\* JDK, JRE và JVM có vai trò gì trong hệ sinh thái Java? Hãy phân biệt sự khác nhau giữa ba khái niệm này và nêu mối quan hệ giữa chúng

- Java là một ngôn ngữ lập trình phổ biến, và để chạy hoặc phát triển ứng dụng Java, chúng ta cần hiểu về **JDK, JRE và JVM**. Ba khái niệm này có vai trò quan trọng trong hệ sinh thái Java và có mối quan hệ chặt chẽ với nhau.

| **Tiêu chí** | **JDK (Java Development Kit)** | **JRE (Java Runtime Environment)** | **JVM (Java Virtual Machine)** |
| --- | --- | --- | --- |
| **Chức năng** | Cung cấp công cụ để viết, biên dịch và chạy ứng dụng Java | Cung cấp môi trường để chạy ứng dụng Java | Chạy mã bytecode của Java |
| **Thành phần** | Gồm JRE + Trình biên dịch javac + Công cụ phát triển | Gồm JVM + Thư viện Java | Chỉ có trình thông dịch và công cụ quản lý bộ nhớ |
| **Dành cho ai?** | Lập trình viên muốn phát triển ứng dụng Java | Người dùng chỉ muốn chạy ứng dụng Java | Hệ thống thực thi mã Java ở cấp thấp |
| **Có thể biên dịch mã Java không?** | ✅ Có (javac) | ❌ Không | ❌ Không |
| **Có chứa JVM không?** | ✅ Có | ✅ Có | ✅ Chính là JVM |
| **Có chứa JRE không?** | ✅ Có | ✅ Chính là JRE | ❌ Không |
| **Chạy ứng dụng Java được không?** | ✅ Có | ✅ Có | ✅ Có, nhưng cần JRE để hoạt động |

\* Hãy giải thích khái niệm "Write once, run anywhere" trong Java. Cơ chế nào giúp Java có thể chạy trên nhiều nền tảng khác nhau mà không cần biên dịch lại?

- **"Write Once, Run Anywhere" (WORA)** là một đặc điểm nổi bật của Java, có nghĩa là **chỉ cần viết và biên dịch mã Java một lần**, sau đó chương trình có thể chạy trên **bất kỳ hệ điều hành nào** mà không cần phải biên dịch lại.

Điều này giúp Java trở nên **đa nền tảng**, nghĩa là một ứng dụng Java có thể chạy trên **Windows, macOS, Linux, hoặc bất kỳ hệ điều hành nào** mà không cần sửa đổi mã nguồn hoặc biên dịch lại.

\* Quy trình để cài đặt và chạy chương trình Java đầu tiên bằng IntelliJ IDEA như thế nào? Hãy mô tả các bước cần thực hiện để tạo và chạy một dự án Java cơ bản.

-Dưới đây là bảng chi tiết hướng dẫn **cài đặt và chạy chương trình Java đầu tiên bằng IntelliJ IDEA**:

| **Bước** | **Mô tả chi tiết** |
| --- | --- |
| **Bước 1**: Cài đặt JDK và IntelliJ IDEA | - Tải và cài đặt **JDK** từ [Oracle](https://www.oracle.com/java/technologies/javase-downloads.html" \t "_new) hoặc [Adoptium](https://adoptium.net/" \t "_new).  - Kiểm tra cài đặt JDK bằng lệnh:  java -version và javac -version trong Terminal hoặc CMD.  - Tải và cài đặt **IntelliJ IDEA** từ [JetBrains](https://www.jetbrains.com/idea/download/" \t "_new). |  |
| **Bước 2**: Tạo dự án Java mới | - Mở **IntelliJ IDEA** và chọn **New Project**.  - Chọn **Java** và đảm bảo **Project SDK** đã được thiết lập đúng.  - Đặt tên cho dự án, chọn vị trí lưu trữ, sau đó nhấn **Finish**. |  |
| **Bước 3**: Tạo tệp Java mới | - Mở thư mục **src** trong IntelliJ.  - Nhấn chuột phải vào thư mục **src** → Chọn **New** → **Java Class**.  - Đặt tên lớp, ví dụ: Main. |  |
| **Bước 4**: Viết chương trình Java | - Nhập đoạn mã sau vào Main.java:  java <br> public class Main { <br> public static void main(String[] args) { <br> System.out.println("Hello, Java!"); <br> } <br> } <br> |  |
| **Bước 5**: Chạy chương trình Java | - Nhấn chuột phải vào Main.java và chọn **Run 'Main.main()'**.  - Hoặc nhấn nút **Run ▶** trên thanh công cụ.  - Kết quả sẽ hiển thị trong cửa sổ **Run**:  Hello, Java! |  |
| **Bước 6**: Khắc phục lỗi (nếu có) | - Nếu gặp lỗi **"No JDK found"**, vào **File > Project Structure > Project SDK** để kiểm tra lại.  - Nếu chương trình không chạy, kiểm tra lại cú pháp của phương thức main(). |  |

\* Java có những phiên bản nào? Hãy phân biệt Java SE, Java EE và Java ME về mục đích sử dụng và đối tượng hướng đến

| **Phiên bản** | **Năm phát hành** | **Tính năng chính** |
| --- | --- | --- |
| **Java 1.0 - 1.4** | 1995 - 2002 | Phiên bản đầu tiên, hỗ trợ Applet, Collections Framework (Java 1.2) |
| **Java 5** | 2004 | Hỗ trợ Generics, Enum, AutoBoxing, Annotations |
| **Java 6** | 2006 | Cải thiện hiệu suất, hỗ trợ scripting với Java Compiler API |
| **Java 7** | 2011 | Try-with-resources, Diamond Operator (<>), NIO.2 |
| **Java 8** | 2014 | Hỗ trợ **Lambda Expressions**, **Stream API**, Optional<> |
| **Java 9** | 2017 | Module System (Jigsaw), JShell |
| **Java 10** | 2018 | Biến var, Garbage Collector cải tiến |
| **Java 11 (LTS)** | 2018 | Hỗ trợ var trong Lambda, HttpClient API |
| **Java 17 (LTS)** | 2021 | Sealed Classes, Pattern Matching, Text Blocks |
| **Java 21 (LTS - mới nhất)** | 2023 | Record Patterns, Virtual Threads (Project Loom) |