**4 đặc tính cơ bản của OOP**

* Tính trừu tượng (Abstraction)

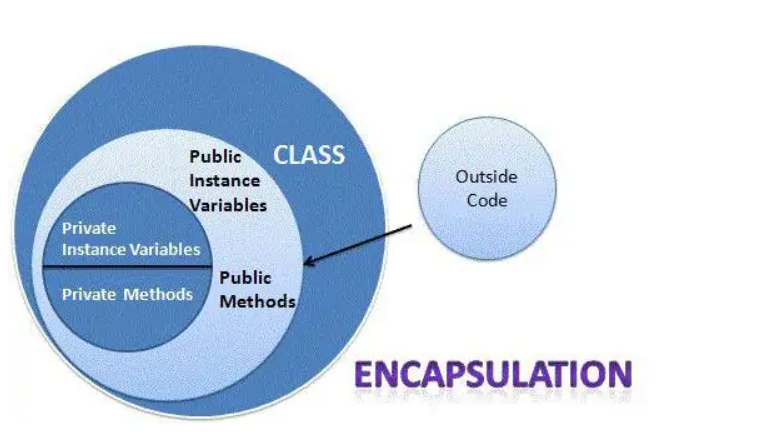
+ Tính trừu tượng giúp loại bỏ những thứ phức tạp, không cần thiết của đối tượng và chỉ tập trung vào những gì cốt lõi, quan trọng.

* Tính kế thừa (Inheritance)

+ Tính kế thừa cho phép xây dựng một lớp mới (lớp Con), kế thừa và tái sử dụng các thuộc tính, phương thức dựa trên lớp cũ (lớp Cha) đã có trước đó.

+ Các lớp Con kế thừa toàn bộ thành phần của lớp Cha và không cần phải định nghĩa lại. Lớp Con có thể mở rộng các thành phần kế thừa hoặc bổ sung những thành phần mới.

* Tính đóng gói (Encapsulation)



+ Tính đóng gói cho phép che giấu thông tin và những tính chất xử lý bên trong của đối tượng. Các đối tượng khác không thể tác động trực tiếp đến dữ liệu bên trong và làm thay đổi trạng thái của đối tượng mà bắt buộc phải thông qua các phương thức công khai do đối tượng đó cung cấp.

+ Tính chất này giúp tăng tính bảo mật cho đối tượng và tránh tình trạng dữ liệu bị hư hỏng ngoài ý muốn.

* Tính đa hình (Polymorphism)

+ Tính đa hình trong lập trình OOP cho phép các đối tượng khác nhau thực thi chức năng giống nhau theo những cách khác nhau.

JAVA CORE LEARN:

1. JVM la gi

2. Quan ly bo nho trong JVM

3. Cac loai bien

4. Polymorphism Runtime + CompileTime

5. Excaption handling

6. JDBC

1. JVM là gì?

* JVM (Java Virtual Machine) là 1 máy ảo java – nó được dùng để thực thi các chương trình Java hay hiểu nôm na là trình thông dịch của Java. Nó cung cấp môi trường để code java có thể được thực thi

1. Quản lý bộ nhớ trong JVM: Khi chạy chương trình Java, JVM sẽ yêu cầu hệ điều hành cấp cho một không gian bộ nhớ trong RAM để dùng cho việc chạy chương trình. JVM sẽ chia bộ nhớ được cấp phát này thành 2 phần: Heap và Stack cho việc quản lý.

