Java Spring

# ARCHITECTURE



## Test

Tầng này cung cấp khả năng hỗ trợ kiểm thử với JUnit và TestNG.

## Core Container

Bao gồm :

* **Core** : cung cấp những phần cơ bản của framework, bao gồm cả [IoC](#_Ioc_là_gì) và Dependency Injection.
* **Bean** : là những module chính của chương trình, được tạo ra và quản lý bởi Spring IoC container. [Xem thêm](https://viblo.asia/p/tong-quan-ve-spring-bean-WR5JRbZ0JGv)
* **Context** : được xây dựng dựa trên Core và Bean module, nó là phương tiện để truy cập bất kỳ đối tượng nào được xác định và định cấu hình. <Note: có nhiều định nghĩa khác nhau, chưa hiểu rõ>.
* **SpEL** : nó là phần mở rộng của [EL](#_Expression_Language_là) được định nghĩa trong [JSP](#_JavaServer_Pages_là). Nó cung cấp hỗ trợ để thiết lập và nhận các giá trị thuộc tính, gọi phương thức, truy cập các bộ sưu tập và bộ chỉ mục, các biến được đặt tên, toán tử logic và số học, truy xuất các đối tượng theo tên, v.v.

## Data Access/Integration

* **JDBC** : Spring Data JDBC, một phần của họ Spring Data lớn hơn, giúp dễ dàng triển khai các kho lưu trữ dựa trên JDBC. Mô-đun này đề cập đến việc hỗ trợ nâng cao cho các lớp truy cập dữ liệu dựa trên JDBC. Nó giúp việc xây dựng các ứng dụng hỗ trợ Spring sử dụng công nghệ truy cập dữ liệu trở nên dễ dàng hơn.
* **ORM** : cung cấp các lớp tích hợp cho các API ánh xạ quan hệ đối tượng phổ biến, bao gồm [JPA](#_Java_Persistence_API), [JDO](#_Java_Data_Objects), [Hibernate](#_Hibernate_là_gì) và iBatis.
* **OXM** : Object XML Mapping là module cung cấp lớp trừu tượng hỗ trợ triển khai ánh xạ Đối tượng/XML cho JAXB, Castor, XMLBeans, JiBX và XStream.
* **JMS** : chứa các tính năng cho tạo và xóa các message.
* **Transaction :** hỗ trợ quản lý giao dịch chương trình và khai báo cho các lớp implement những interface đặc biệt và cho tất cả các POJOs.

## Web

* **Web : cung cấp các tính năng tích hợp web cơ bản như chức năng upload tập tin nhiều phần dữ liệu và khởi tạo của các container IoC sử dụng để lắng nghe servlet và context ứng dụng web.**
* **Web-MVC :** chứa model-view-controller (MVC) của Spring cho các ứng dụng web.
* **Web-Socket** : cung cấp hỗ trợ cho WebSocket, giao tiếp hai chiều giữa máy khách và máy chủ trong các ứng dụng web.
* **Web-Portlet** : cung cấp việc xử lý MVC sẽ được sử dụng trong một môi trường [portlet](#_Portlet_là_gì) và mô phỏng các chức năng của module Web-Servlet.

