Ministry of SMEs and Startups × KOSME Korea SMEs and Startups Agency × LIKELION

Giới thiệu về Java 8

Nguyễn Anh Tuấn

KTECH COLLEGE

Nội dung bài giảng

1 Từ khoá this

2 Từ khoá super

3 Giới thiệu về Java 8

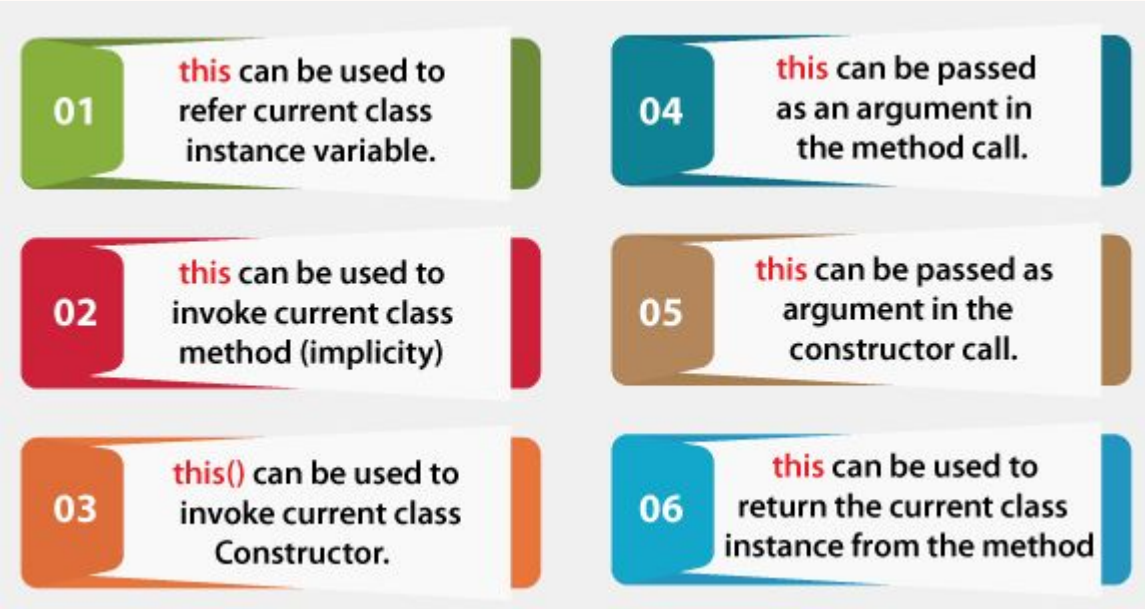
4 Default/static method trong interface

TỪ KHÓA THIS

TRONG JAVA

Từ khoá this

- **Định nghĩa**: **this** là "cái này", có nghĩa là chính đối tượng này. Nó tham chiếu ngược lại đến đối tượng, truy xuất đến các giá trị của đối tượng đó



Từ khoá this

❖ Tham chiếu tới thuộc tính của class hiện tại

```
class Dog {
        public String type="Animal";

        public Dog(String type) {
                // Gán giá trị của parameter là type vào biến type của class Dog
                this.type = type;
                System.out.println("I am a " + type);
        }
}
```

Từ khoá this

❖ Gọi phương thức của class hiện tại

```
class Dog {
        public String type="Animal";

        public Dog(String type) {
                this.type = type;
                // Gọi đến phương thức printText()
                this.printText();
                System.out.println("I am a " + type);
        }

        public void printText() {
                System.out.println(“Hello World”);
        }
}
```

Từ khoá this

❖ Gọi constructor của class hiện tại

```
class Dog {
        public String type="Animal";

        public Dog() { System.out.println("I am a " + type); }

        public Dog(String type) {
                this();
                this.type = type;
        }

        public void printText() {
                System.out.println(“I am a “ + type);
        }
}
```