* Vùng nhớ Heap:
  + Là bộ nhớ được sử dụng ở runtime để lưu các Objects. Bất cứ khi nào, ở đâu trong chương trình của bạn khi bạn tạo Object thì nó sẽ được lưu trong Heap (thực thi toán tử new).
  + Các objects trong Heap đều được truy cập bởi tất cả các các nơi trong ứng dụng, bởi các threads khác nhau.
  + Thời gian sống của object phụ thuộc vào Garbage Collection của java.
  + Garbage Collection sẽ chạy trên bộ nhớ Heap để xoá các Object không được sử dụng nữa, nghĩa là object không được referece trong chương trình.
  + Dung lượng sử dụng của Heap sẽ tăng giảm phụ thuộc vào Objects sử dụng.
  + Dung lượng Heap thường lớn hơn Stack.
* Vùng nhớ Stack:
  + Bộ nhớ để lưu các biến local trong hàm và lời gọi hàm ở runtime trong một Thread java.
  + Các biến local bao gồm loại nguyên thuỷ (primitive) và loại tham chiếu tới đối tượng trong heap (reference) khai báo trong hàm, hoặc đối số được truyền vào hàm, thường có thời gian sống ngắn.
  + Bố nhớ stack thường nhỏ.
  + Cơ chế hoạt động là LIFO (Last-In-First-Out), chạy sau chết trước.
  + Bất cứ khi nào gọi 1 hàm, một khối bộ nhớ mới sẽ được tạo trong Stack cho hàm đó để lưu các biến local. Khi hàm thực hiện xong, khối bộ nhớ cho hàm sẽ bị xoá, và giải phóng bộ nhớ trong stack.

\* Ngắn gọn:

* Bộ nhớ Heap và bộ nhớ Stack bản chất đều cùng là vùng nhớ được tạo ra và lưu trữ trong RAM khi chương trình được thực thi.
* Bộ nhớ Stack được dùng để lưu trữ các biến cục bộ trong hàm, tham số truyền vào

3. Các kiểu biến

Có 3 kiểu biến trong java, bao gồm biến local, biến instance, biến static.

* Biến local
  + Biến local được khai báo trong các phương thức, hàm contructor hoặc trong các block.
  + Biến local được tạo bên trong các phương thức, contructor, block và sẽ bị phá hủy khi kết thúc các phương thức, contructor và block.
  + Không được sử dụng "access modifier" khi khai báo biến local.
  + Các biến local được lưu trên vùng nhớ stack của bộ nhớ.
  + Cần khởi tạo giá trị mặc định cho biến local trước khi sử dụng.
* Biến instance
  + Biến instance được khai báo trong một lớp(class), bên ngoài các phương thức, constructor và các block.
  + Biến instance được lưu trong bộ nhớ heap.
  + Biến instance được tạo khi một đối tượng được tạo bằng việc sử dụng từ khóa “new” và sẽ bị phá hủy khi đối tượng bị phá hủy.
  + Biến instance có thể được sử dụng bởi các phương thức, constructor, block, ... Nhưng nó phải được sử dụng thông qua một đối tượng cụ thể.
  + Được phép sử dụng "access modifier" khi khai báo biến instance, mặc định là "default".
  + Không cần khởi tạo giá trị cho biến instance trước khi sử dụng.
  + Bên trong class mà bạn khai báo biến instance, có thể gọi nó trực tiếp bằng tên khi sử dụng ở khắp nới bên trong class đó.
* Biến static
  + Biến static được khai báo trong một class với từ khóa "static", phía bên ngoài các phương thức, constructor và block.
  + Sẽ chỉ có duy nhất một bản sao của các biến static được tạo ra, dù tạo bao nhiêu đối tượng từ lớp tương ứng.
  + Biến static được lưu trữ trong bộ nhớ static riêng.
  + Biến static được tạo khi chương trình bắt đầu chạy và chỉ bị phá hủy khi chương trình dừng.
  + Giá trị mặc định của biến static phụ thuộc vào kiểu dữ liệu bạn khai báo tương tự biến instance.
  + Biến static được truy cập thông qua tên của class chứa nó, với cú pháp: TenClass.tenBien.
  + Trong class, các phương thức sử dụng biến static bằng cách gọi tên của nó khi phương thức đó cũng được khai báo với từ khóa "static".

4. Runtime + Compile-time Polymorphism

- Compile-time Polymorphism: Overloading (Nạp chồng phương thức)

- Runtime Polymorphism: Overriding (Ghi đè phương thức)

5. Exception handling

- Các ngoại lệ Java:

1. Checked Exception

* Các class kế thừa từ class Throwable (ngoại trừ RuntimeException và Error) được gọi là Checked Exception. Checked Exception được kiểm tra tại Compile-time. Ví dụ: IOException, SQLException, v.v.

