet tính theo pixel
- Ví dụ:

```
<applet code =“AppletApp.class” width = 200 height = 300>
```

```
</applet>
```





# So sánh chương trình ứng dụng và applet

	Ứng dụng độc lập	Java Applet
Khai báo	Là lớp con của bất kỳ lớp nào trong các gói thư viện các lớp	Phải là lớp con của Applet
Giao diện đồ họa	Tùy chọn	Do trình duyệt Web quyết định
Yêu cầu bộ nhớ	Bộ nhớ tối thiểu	Bộ nhớ dành cho cả trình duyệt và applet đó
Cách nạp chương trình	Nạp dòng lệnh	Thông qua trang Web
Dữ liệu vào	Thông qua các tham số trên dòng lệnh	Các tham số đặt trong tệp HTML gồm địa chỉ, kích thước của trình duyệt
Cách thức thực hiện	Mọi hoạt động được bắt đầu và kết thúc ở main()	Gọi các hàm: init(), start(), stop(), destroy(), paint()
Kiểu ứng dụng	<ul style="list-style-type: none"><li>- Ứng dụng trên các máy chủ server</li><li>- Công cụ phát triển phần mềm</li><li>- Ứng dụng trên các máy khách</li></ul>	Các ứng dụng trên Web



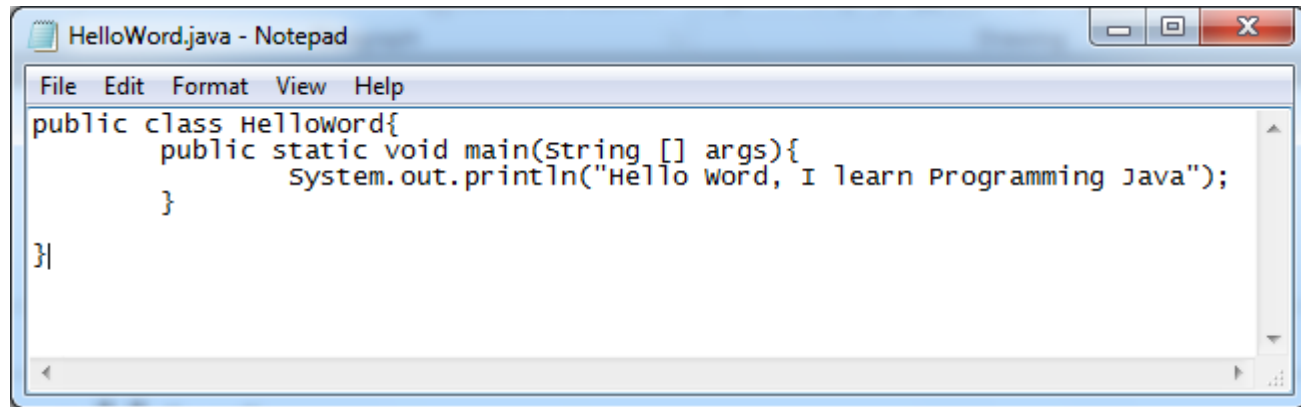
# GIỚI THIỆU VỀ LẬP TRÌNH JAVA

- ❖ Lịch sử phát triển Java
- ❖ Cách biên dịch và thông dịch trong Java
- ❖ Môi trường lập trình Java
- ❖ Cấu trúc tệp chương trình Java
- ❖ **Thực hiện chương trình Java**



# Chương trình Java đơn giản

- ❖ Chương trình viết bằng **Notepad** và chạy bằng lệnh **cmd**.
  - **B1:** mở notepad và gõ lệnh, lưu dưới dạng **tenclass.java**



```
File Edit Format View Help
public class Helloword{
    public static void main(String [] args){
        System.out.println("Hello word, I learn Programming Java");
    }
}
```



# Chương trình Java đơn giản

- ❖ **B2: Vào Run-> gõ cmd và enter.**
  - Gõ `cd c:/Test` sau rồi gõ `javac`.
- ❖ **B3: Nếu báo ko tìm thấy javac thì thực hiện copy đường dẫn đến bin của JDK rồi edit path, paste vào cuối ->ok.**
- ❖ **B4: Tắt đi và gõ cmd lại, gõ javac chạy các file.**
- ❖ **B5: gõ java HelloWorld để thực hiện biên dịch chương trình. Khi đó chương trình sẽ hiển thị nội dung của câu lệnh hiển thị.**