Từ khoá this

❖ Trả về instance của class hiện tại, kiểu trả về của phương thức phải là class của lớp hiện tại

```
class Dog {
        public String type="Animal";
        public Dog setType(String type) {
                this.type = type;
                return this;
        }

        public void printText() {
                System.out.println(“I am a “ + type);
        }
}
```

Từ khoá this

❖ Được truyền như một tham số trong phương thức (method).

```
class Dog {
        public String type="Animal";

        Helper helper = new Helper();
        helper.print(this);
}

class Helper {
        public void print(Dog d) {
                System.out.println("name = " + d.type);
        }
}
```

Từ khoá this

❖ Được truyền như một tham số trong constructor.

```
class Dog {
        public String type="Animal";
        Helper helper = new Helper(this);
        helper.print(this);

        public void printId() {
                System.out.println("type = " + type);
         }

}

class Helper {
        public Helper(Dog d) { System.out.println("name = " + d.print(); }
}
```

TỪ KHÓA SUPER

TRONG JAVA

Từ khoá super

- **Định nghĩa**:
  - Từ khóa **super** là một biến tham chiếu được sử dụng để tham chiếu trực tiếp đến đối tượng của lớp cha gần nhất
  - Bất cứ khi nào tạo ra instance của lớp con, 1 instance của lớp cha được tạo ra ngầm định, nghĩa là được tham chiếu bởi biến **super**