1. Unchecked Exception

* Các lớp kế thừa RuntimeException được gọi là Unchecked Excaption. Unchecked Exception không được kiểm tra tại Compile-time nhưng chúng được kiểm tra tại Runtime. Ví dụ: ArithmeticException, NullPulumException, ArrayIndexOutOfBoundsException, v.v.

1. Error

* Một số ví dụ về Error là OutOfMemoryError, VirtualMachineError, AssertionError, v.v.

Java Excaption Keyword

* Try: Đặt trong **try** một khối lệnh, nơi ta nên đặt code ngoại lệ. Không thể sử dụng khối **try** một mình, bắt buộc phải theo sau bởi **catch** hoặc **finally**
* Catch: Khối **catch** dùng để xử lý ngoại lệ. Không thể sử dụng khối **catch** một mình, khối **catch** bắt buộc phải đặt sau khối **try**. Nó cũng có thể theo sau bởi khối **finally**
* Finally: Được sử dụng để thực thi code cần thiết của chương trình. Nó được thực thi bất kể ngoại lệ có được xử lý hay không
* Throw: Được sử dụng để đưa ra một ngoại lệ
* Throws: Được sử dụng để khai báo một ngoại lệ

**try => catch (=> finally)**

6. JDBC

- Java JDBC là một java API được sử dụng để kết nối và thực hiện truy vấn với cơ sở dữ liệu. JDBC API sử dụng trình điều khiển jdbc để kết nối với cơ sở dữ liệu.

Một **Connection** trong java là phiên làm việc giữa ứng dụng java và cơ sở dữ liệu. Đối tượng Connection được sử dụng để tạo Statement, PreparedStatement, và DatabaseMetaData. Giao diện Connection cung cấp nhiều phương thức quản lý transaction như commit(), rollback(), ...

Đối tượng của **ResultSet** duy trì một con trỏ trỏ đến một hàng của một bảng. Ban đầu, con trỏ trỏ đến hàng đầu tiên.

**Statement interface** trong java cung cấp các phương thức để thực thi các câu lệnh truy vấn với cơ sở dữ liệu SQL. Statement interface là một nhà máy của ResultSet, tức là nó cung cấp phương thức để tạo ra đối tượng ResultSet.

Giao diện **PreparedStatement** trong java là một sub-interface của Statement. Nó được sử dụng để thực hiện truy vấn tham số.

Giao diện **ResultSetMetaData** trong java được sử dụng để lấy ra các metadata từ đối tượng ResultSet.

Giao diện **DatabaseMetaData** trong java cung cấp các phương thức để lấy metadata của cơ sở dữ liệu như tên sản phẩm cơ sở dữ liệu, phiên bản sản phẩm cơ sở dữ liệu, tên driver, tên của tổng số bảng, tên của tổng số các view, ...

**SQL Fundamental:**

1. DDL(Data Definition Language-Ngôn ngữ định nghĩa dữ liệu):
   * Liên quan đến các lược đồ và mô tả CSDL,về cách dữ liệu sẽ nằm trong CSDL.
     + CREATE
     + ALTER
     + DROP
     + TRUNCATE:loại bỏ tất cả các bản ghi khỏi bảng,bao gồm tất cả các khoảng trắng được phân bổ cho các bản ghi sẽ bị xóa.
     + COMMENT
     + RENAME
2. DML(Data Manipulation Language-Ngôn ngữ thao tác dữ liệu):
   * Liên quan đến các thao tác dữ liệu
     + SELECT
     + INSERT
     + UPDATE
     + DELETE
     + MERGE-UPSERT
     + CALL:gọi chương trình con PL/SQL hoặc java
     + EXPLAIN PLAN:Giải thích đường dẫn truy cập dữ liệu.
     + LOCK TABLE:kiểm soát đồng thời

**HTML,CSS,JS,JQUERY**

1.HTML

-HTML viết tắt bởi cụm từ HyperText Markup Language (ngôn ngữ đánh dấu

siêu văn bản) sử dụng các thẻ tag để trình bày các văn bản, đồ họa, đa phương tiện trên trình duyệt web.

