Cập nhật tháng 8 năm 2024

[Bài Đọc] Giới thiệu Java 8

Java 8, được phát hành vào tháng 3 năm 2014, là một trong những phiên bản quan trọng nhất trong lịch sử phát triển của Java. Phiên bản này mang lại nhiều cải tiến vượt bậc, đặc biệt tập trung vào hiệu năng, sự linh hoạt trong lập trình, và hỗ trợ lập trình hàm.

1. **Phương thức mặc định (Default method) và phương thức tĩnh (Static method) trong interface**

- **Phương thức mặc định (Default method)**
  - **Khái niệm:** Phương thức mặc định trong interface cho phép định nghĩa sẵn một hành vi chung trong interface mà không bắt buộc tất cả các lớp cài đặt (implementing classes) phải ghi đè (override).
  - **Lý do sử dụng**
    - Duy trì khả năng tương thích ngược khi thêm phương thức mới vào interface
    - Cho phép cài đặt sẵn logic mà các lớp có thể tái sử dụng
  - **Ví dụ**

```
interface Vehicle {
    default void start() {
        System.out.println("Starting vehicle...");
    }
}

class Car implements Vehicle {}

public class Main {
    public static void main(String[] args) {
        Car car = new Car();
        car.start(); // Output: Starting vehicle...
    }
}
```

- **Phương thức tĩnh (Static method)**
  - **Khái niệm:** Phương thức tĩnh trong interface là các phương thức được định nghĩa với từ khóa static. Chúng thuộc về chính interface, không phải đối tượng của

các lớp cài đặt

- **Lý do sử dụng**
  - Tổ chức các phương thức tiện ích liên quan trực tiếp đến interface
  - Tăng tính mô-đun và rõ ràng trong mã nguồn
- **Ví dụ**

```
interface Utility {
    static int add(int a, int b) {
        return a + b;
    }
}

public class Main {
    public static void main(String[] args) {
        int result = Utility.add(5, 10); // Gọi trực tiếp qua tên interface
        System.out.println(result); // Output: 15
    }
}
```

2. **Functional interface và biểu thức lambda**

- **Functional interface**
  - **Khái niệm:** Functional Interface là interface chỉ có duy nhất một phương thức trừu tượng. Chúng được sử dụng như các "hàm" trong lập trình hàm
  - **Chú thích:** Java cung cấp annotation @FunctionalInterface để đánh dấu functional interface
  - Ví dụ:

```
@FunctionalInterface
interface Calculator {
    int calculate(int a, int b);
}

public class Main {
    public static void main(String[] args) {
        Calculator sum = (a, b) -> a + b;
        System.out.println(sum.calculate(5, 3)); // Output: 8
    }
}
```

- **Biểu thức lambda**
  - **Khái niệm:** Lambda expression là một cách biểu diễn các phương thức trong Java theo cách ngắn gọn hơn, đặc biệt khi làm việc với functional interface
  - **Cú pháp:**

```
(parameters) -> expression
```

- **Ví dụ:**

```
Runnable r = () -> System.out.println("Hello Lambda!");
r.run(); // Output: Hello Lambda!
```

- **Lợi ích:**
  - Giảm mã lệnh dư thừa

- Tăng khả năng đọc và bảo trì mã nguồn

3. **Method references (Tham chiếu phương thức)**

- **Khái niệm:**
  - Method references là một cách ngắn gọn hơn để biểu diễn các biểu thức lambda, sử dụng tên phương thức đã tồn tại
- **Các loại Method references:**
  - Có 4 loại chính
  - Tham chiếu đến phương thức tĩnh (Static Method)
    - Cú pháp: **ClassName::methodName**
    - Ví dụ:

```
Consumer<String> print = System.out::println;
print.accept("Hello Method Reference!"); // Output: Hello Method Reference!
```

- Tham chiếu đến phương thức của một đối tượng (Instance Method)
  - Cú pháp: **objectName::methodName**
  - Ví dụ:

```
String str = "Java 8";
Supplier<String> supplier = str::toUpperCase;
System.out.println(supplier.get()); // Output: JAVA 8
```

- Tham chiếu đến phương thức của một lớp bất kỳ (Instance Method of Arbitrary Object)
  - Cú pháp: **ClassName::methodName**
  - Ví dụ:

```
List<String> names = Arrays.asList("Alice", "Bob", "Charlie");
names.forEach(String::toUpperCase); // Từng phần tử được chuyển thành chữ in hoa
```

- Tham chiếu đến Constructor (Constructor Reference)
  - Cú pháp: **ClassName::new**
  - Ví dụ:

```
Supplier<List<String>> listSupplier = ArrayList::new;
List<String> list = listSupplier.get();
System.out.println(list); // Output: []
```

Link tài nguyên đọc thêm:

- Default method vs Static method: https://www.baeldung.com/java-static-default-methods
- Functional interface: https://www.baeldung.com/java-8-functional-interfaces
- Lambda expression: https://www.geeksforgeeks.org/lambda-expressions-java-8/
- Method reference: https://www.baeldung.com/java-method-references