# Chương trình Java đơn giản

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd c:/Test

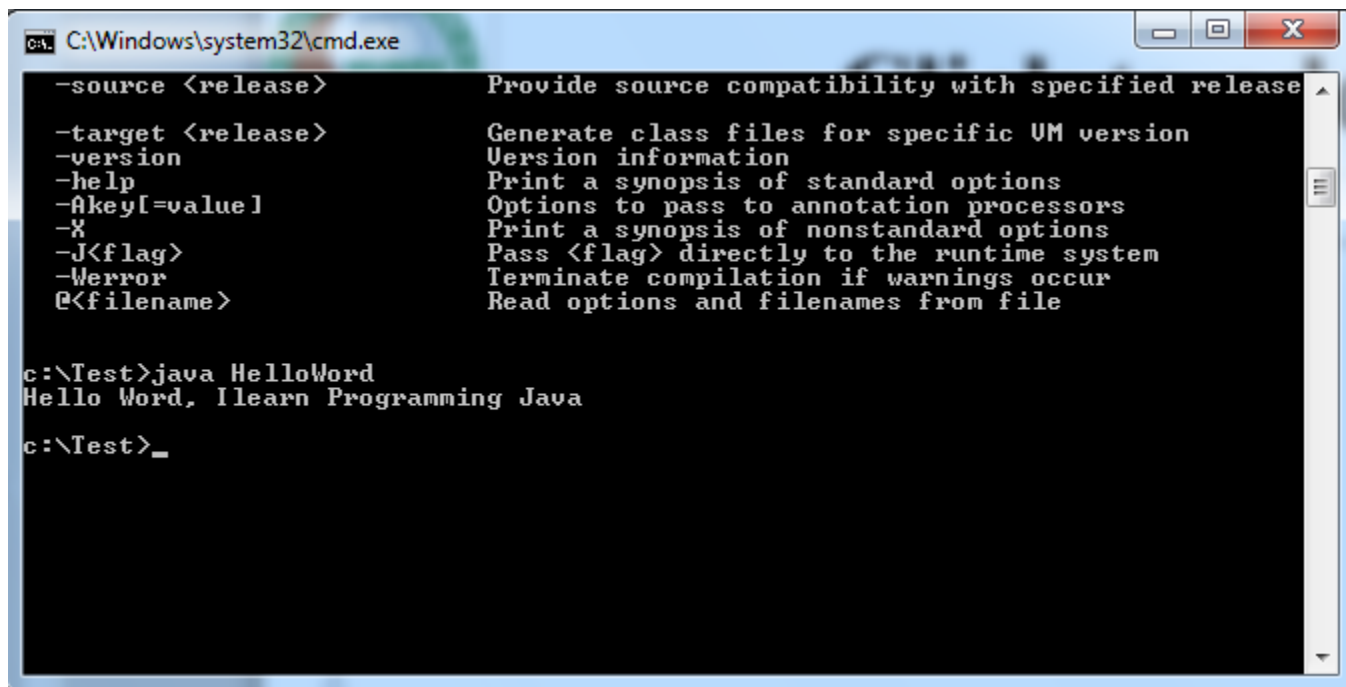
c:\Test>javac
Usage: javac <options> <source files>
where possible options include:
  -g               Generate all debugging info
  -g:none          Generate no debugging info
  -g:{lines,vars,source}  Generate only some debugging info
  -nowarn          Generate no warnings
  -verbose         Output messages about what the compiler is doing
  -deprecation     Output source locations where deprecated APIs are used
  -classpath <path> Specify where to find user class files and annotations
  -cp <path>       Specify where to find user class files and annotations
  -sourcepath <path> Specify where to find input source files
  -bootclasspath <path> Override location of bootstrap class files
  -extdirs <dirs>   Override location of installed extensions
  -endorseddirs <dirs> Override location of endorsed standards path
  -proc:{none,only} Control whether annotation processing and/or compilation is done.
```





# Chương trình Java đơn giản

## ❖ Kết quả



```
C:\Windows\system32\cmd.exe

-source <release>      Provide source compatibility with specified release
-target <release>      Generate class files for specific VM version
-version                Version information
-help                  Print a synopsis of standard options
-Akey[=value]          Options to pass to annotation processors
-X                    Print a synopsis of nonstandard options
-J<flag>               Pass <flag> directly to the runtime system
-Werror               Terminate compilation if warnings occur
@<filename>             Read options and filenames from file


c:\Test>java HelloWorld
Hello Word, Ilearn Programming Java

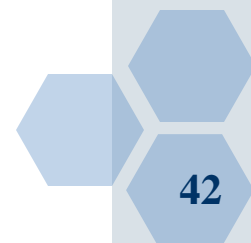
c:\Test>_
```



# Chương trình Java đơn giản

## ❖ Chương trình chạy bằng IDE Eclipse:

- B1: Khởi động IDE Eclipse/File /New/Project Java
- B2: Xuất hiện hộp thoại chọn Java Project
- B3: Đặt tên Project -> Finish.
- B4: New -> Package
- B5: New -> Class
- B6: Xuất hiện cửa sổ source code -> viết lệnh.
- B7: Ctrl + F11





# Chương trình Java đơn giản

The screenshot displays the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons for file operations and development. The left sidebar contains the Outline view and the Package Explorer. The Outline view shows a package named 'helloworld' containing a class 'HelloWord' with a method 'main(String[]): void'. The Package Explorer shows the 'HelloWord' package containing the 'JRE System Library [JavaSE-9]' and the 'src' folder. The main editor window displays the source code for 'HelloWord.java'.

```
1 package helloworld;
2
3 public class HelloWord {
4
5     public static void main(String[] args) {
6         //In ra dòng chữ Hello Word
7         //System.out.println("Hello World!");
8         System.out.println("Hello World!");
9     }
10
11 }
12
13
```

The bottom of the IDE shows the Console view with the output: <terminated> HelloWord [Java Application] C:\Program Files\Java\jre-9.0.4\bin\javaw.exe (Mar 15, 2018, 4:08:53 PM) Hello World!