-HEAD: Thẻ Head được dùng để xác định phần mở đầu cho tài liệu và các thành phần trên trình duyệt như thanh tiêu đề, thanh trạng thái hoặc là bộ phông chữ hỗ trợ trong trang web.

-BODY: Thẻ này được sử dụng để xác định phần nội dung chính của tài liệu.

- FORM:

+ACTION: Xác định địa chỉ của script sẽ xử lý biểu mẫu hoàn chỉnh và

được gửi đi.

+ACCEPT: Xác định danh sách các kiểu MIME được máy chủ nhận ra,trong đó có chứa script để xử lý biểu mẫu.

2.CSS

-Là một ngôn ngữ quy định cách trình bày của các thẻ HTML trên trang web.

- CSS có thể quản lý toàn bộ các thành phần (giao diện) của trang web

thông qua 1 file CSS, điều khiển chính xác hơn cách trình bày Layout.

-Quan trọng nhất là tính tái sử dụng để áp dụng cho nhiều thành phần giống nhau. Sử dụng được các kỹ thuật tiên tiến, phức tạp mà HTML không làm được.

- Cú pháp CSS cơ bản:

Tên\_thẻ{Thuộc\_tính:giá\_trị;}

Hoặc

.Tên\_class{Thuộc\_tính:giá\_trị;}

Hoặc

#Tên\_ID{Thuộc\_tính:giá\_trị;}

----1 Số thuộc tính cần chú ý:

POSITION

+Thuộc tính position trong CSS dùng để xác định vị trí hiển thị cho thẻ HTML và thường được dùng để xây dựng CSS cho menu đa cấp, tooltip hoặc một số chức năng khác. Position có tổng cộng 5 giá trị :

+static Dạng mặc định - sẽ hiển thị theo đúng thứ tự của nó (thường dùng để hủy

các thuộc tính bên dưới).

+relative :Định vị trí tương đối theo thẻ cha (thẻ khai báo relative) hoặc thẻ body nếuko có khai báo.

+absolute :Định vị trí tuyệt đối (vị trí bao ngoài), lúc này các thẻ HTML bên trong sẽ coi nó là thẻ cha.

+Fixed: Định vị trí tương đối cho của sổ Browser của trình duyệt (khi kéo scroll nó sẽ không bị ẩn đi).

+Inherit: Thừa hưởng các thuộc tính từ thành phần cha (thành phần bao ngoài nó).

**3.JS**

**-**JavaScript là ngôn ngữ kịch bản phía máy khách được nhà phát triển sử dụng để tạo trang web tương tác

**JAVA WEB**

== TOMCAT9,DEPLOY lên TOMCAT,thế nào là serverlet,staeful,staeteless,conect db dùng gì(JDBC)

-Set default :welcome first web.xml(trong file web.xml)

1.Serverlet

-Java Servlet là chương trình chạy trên một Web hoặc ứng dụng máy chủ (Application Server). Nó hoạt động như một lớp trung gian giữa một yêu cầu đến từ một trình duyệt Web hoặc HTTP khách (Client) khác và cơ sở dữ liệu hoặc các ứng dụng trên máy chủ HTTP (HTTP Server).

Hiểu đơn giản, Servlet là một chương trình chạy trên môi trường Web\_Server hoặc môi trường Application có thực thi mã java với nhiệm vụ chính là giúp thực thi câu lệnh một cách độc lập giúp kết nối các lớp với nhau.

-Công dụng:+ nhận client request và lấy thông tin từ request: Đọc dữ liệu rõ ràng do khách hàng (trình duyệt) gửi

+Tạo và gửi request đến client hoặc tại request mới đến Servlet mới hoặc JSP mới

2.Stateless

-**Stateless** còn gọi là **tình trạng phi trạng thái**. Cụ thể, stateless là **thiết kế không lưu dữ liệu của client trên server**. Điều đó có nghĩa là sau khi client gửi dữ liệu lên server, khi server thực thi xong, trả kết quả thì quan hệ giữa client và server sẽ bị cắt đứt. Server sẽ không lưu bất kỳ dữ liệu gì của client.