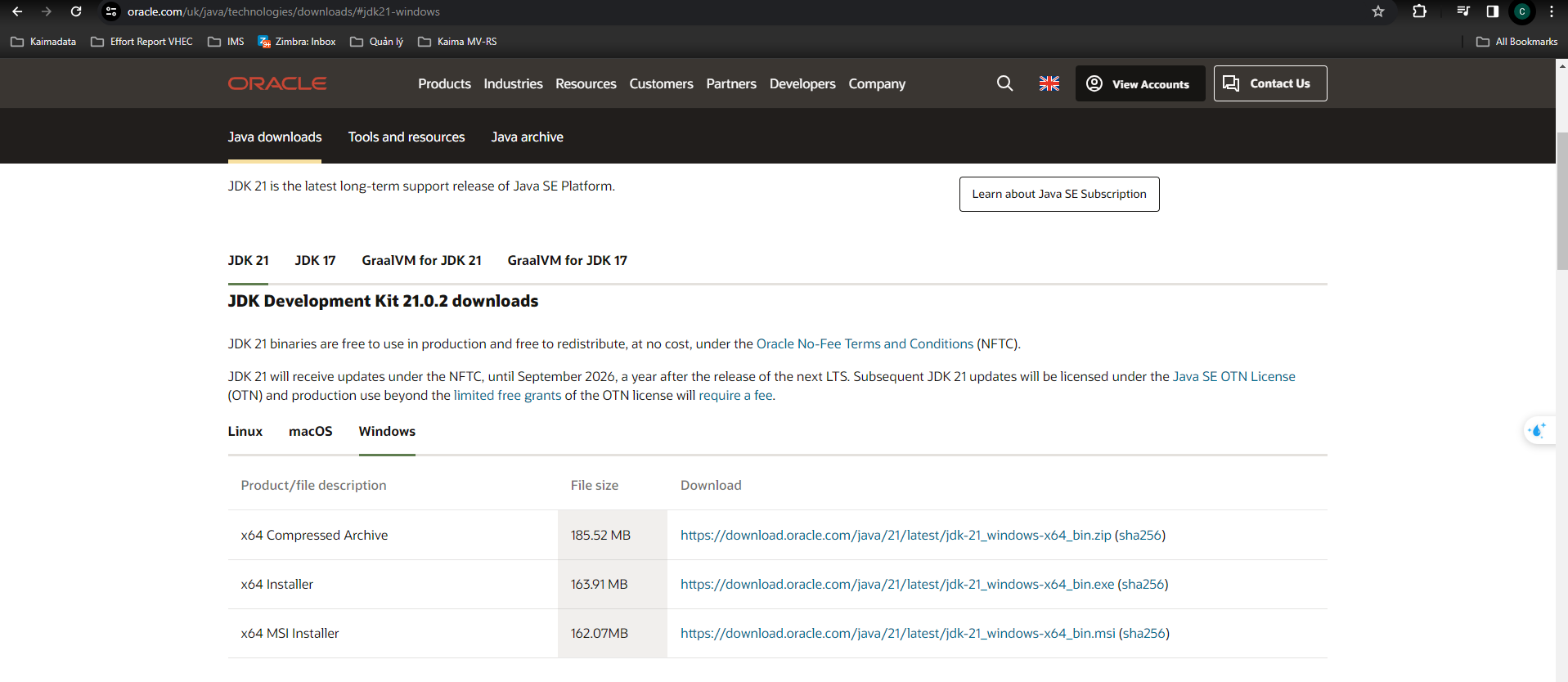
## Miscellaneous

* **AOP : cung cấp thực hiện lập trình hướng khía cạnh cho phép bạn xác định method-interceptors, pointcuts để tách mã thực hiện chức năng.**
* **Aspects :** cung cấp tích hợp với AspectJ - là một framework (AOP) lập trình hướng khía cạnh mạnh mẽ.
* **Instrumentation** : cung cấp lớp hỗ trợ instrumentation và lớp loader instrumentation để sử dụng trong các máy chủ ứng dụng nhất định. Một định nghĩa khác : hỗ trợ đo đạc và triển khai lớp bộ nạp.
* **Messaging** : cung cấp hỗ trợ cho [STOMP](#_STOMP_là_gì) như WebSocket sub-protocol để sử dụng trong các ứng dụng. Nó cũng hỗ trợ model chú thích cho việc định tuyến và xử lý tin nhắn STOMP từ WebSocket.