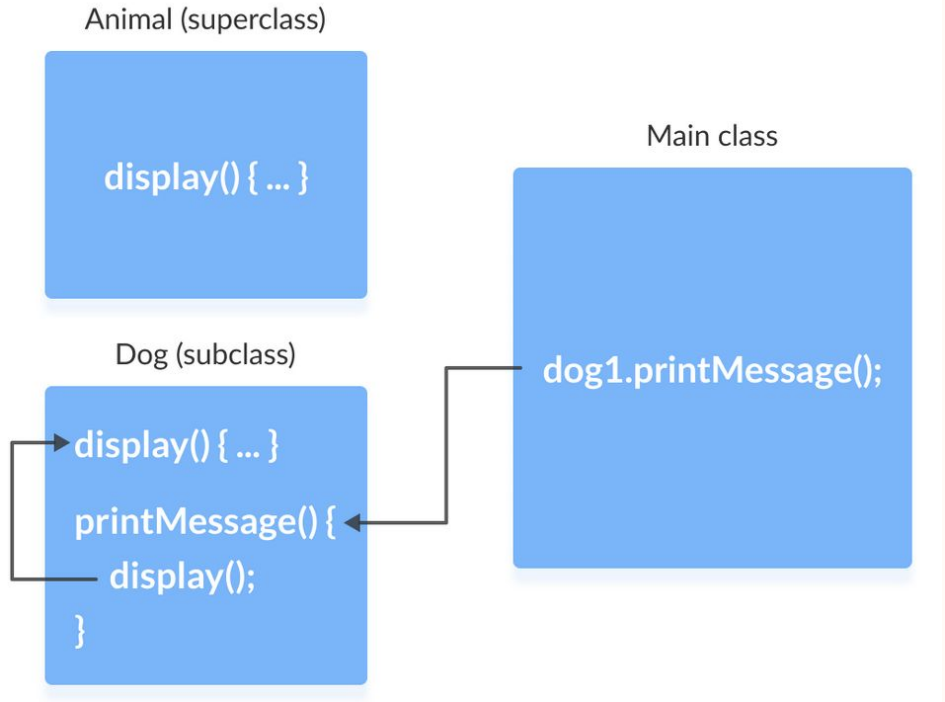
Từ khoá super

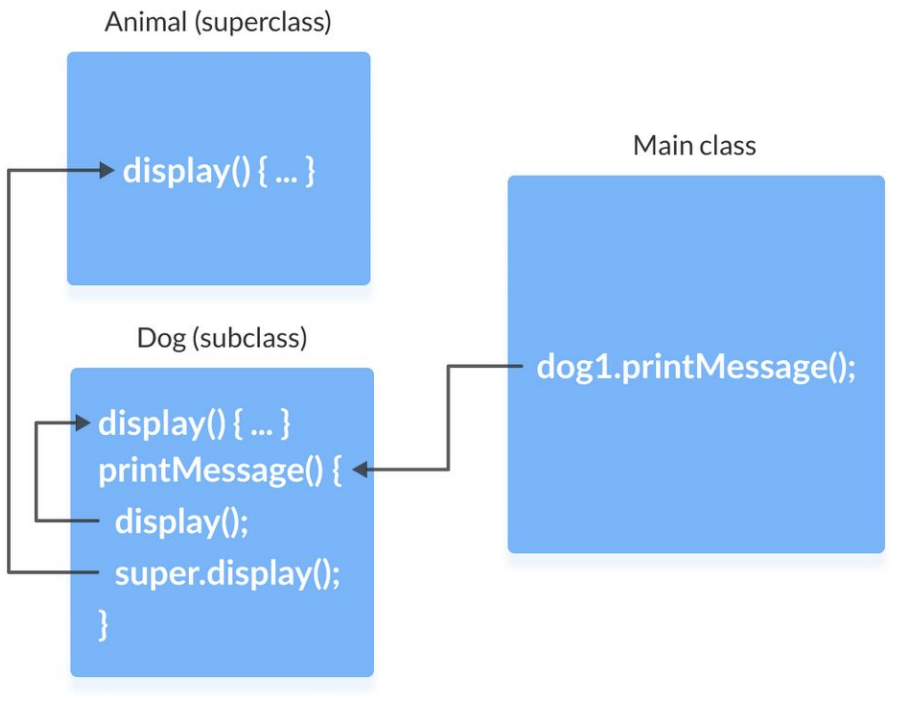
❖ Sử dụng super để gọi thuộc tính trùng tên trong superclass

```
class Animal {
        protected String type="animal";
}

class Dog extends Animal {
        // Biến type trong subclass trùng tên với biến type trong superclass
        public String type="mammal";

        public void printType() {
                System.out.println("I am a " + type);
                System.out.println("I am an " + super.type);
        }
}
```