3.stateful

Cụ thể ở **stateful**, **server cần lưu dữ liệu của client**. Điều đó có nghĩa là **có sự ràng buộc rõ ràng giữa client và server**. Dữ liệu được lưu giữ sau mỗi lần request (yêu cầu) của client. Dữ liệu sẽ được lưu lại phía server, có thể làm đầu vào (input parameters) cho những lần tiếp theo. Hoặc dữ liệu sẽ được dùng trong quá trình xử lý, phục vụ cho bất kỳ nhu cầu nào có phụ thuộc vào nghiệp vụ cài đặt.

Như vậy, nếu 1 hoạt động có stateful bị gián đoạn thì chúng được lưu trữ để bạn có thể tiếp tục thực hiện tại chính vị trí mà bạn đã dừng lại. Khác với stateless, với stateful, chúng ta cần sử dụng 1 máy chủ để có thể xử lý tất cả các yêu cầu được liên kết cùng với 1 thông tin trạng thái hoặc thông tin trạng thái cần phải chia sẻ với các máy chủ cần nó.

**Clean code and refactoring**

**1.Cấu trúc project**

* src/main/java: Đối với tệp nguồn
* src/main/resources: Đối với các tệp tài nguyên
* src/test/java: Đối với các tệp nguồn thử nghiệm
* src/test/resources: Đối với các tệp tài nguyên thử nghiệm

**2.Cấu trúc Source file**

- package

- Import statements

+All static imports

+All non-static imports

- Class:

+Class

+Instance variable

+Constructor

+Mehtod

**3.Quy tắc đặt tên**

- Class: Sử dụng danh từ để đặt tên.

-Biến:+Tên của biến phải mô tả rõ ý định của biến(private String customerName)

+Biến static,enum,final toàn cục:viết hoa,các từ cách nhau dấu gạch dưới

-Phương thức:Sử dụng động từ(public String getCustomerName)

**4.Khoảng trắng**

-Những dòng code có quan hệ với nhau (cùng thực hiện một công việc) thì gom lại thành một block.

- không có dòng trống giữa các đoạn code như trên.

-Hai block code thì cách nhau ít nhất một dòng trống.

-Đặt khoảng trắng sau dấu phẩy và dấu chấm phẩy.

-Đặt khoảng trắng xung quanh các toán tử.

-Hai dòng code cách nhau 1 bậc thì cách nhau 1 tab.

**Rrafactoring(**[Code Refactoring (viblo.asia)](https://viblo.asia/p/code-refactoring-PaLkDYldvlX)**)**

-Thay đổi cấu trúc của code mà không thay đổi chức năng của nó

-Rrafactoring code để đạt được sự đơn giản và ngắn gọn trong code,cải thiện cấu trúc code.

-khi code yêu cầu refactoring thì nó được coi là “code smell”.

#### - Large class: quá nhiều chức năng được nhồi nhét vào một lớp như vậy. Trong trường hợp này, nó giúp di chuyển một phần chức năng vào một lớp riêng biệt.

#### - Long method: có 2 phương pháp để refactoring một long method

#### + Nếu muốn thêm nhận xét khi viết một method, bạn nên đặt chức năng vào một method riêng biệt

+ Nếu một phương thức mất hơn 10-15 dòng mã,nên xác định các nhiệm vụ và nhiệm vụ con mà nó thực hiện và cố gắng đặt các nhiệm vụ con vào một phương thức riêng biệt.

#### Unit Testing with junit

#### Phân biệt giữa unit testing và integration testing

#### (<https://www.geeksforgeeks.org/difference-between-unit-testing-and-integration-testing/>)

#### unit testing

#### Unit Testing là một loại kiểm thử phần mềm trong đó các thành phần phần mềm riêng lẻ được kiểm tra.

#### Unit testing được thực hiện trong quá trình phát triển ứng dụng.

#### Một thành phần riêng lẻ có thể là một chức năng riêng lẻ hoặc một thủ tục.

#### Unit Testing thường được thực hiện bởi nhà phát triển.

#### Đây là một phương pháp kiểm tra sử dụng mọi mô-đun độc lập được kiểm tra để xác định xem có bất kỳ vấn đề nào do chính nhà phát triển kiểm tra hay không.