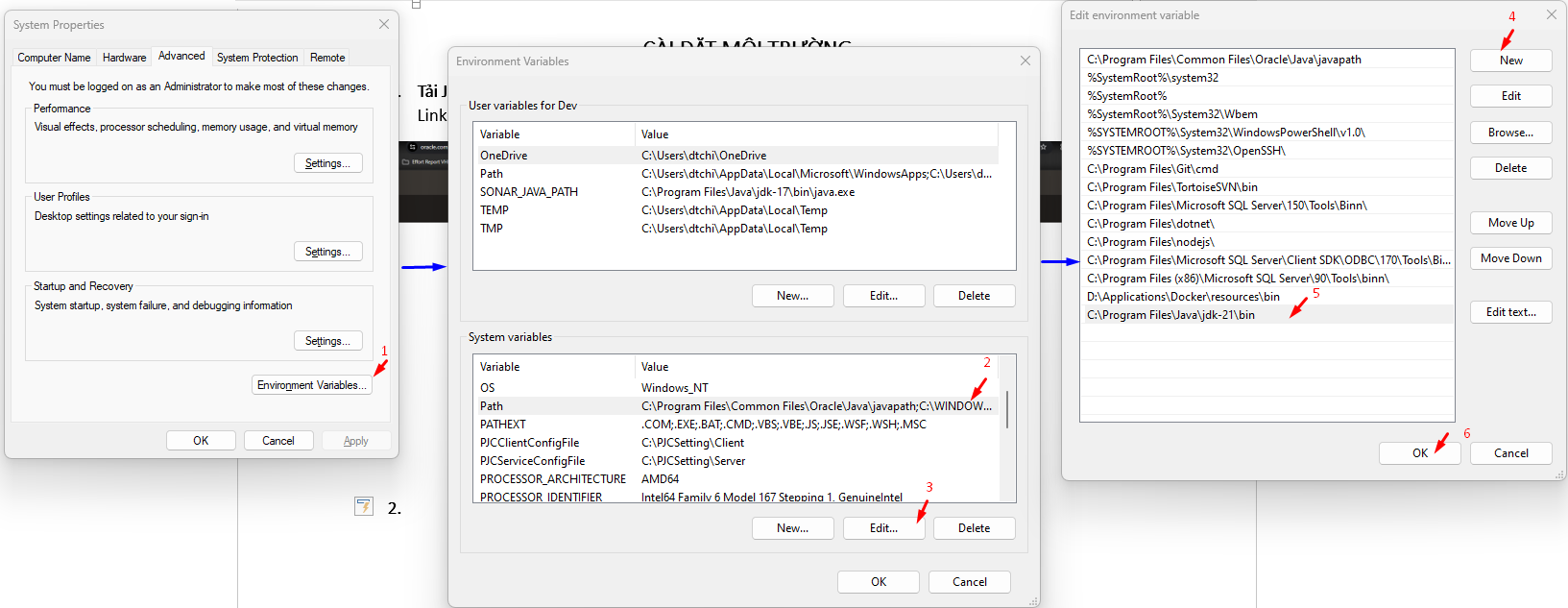
# CÀI ĐẶT MÔI TRƯỜNG

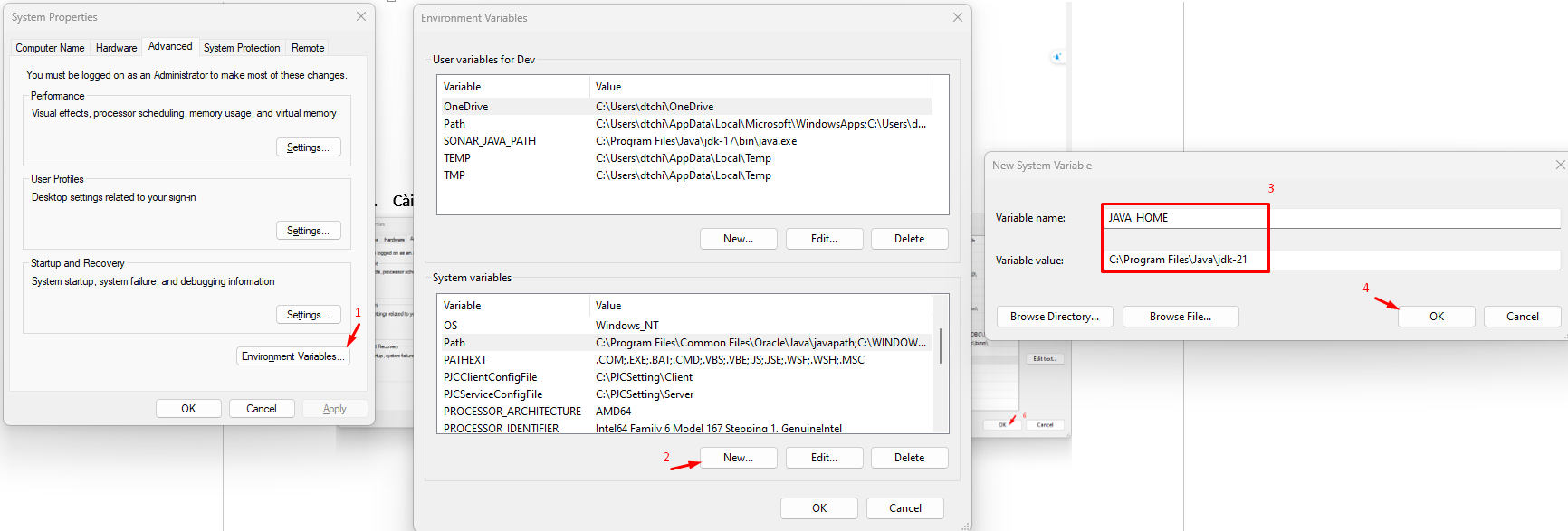
## Cài đặt JDK

* Link : <https://www.oracle.com/uk/java/technologies/downloads/>

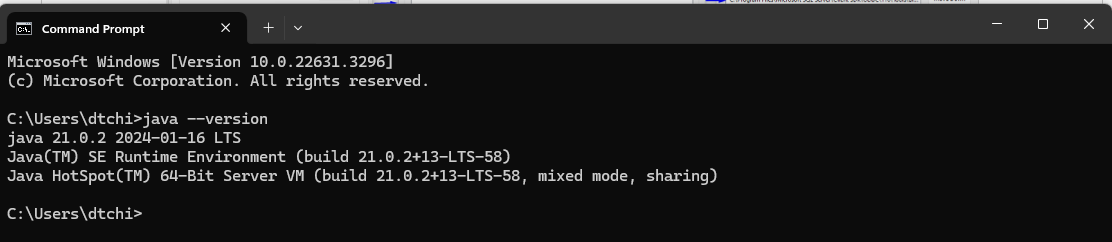


* Cài đặt biến môi trường



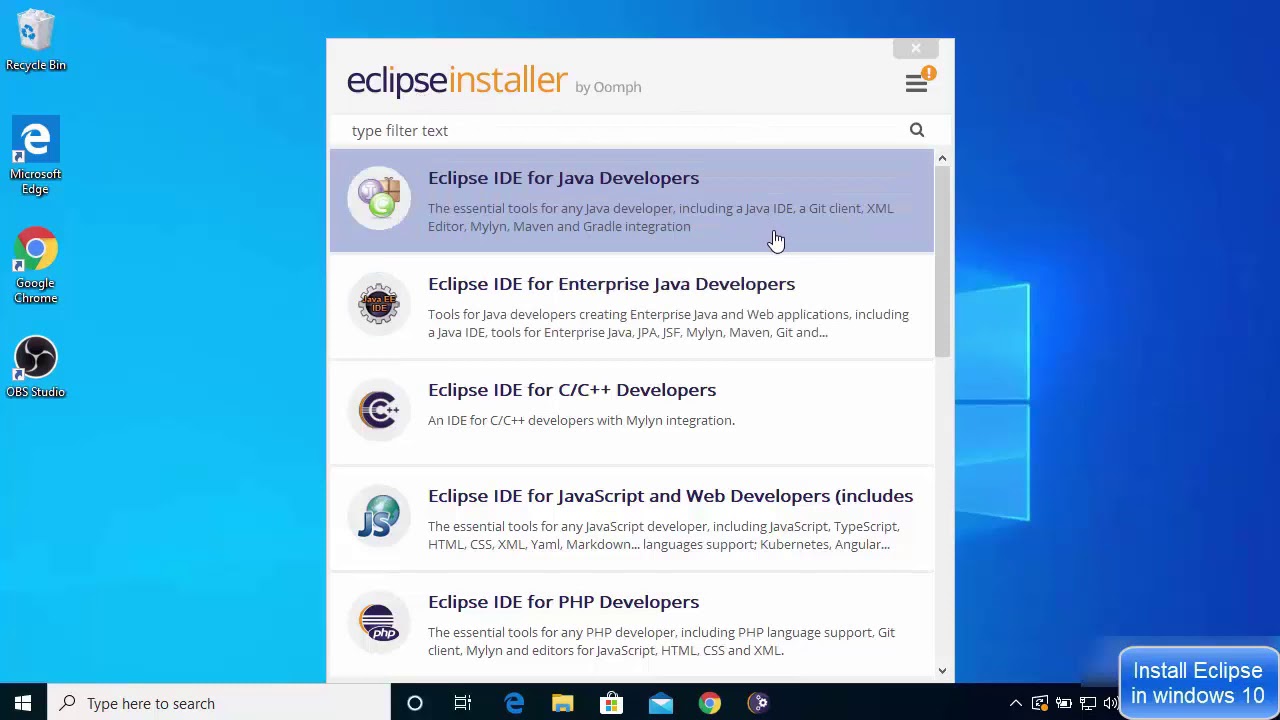


* Kiểm tra version



## Cài đặt Eclipse IDE

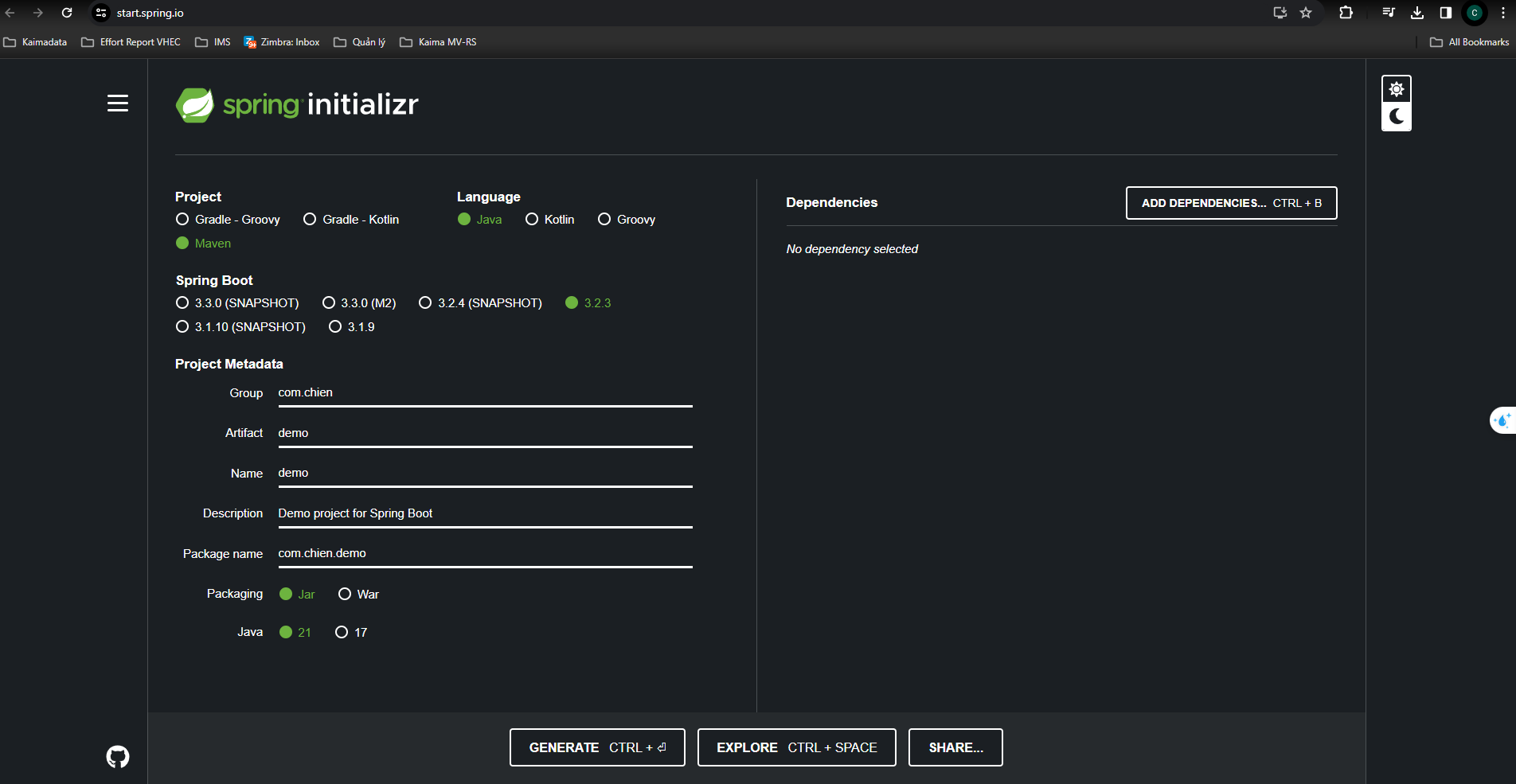
* Link : <https://www.eclipse.org/downloads/>
* Chọn Eclipse IDE for Java Developers



# TẠO PROJECT SPRING

## Tạo trên web

Link : <https://start.spring.io/>



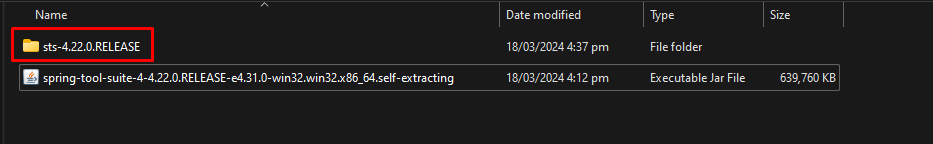
* **Project** :
  + Maven: Tạo dự án sử dụng Maven làm công cụ quản lý dự án.
  + Gradle: Tạo dự án sử dụng Gradle làm công cụ quản lý dự án.
  + Kotlin: Tạo dự án sử dụng Kotlin thay vì Java làm ngôn ngữ chính.
* **Language** :
  + Java: Sử dụng ngôn ngữ Java để phát triển ứng dụng.
  + Kotlin: Sử dụng ngôn ngữ Kotlin (mặc định hoặc tùy chọn thay thế cho Java).
* **Spring Boot** : Chọn phiên bản Spring Boot bạn muốn sử dụng
* **Project Metadata** :
  + Group: Nhóm (group) của dự án, thường được sử dụng như gói (package) cơ bản cho mã nguồn của bạn.
  + Artifact: Tên của dự án (artifact), thường được sử dụng để xác định dự án trong tệp jar hoặc war khi bạn xây dựng dự án.
  + Name: Tên của dự án, được hiển thị trong các công cụ quản lý dự án và trang web.
* **Dependencies** : Lựa chọn các dependencies (thư viện) muốn sử dụng trong dự án . Các dependencies này cung cấp các tính năng cụ thể và được tự động thêm vào dự án khi tạo nó.