Từ khoá super

❖ Sử dụng super để gọi phương thức bị ghi đè trong superclass Animal





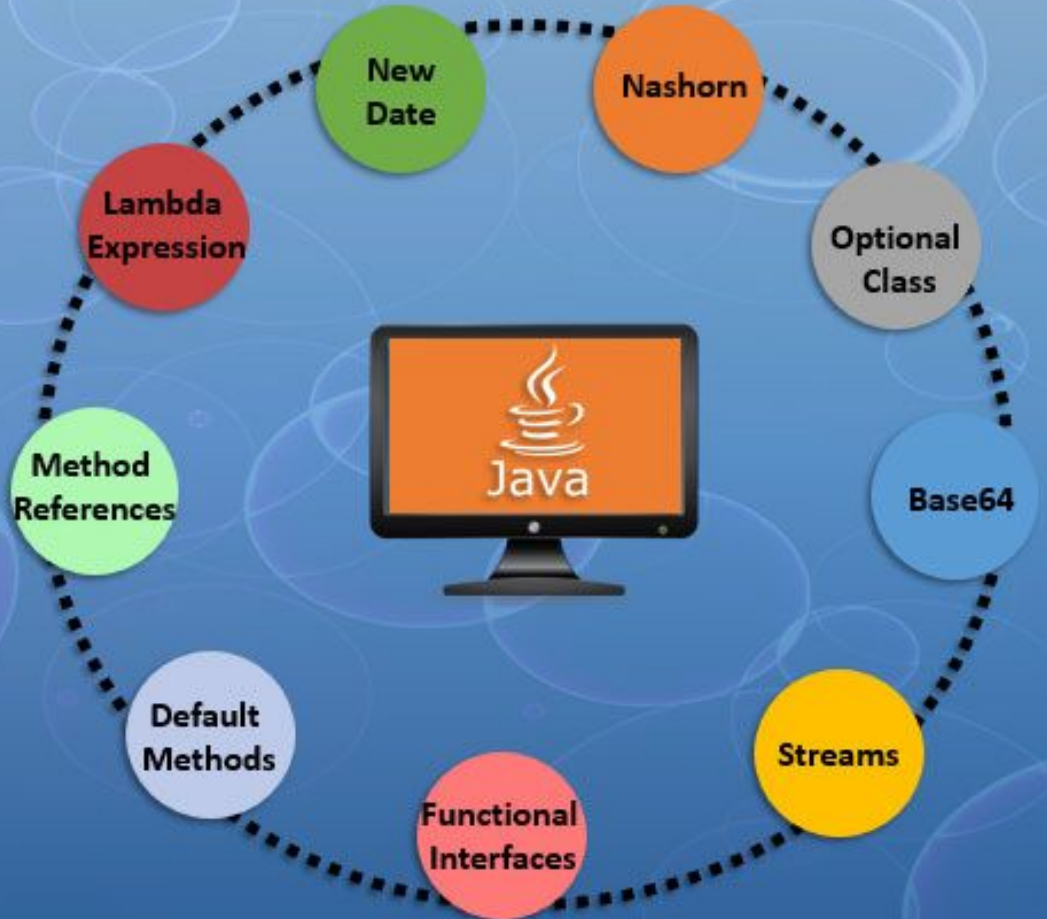
Từ khoá super

❖ Sử dụng super để gọi constructor mặc định hoặc constructor có đối số trong superclass

```
class Animal {
        public Animal() { System.out.println("Animal"); }
}

class Dog extends Animal {
        public Dog() {
                super();
                System.out.println("Dog");
        }
}

class Main {
        public static void main(String[] args) { Dog dog1 = new Dog(); }
}
```

Tại sao phải sử dụng super() nếu trình biên dịch tự động khởi tạo super()?