#### Integration testing

#### Integration testing là quá trình kiểm tra giao diện giữa hai đơn vị phần mềm hoặc mô-đun.

#### Trọng tâm của nó là xác định tính chính xác của giao diện.

#### Kiểm thử tích hợp nhằm mục đích phơi bày các lỗi trong tương tác giữa các đơn vị tích hợp.

#### Khi tất cả các mô-đun đã được kiểm tra đơn vị, kiểm tra tích hợp được thực hiện.

#### Tìm hiểu về Junit

#### ([A Guide to JUnit 5 | Baeldung](https://www.baeldung.com/junit-5))

#### ([Giới thiệu JUnit 5 - Java testing framework - Deft Blog (shareprogramming.net)](https://shareprogramming.net/gioi-thieu-junit-5-java-testing-framework/#JUnit_Platform))

#### 1.Junit là gì: JUnit là một framework đơn giản dùng cho việc tạo các unit testing tự động, và chạy các test có thể lặp đi lặp lại

#### 2.Junit 5 là gì: JUnit 5 là phiên bản mới của JUnit với mục đích hỗ trợ các tính năng mới ra mắt từ Java 8 trở về sau.

**3.** JUnit 5 gồm có 3 module khác nhau từ 3 sub-project khác nhau:

* JUnit platform
* JUnit Jupiter
* JUnit Vintage

#### Junit platform

**-** Unit platform chịu trách nhiệm khởi chạy JUnit framework trên [JVM](https://shareprogramming.net/jvm-va-su-khac-biet-giua-jvm-jdk-jre/). Nó định nghĩa các interface mạnh mẽ và ổn định để tương thích tốt với các môi trường khác nhau như chạy trên các IDE khác nhau (Intellij, Eclipse, NetBean etc).

- Ngoài ra, JUnit Platform cũng định nghĩa TestEngine API để phát triển các testing framework chạy nên JUnit platform. Chúng ta có thể sử dụng các thư viện tesing thứ 3 trực tiếp trên JUnit bằng cách implement TestEngine.

3.2 Junit Jupiter

- JUnit Jupiter bao gồm các mô hình lập trình và mở rộng mới phục vụ để viết unit test trong JUnit 5. Nó bao gồm các annotation từ JUnit 4 và các annotation mới được thêm trong JUnit 5:

* **@Test** – Đặt ở đầu method để thông báo rằng method được dùng để kiểm thử(test method).
* **@ParameterizedTest –** Test method chạy nhiều lần với các tham số khác nhau.
* **@RepeatedTest** -Test method chạy kiểm thử lặp lại n lần.
* **@TestFactory** – Chỉ định method là một test factory cho các dynamic test.
* **@TestTemplate** – Chỉ định method là một mẫu cho các test case.
* **@TestMethodOrder** – Cấu hình thứ tự thực thi cho các @Test.
* **@TestInstance** – Quy định vòng đời của cho các annotation test class (@Nested, biến static etc)
* **@DisplayName** – Đặt tên cho test class hoặc test method.
* **@DisplayNameGeneration** – Đặt tên cho các test class được generate.
* **@BeforeEach** – chỉ định 1 method luôn được thực thi trước mỗi test method thực thi.
* **@AfterEach** – chỉ định 1 method luôn thực thi sau khi 1 test method thực thi xong.
* **@BeforeAll** – Chỉ định method sẽ được thực thi đầu tiên trong test class.
* **@AfterAll** – Chỉ định method sẽ được thực thi khi tất cả các test method trong class thực thi xong.
* **@Disable –** Vô hiệu hoá một test method hay một test class thực thi (Tương ứng với @Ignore ở version cũ).
  1. Junit Vintage

#### JUnit Vintage hỗ trợ chạy thử nghiệm dựa trên JUnit 3 và JUnit 4 trên nền tảng JUnit 5.

#### 4.Assertion và Assumption

#### 4.1 Assertion

#### - **Assertion** dùng để kiểm tra kết quả của quá trình thực thi. Project phải luông cố gắng đạt đến trạng trái không có mệnh đề Assertion nào bị false vì bất cứ lý do gì.