Một số dependencies phổ biến:

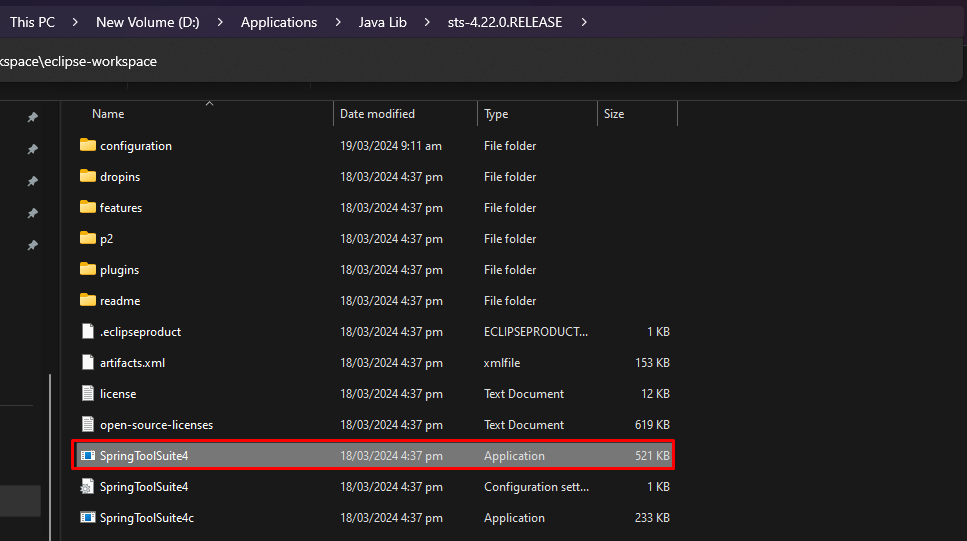
* + Spring Web: Hỗ trợ phát triển ứng dụng web với Spring MVC và RESTful Web Services.
  + Spring Data JPA: Hỗ trợ tương tác với cơ sở dữ liệu thông qua JPA (Java Persistence API).
  + Spring Security: Cung cấp các tính năng bảo mật cho ứng dụng Spring Boot.

## Tạo bằng tool

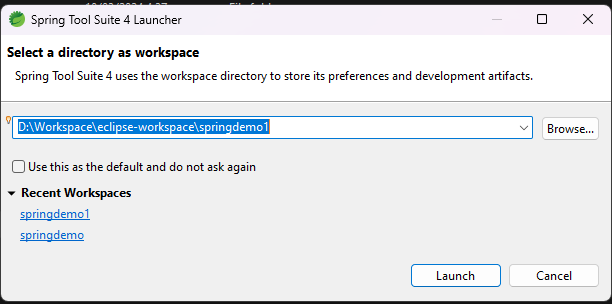
* Cài Spring tool Suite :
  + Link : <https://spring.io/tools>
  + Giải nén file jar sẽ được thư mục như ảnh bên dưới :