Java 8 Features

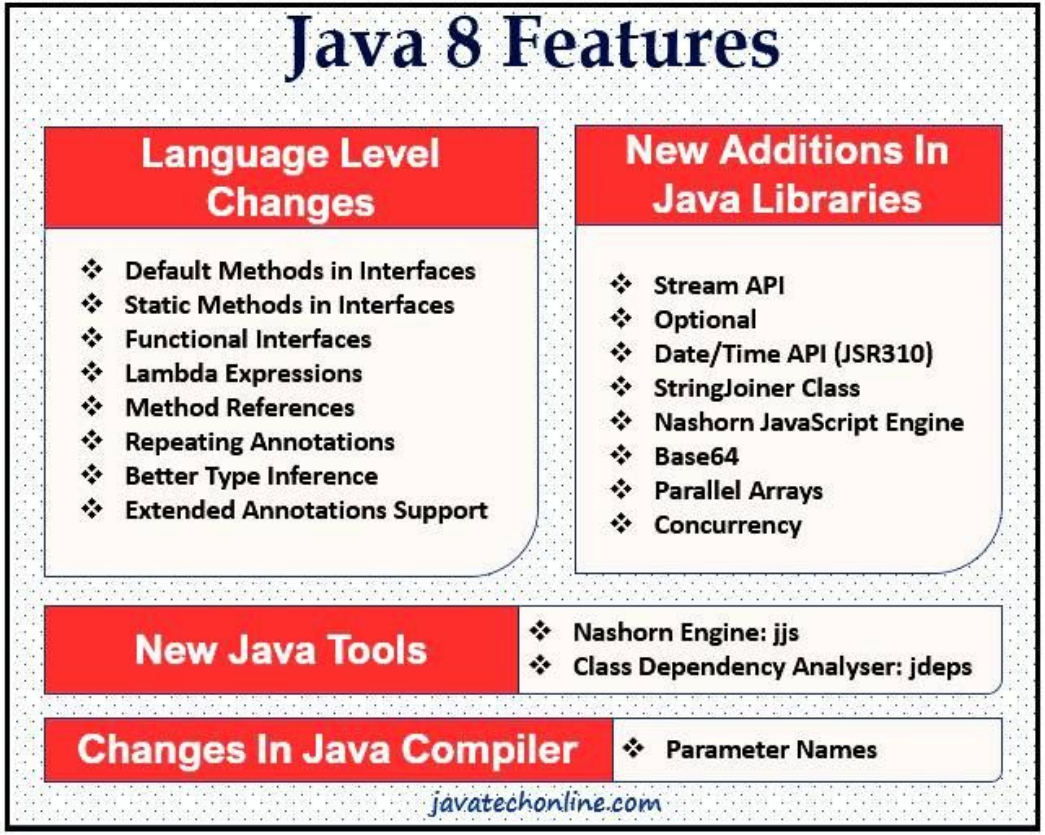


Giới thiệu về Java 8

- Oracle đã phát hành một phiên bản Java 8 vào ngày 18/03/2014. Đây là một phiên bản mang tính cách mạng của Java cho nền tảng phát triển phần mềm. Nó bao gồm các nâng cấp khác nhau cho lập trình Java, JVM, Tools và các thư viện

https://www.oracle.com/java/technologies/java-se-support-roadmap.html

Giới thiệu về Java 8



Giới thiệu về Java 8

- **Một số tính năng sẽ học trong khoá Java cơ bản**
  - **Default method:** Cung cấp phương thức mặc định cho Interface.
  - **Lambda expression:** Thêm khả năng xử lý function cho Java.
  - **Method references:** Các hàm tham chiếu theo tên của phương thức thay vì gọi trực tiếp. Sử dụng các function làm tham số.
  - **Stream API:** Bao gồm các class, interface và enum để cho phép các hoạt động kiểu function trên các element (phần tử) của một Collection, Array. Nó thực hiện chỉ khi nó yêu cầu (lazy).
  - **Optional:** Là một lớp được sử dụng để hạn chế với lỗi NullPointerException trong ứng dụng Java.

Default/Static method

Default method

❖ **Định nghĩa**:

- ➢ **Default Method** cho phép thêm vào các chức năng cho interface mà không làm phá vỡ các lớp implement từ interface này
- ➢ **Default Method** cho phép định nghĩa phần thân của phương thức
- ➢ Khi 1 class được implements từ interface có chứa phương thức **default**, nó không bắt buộc phải implement phương thức **default**

```
public interface Animal {
        // Khởi tạo phương thức default cho interface Animal
        // Các class implements từ interface này ko cần phải implements default method
        default void setColor(String color) {
                System.out.println("This animal has color is: " + color);
        }
}
```

Điều gì sẽ xảy ra nếu 1 class đa thừa kế các interface có default method cùng tên?

Static method

- **Định nghĩa**: **Static Method** cho phép thêm vào các chức năng cho interface mà không làm phá vỡ các lớp implement từ interface này
- **Static Method** cho phép định nghĩa phần thân của phương thức
- Khi 1 class được implements từ interface có chứa phương thức **static**, nó không bắt buộc phải implement phương thức **static**
- 1 lớp khi kết thừa static method thì không thể @override lại phương thức này. Nó có thể gọi trực tiếp thông qua class

```
public interface Animal {
        // Khởi tạo phương thức static cho interface Animal
        // Các class implements từ interface này ko cần phải implements static method
        // Các class implements từ interface này có thể gọi trực tiếp setColor từ class Animal
        static void setColor(String color) {
                System.out.println("This animal has color is: " + color);
        }
}
```

ROAD TO KOREA

Nếu có bất kỳ thắc mắc nào, hãy đặt câu hỏi qua

mail@mail.com hoặc Zalo 0xxx xxx xxx