#### - Với JUnit 5 chúng ta có thể sử dụng[lambda expression](https://shareprogramming.net/lambda-expression-trong-java/) chung với assertion.

#### 4.2 Assumption

#### - **Assumption** được sử dụng để đảm bảo rằng các dữ liệu được chuẩn bị cho test method phải đúng trước khi chúng thực thi. Nếu các điều kiện bên trong Assumption false, test method sẽ bị ignore.

#### 5. Exception testing

-Đôi lúc chúng ta mong đợi kết quả trả về là một exception vì dữ liệu của chúng ta không đúng hoặc dữ liệu xung đột etc. Sử dụng assertThrows() để mong đợi kết quả trả về là một exception.

**6. Test suite**

**-** JUnit Test suite cho phép chúng ta chạy một lúc nhiều test class

- JUnit cung cấp @SelectPackages and @SelectClasses để khởi tạo một test suite.

- **Lưu ý** :Để tạo test suite chúng ta phải thêm junit-platform-runner dependency cho maven.

## **7. Dynamic test**

- JUnit 5 cho phép chúng ta tạo các test case tạo thời điểm runtime bằng cách định nghĩa chúng bên trong method được bao bởi @TestFactory.

- @TestFactory method phải trả về các kiểu dữ liệu dạng danh sách như Stream, Collection, Iterable, etc.

- Ngoài ra @TestFactory method không được là static method hoặc access private.

**Maven/Gradle**

Maven

1.Maven là gì:

- Maven là một công cụ quản lý dự án mạnh mẽ dựa trên POM (mô hình đối tượng dự án). Nó được sử dụng cho các dự án xây dựng, phụ thuộc và tài liệu.

* Nó làm cho một dự án dễ xây dựng.
* Nó cung cấp quy trình xây dựng thống nhất (maven project có thể được chia sẻ bởi tất cả các dự án maven)
* Nó cung cấp thông tin dự án ( log document, cross referenced sources, mailing list, dependency list, unit test reports etc.)
* Dễ dàng di chuyển các tính năng mới.
* Maven giúp quản lý:
* Builds
* Documentation
* Reporing
* SCMs
* Releases
* Distribution

2.Maven-POM

- POM là từ viết tắt của Project Object Model. File pom.xml chứa thông tin về project và configuration để maven xây dựng project như dependencies, build directory, source directory, test source directory, plugin, goals...

- Maven đọc tệp pom.xml, sau đó thực thi mục tiêu.

-các yếu tố của tệp POM:

+project: Nó là phần tử gốc của tệp pom.xml.

+modelVersion: Nó là yếu tố phụ của dự án. Nó chỉ định modelVersion. Nó sẽ được đặt thành 4.0.0.

+groupID: Nó là yếu tố phụ của dự án. Nó chỉ định id cho nhóm dự án.

+artifacID:Nó là yếu tố phụ của dự án.Noschir định ID cho artifact(dự án).Một hiện vật là một cái gì đó được sản xuất hoặc sử dụng bởi một dự án.Ví dụ về các hiện vật do Maven sản xuất cho một dự án bao gồm:jARs,phân phố nguồn và nhị phân ,WARs.

-Các thành phần bổ sung của file POM:

+ **packaging:** xác định loại paking như jar,war

**+ name:** xác định tên proj

**+ url:** xác định url của proj

**+ dependencies:** xác định các phụ thuộ cho dự án

**+ dependency:** xác định một phụ thuộc. Nó được sử dụng bên trong các phụ thuộc.

**+ scope:** xác định scope cho maven project này. Nó có thể được biên dịch, cung cấp, thời gian chạy, kiểm tra và hệ thống.