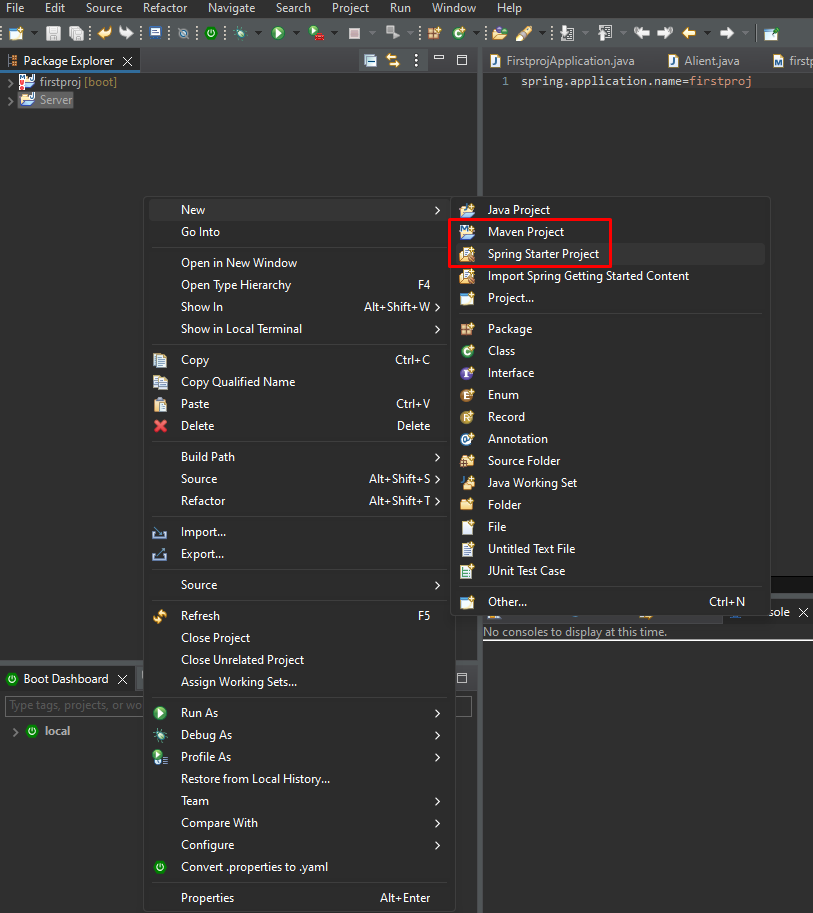
* + Chạy SpringToolSuite



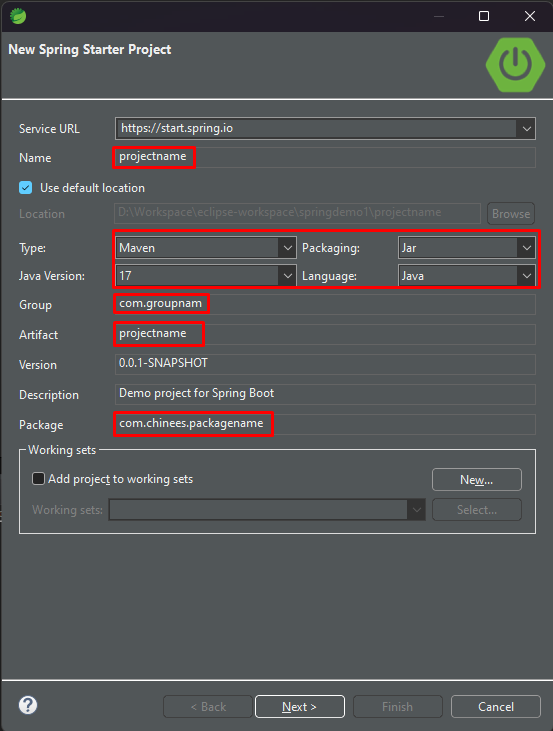
Chọn workspace.



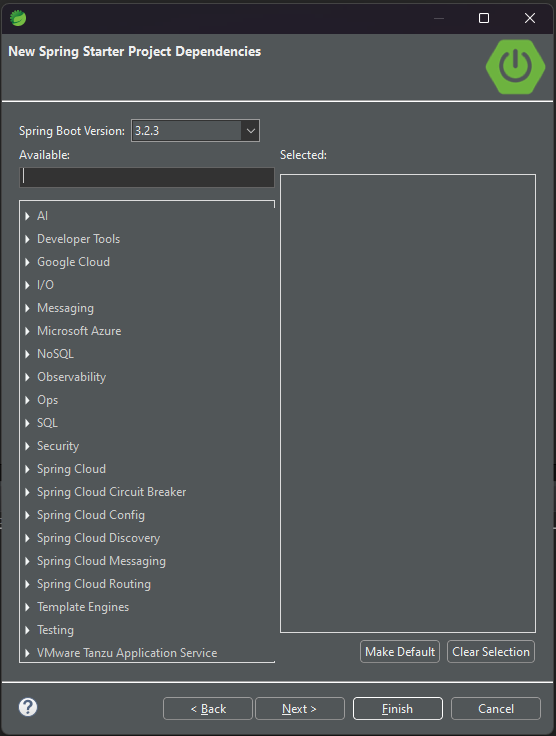
Tạo project mới - Spring Starter Project



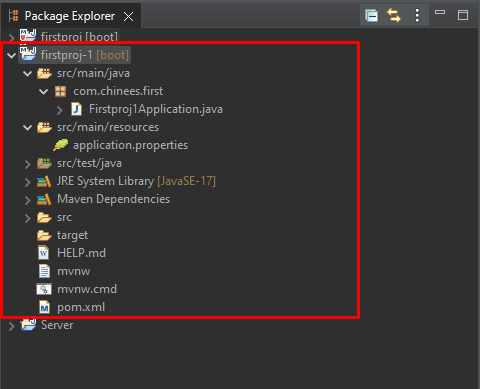
Điền thông tin và chọn các tùy chọn khởi tạo.



Chọn các dependencies



# CẤU TRÚC DỰ ÁN



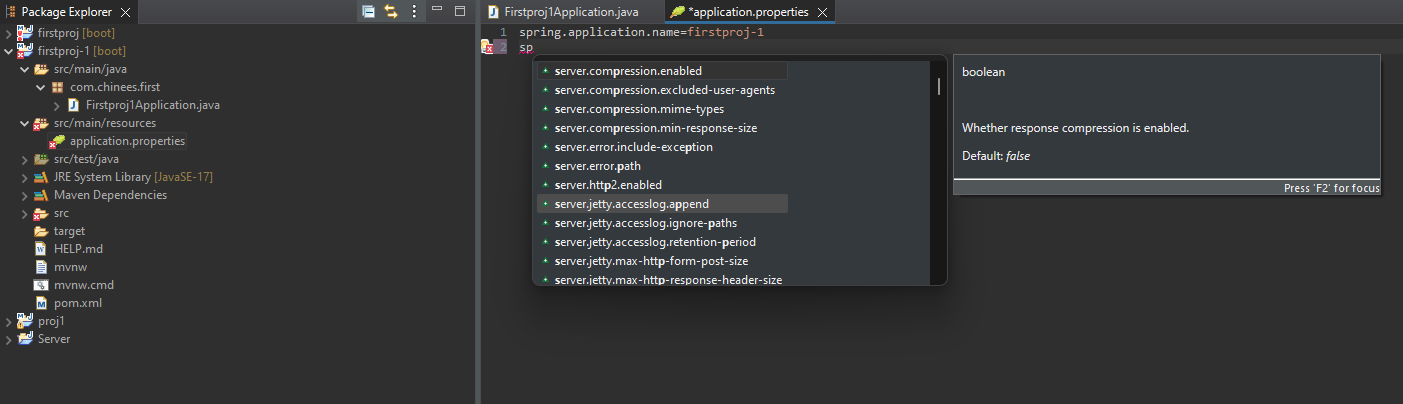
1. src/main/java : nơi chứa code xử lý chính của dự án