3.Vòng đời

* *validate* – kiểm tra tính đúng đắn của dự án
* *compile* – biên dịch mã nguồn được cung cấp thành các tạo phẩm nhị phân
* *test* – thực hiện unit tests
* *package* – gói mã được biên dịch vào một tệp lưu trữ
* *integration-test* – hực hiện các kiểm tra bổ sung, yêu cầu đóng gói
* *verify* – kiểm tra xem gói có hợp lệ không
* *install* – cài đặt tệp gói vào kho lưu trữ Maven cục bộ
* *deploy* – deploys the package file to a remote server or repository

Doget.dopost,http,lifecycle class,MVC

LifeCycle Serverlet

-Serverlet được khởi tạo bằng cách khởi tạo phương thức init().

-Phương thức Serverlet service() được gọi để xử lý yêu cầu của khách hàng.

-Serverlet được hủy bằng cách gọi phương thức destroy().

-Cuối cùng,serverlet được thu thập bởi trình dọn rác của JVM.

-Doget:+là phương thức để xử lý các yêu cầu từ Client(hoặc trình duyệt) và viết phản hồi đã được định dạng trở lại client.

+mang theo dữ liệu trên thanh tiêu đề.

+Được sử dụng để truy vấn hoặc lấy thông tin từ máy chủ.

+Kích thước tối đa là 240 byte.

-Dopost:+ phương thức này được thiết kế để gửi lượng dữ liệu không giới hạn cùng với yêu cầu đến tài web resouses

-Http:+Là giao thức truyền tải siêu văn bản được sử dụng trong www dùng để truyền tải dữ liệu giữa web server đến các trình duyệt web và ngược lại.

+ Khi trình duyệt gửi yêu cầu đến web server thông qua giao thức http,web server sẽ nhận yêu cầu này và trả lại kết quả cho trình duyệt.

* Https là http có sử dụng thêm SSL để mã hóa dữ liệu trong lúc truyền tải dữ liệu nhằm mục đích bảo mật.Truy cập bằng https chậm hơn http.

**Logging**

1.Logging :là quá trình viết log message trong quá trình thực hiện chương trình đến một nơi cụ thể (console,file,databases).

2.Logger:là một object để thực hiện quá trình logging.

3.Logging level:

- DEBUG:

+là logging level thấp nhất.

+ Được sử dụng để debug các chương trình hoặc sản phẩm

+ Chỉ được sử dụng trong development và testing.

+ Khonng được sử dụng trong production vì sẽ ảnh hưởng đến performance.

-INFO:

+Bị giới hạn hơn so với debug.

+Mức nhật ký tiêu chuẩn chỉ ra rằng có điều gì đó đã xảy ra, ứng dụng đã vào một trạng thái nhất định.

+ Nó sẽ log lại những thông tin như là severs started,incoming message,outgoing message hoặc những activity thông thường của người dùng.

+Thông tin được ghi lại bằng log level INFO phải hoàn toàn mang tính thông tin và việc không xem xét chúng một cách thường xuyên sẽ không dẫn đến việc thiếu bất kỳ thông tin quan trọng nào.

-WARN:

+Bị giới hạn hơn so với INFO.

+Mức nhật ký chỉ ra rằng có điều gì đó không mong muốn đã xảy ra trong ứng dụng, sự cố hoặc tình huống có thể làm xáo trộn một trong các quy trình.

+Mức WARN nên được sử dụng trong các tình huống bất ngờ, nhưng code có thể tiếp tục công việc.

-ERROR:

+Bị hạn chế hơn WARN.

+Được sử dụng để log exception và error.

+ Phải có trong code.

+Mức nhật ký nên được sử dụng khi ứng dụng gặp sự cố ngăn một hoặc nhiều chức năng hoạt động bình thường.

+Mức nhật ký ERROR có thể được sử dụng khi một trong các hệ thống thanh toán không khả dụng, nhưng vẫn có tùy chọn kiểm tra giỏ hàng trong ứng dụng thương mại điện tử hoặc khi tùy chọn ghi nhật ký mạng xã hội của bạn không hoạt động vì một số lý do.