* Mã nguồn Java: Thư mục src/main/java chứa tất cả các file mã nguồn Java của ứng dụng, bao gồm các class, interfaces, enum và các tệp tin khác.
* Spring Beans và Components: Trong Spring Framework, bạn sẽ tạo các Spring Beans và các thành phần của ứng dụng bằng cách viết mã nguồn Java và đánh dấu chúng với các annotation như @Component, @Service, @Repository, @Controller, v.v. Các class được đánh dấu như vậy sẽ được Spring quản lý và có thể được tự động cấu hình và chèn vào các thành phần khác của ứng dụng.
* Controller và REST Endpoints: Trong Spring MVC hoặc Spring WebFlux, các class Controller được đặt trong thư mục src/main/java để xử lý yêu cầu từ phía client và trả về các phản hồi thích hợp. Đối với ứng dụng RESTful, các REST endpoints cũng được triển khai ở đây.
* Service Layer: Các class service, chứa các logic kinh doanh của ứng dụng, thường được đặt trong thư mục src/main/java. Các class này thường được sử dụng để thực hiện các thao tác nghiệp vụ, gọi các phương thức từ các repository và tương tác với các Spring Beans khác.
* Data Access Layer: Các class DAO (Data Access Object) hoặc repository, được sử dụng để truy cập và tương tác với cơ sở dữ liệu, thường cũng được đặt trong thư mục src/main/java.
* Configuration Classes: Trong một số trường hợp, bạn có thể cần tạo các class cấu hình (configuration classes) để cấu hình các bean, gắn kết các thành phần và xử lý cấu hình cho ứng dụng của mình. Các class này thường được đặt trong thư mục src/main/java.

1. src/main/resources : là nơi chứa các tài nguyên không phải mã nguồn (non-source resources) được sử dụng trong dự án. Các tài nguyên này có thể là tệp tin cấu hình, tệp tin văn bản, hình ảnh, tệp tin XML, và các tài nguyên khác mà ứng dụng của bạn cần sử dụng.

* Tệp tin cấu hình: Thư mục này thường chứa các tệp tin cấu hình cho ứng dụng Spring, như tệp tin application.properties hoặc application.yml để cấu hình các thuộc tính của ứng dụng.
* Tệp tin văn bản và tài nguyên tĩnh: Bạn có thể đặt các tệp tin văn bản, như các tệp tin CSV hoặc tệp tin văn bản để chứa dữ liệu tĩnh của ứng dụng. Ngoài ra, các tài nguyên tĩnh như hình ảnh, font chữ, CSS, JavaScript cũng có thể được đặt trong thư mục này.
* Tệp tin XML: Nếu ứng dụng của bạn sử dụng các tệp tin XML cho mục đích cụ thể, bạn có thể đặt chúng trong thư mục này. Ví dụ, các tệp tin XML cấu hình Spring Bean có thể được đặt ở đây.
* Các tài nguyên khác: Ngoài các loại tài nguyên đã đề cập, bạn có thể đặt các tài nguyên khác mà ứng dụng của bạn cần trong thư mục này, chẳng hạn như các tệp tin template cho các framework như Thymeleaf hoặc Freemarker.

1. src/test/java: Đây là thư mục chứa mã nguồn Java cho các bài kiểm tra (unit tests, integration tests) của ứng dụng. Các test cases và các class test được đặt trong thư mục này.
2. JRE system library : thường xuất hiện trong các IDE như Eclipse, IntelliJ IDEA, hoặc NetBeans khi bạn tạo hoặc mở một dự án Java. Nó đại diện cho tập hợp các thư viện và các lớp cơ bản của Java Runtime Environment (JRE) mà bạn đã cài đặt trên máy tính của mình.



1. Maven dependencies : Maven là một công cụ quản lý dự án và dependencies mạnh mẽ cho Java, cho phép quản lý và tự động hóa việc tải và cài đặt các thư viện và dependencies của dự án.
2. mvnn : Tập tin mvnw (hoặc mvnw.bat trên Windows) là một tập tin script dùng để chạy Maven mà Spring Boot cung cấp sẵn trong các dự án của mình. Tập tin này giúp đảm bảo rằng bạn sử dụng phiên bản Maven được quản lý bởi Spring Boot, đảm bảo tính nhất quán của môi trường phát triển.Khi bạn sử dụng mvnw, Maven sẽ được khởi chạy từ trong thư mục dự án của bạn mà không cần phải cài đặt Maven global trên hệ thống của bạn. Điều này giúp đảm bảo mỗi dự án sử dụng phiên bản Maven tương thích với Spring Boot, giảm thiểu sự phụ thuộc vào phiên bản Maven toàn cục.
3. mvnw.cmd : tương tự mvnw nhưng là tệp lênh cho Windows Command Prompt hoặc PowerShell.
4. pom.xml : chỉ định các dependencies cần thiết cho dự án của mình, bao gồm các thư viện Spring và các thư viện bên thứ ba khác.

# 10 VẠN CÂU HỎI VÌ SAO ???

## Sự khác nhau giữa spring và spring boot ?

|  |  |
| --- | --- |
| **Spring** | **Spring Boot** |
| Spring Framework là một khung Java EE được sử dụng rộng rãi để xây dựng các ứng dụng. | Spring Boot Framework được sử dụng rộng rãi để phát triển các REST APIs. |
| Nó nhằm mục đích đơn giản hóa việc phát triển Java EE giúp các nhà phát triển làm việc hiệu quả hơn. | Nó nhằm mục đích rút ngắn độ dài mã và cung cấp cách dễ dàng nhất để phát triển Web Application. |
| Tính năng chính của Spring Framework là dependency injection. | Tính năng chính của Spring Boot là Autoconfiguration. Nó tự động cấu hình các lớp dựa trên yêu cầu. |
| Nó giúp làm cho mọi thứ đơn giản hơn bằng cách cho phép chúng ta phát triển loosely coupled applications. | Nó giúp tạo một ứng dụng độc lập với ít cấu hình hơn. |
| Lập trình viên cần viết rất nhiều mã (mã viết sẵn) để thực hiện nhiệm vụ tối thiểu. | Nó làm giảm đi mã viết sẵn. |
| Để kiểm tra dự án Spring, chúng ta cần thiết lập sever một cách rõ ràng. | Spring Boot cung cấp máy chủ nhúng như Jetty và Tomcat, v.v. |
| Các nhà phát triển xác định thủ công các dependencies cho dự án Spring trong pom.xml. | Spring Boot đi kèm với khái niệm khởi động trong tệp pom.xml, bên trong xử lý việc tải xuống các JAR phụ thuộc dựa trên Spring Boot Requirement. |

## Sự khác nhau giữa spring và spring mvc ?

|  |  |
| --- | --- |
| **Spring** | **Spring MVC** |
| Spring Boot là một module của Spring để đóng gói ứng dụng dựa trên Spring với các giá trị mặc định hợp lý. | Spring MVC model view controller-based web framework dựa trên Spring. |
| Nó cung cấp các cấu hình mặc định để xây dựng Spring-powered framework. | Nó cung cấp các tính năng sẵn sàng sử dụng để xây dựng một ứng dụng web. |
| Không có yêu cầu đối với các file mô tả khi triển khai. | Cần có nhiều file mô tả triển khai. |
| Nó tránh mã viết sẵn và kết hợp các dependencies lại với nhau trong một đơn vị duy nhất. | Nó chỉ định từng dependency riêng biệt. |
| Nó làm giảm thời gian phát triển và tăng năng suất. | Cần nhiều thời gian hơn để đạt được điều tương tự. |

## IoC là gì ?

Inversion of Control là Đảo ngược điều khiển, nó giúp làm thay đổi luồng điều khiển của chương trình một cách linh hoạt. [Xem thêm](https://viblo.asia/p/spring-ioc-inversion-of-control-trong-spring-4P856gJaKY3)

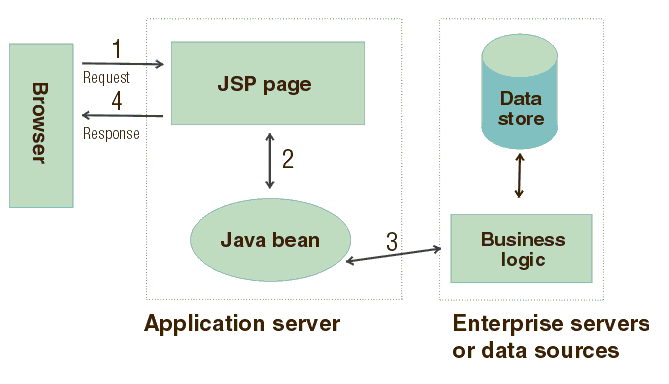
## Expression Language là gì ?

JSP Epression Language (EL) giúp dễ dàng truy cập dữ liệu ứng dụng được lưu giữ trong các thành phần JavaBeans. JSP EL cho phép bạn tạo các Expression, gồm số học và logic. Bên trong một JSP EL, bạn có thể sử dụng các integer, các số floating point, string, các hằng có sẵn true hoặc false cho các giá trị Boolean, và null. [Xem thêm](https://hoclaptrinh.vn/tutorial/hoc-jsp/expression-language-el-trong-jsp)

## JavaServer Pages là gì ?

Nó là 1 công nghệ dùng để tạo web động và là 1 phần của Java EE. JSP cho phép dev nhúng code Java trực tiếp vào trang HTML. [Xem thêm](https://viettuts.vn/jsp)

Vị trí của JSP trong một ứng dụng web:



## Hibernate là gì ?

Là một trong những ORM framework phổ biến nhất trong việc thao tác với cơ sở dữ liệu trong ứng dụng Java. Tuy nhiên, một số nhà phát triển cảm thấy Hibernate có quá nhiều chức năng, mà không phải tất cả đều cần thiết cho ứng dụng của họ. Do đó, JPA ra đời để giải quyết vấn đề đó

## Java Persistence API là gì ?

Là một đặc tả Java cho việc ánh xạ giữa các đối tượng Java với cơ sở dữ liệu quan hệ sử dụng công nghệ phổ biến là ORM (Object Relational Mapping). JPA cung cấp đầy đủ các công cụ cho phép chúng ta có thể thao tác với cơ sở dữ liệu một cách đơn giản và nhanh chóng. [Xem thêm](https://topdev.vn/blog/tong-quan-ve-jpa-java-persistence-api/#jpa-la-gi)

## Java Data Objects là gì ?

Là một API được thiết kế để duy trì dữ liệu hướng đối tượng vào bất kỳ cơ sở dữ liệu nào và cung cấp ngôn ngữ truy vấn thân thiện với người dùng bằng cú pháp Java. [Xem thêm](https://www.baeldung.com/jdo)

## Portlet là gì ?

Portlet MVC là một khung công tác web MVC theo yêu cầu, được thiết kế xung quanh một portlet gửi các yêu cầu đến bộ điều khiển và cung cấp các chức năng khác tạo điều kiện thuận lợi cho việc phát triển các ứng dụng portlet. Tuy nhiên, DispatcherPortlet của Spring còn làm được nhiều việc hơn thế. Nó được tích hợp hoàn toàn với Spring ApplicationContext và cho phép bạn sử dụng mọi tính năng khác mà Spring có.

## STOMP là gì ?

STOMP (Giao thức nhắn tin hướng văn bản đơn giản) ban đầu được tạo cho các ngôn ngữ kịch bản (như Ruby, Python và Perl) để kết nối với các nhà môi giới tin nhắn doanh nghiệp. Nó được thiết kế để giải quyết một tập hợp con tối thiểu các mẫu tin nhắn thường được sử dụng. STOMP có thể được sử dụng trên mọi giao thức mạng truyền phát hai chiều đáng tin cậy, chẳng hạn như TCP và WebSocket. Mặc dù STOMP là giao thức hướng văn bản nhưng tải trọng tin nhắn có thể là văn bản hoặc nhị phân. [Xem thêm](https://docs.spring.io/spring-framework/reference/web/websocket/stomp/overview.html)