**Bài 25 - Các loại hàm khởi tạo (constructor) của lớp trong Java**

**1. Hàm khởi tạo (constructor) là gì?**

**Hàm khởi tạo (constructor)** trong Java là một hàm có tên giống với tên lớp. Hàm này được gọi khi khởi tạo đối tượng. Hàm khởi tạo **không có kiểu trả về**. **Ví dụ**:

class Test {

Test() {

// constructor body

}

}

Một ví dụ khác về hàm khởi tạo.

class Main {

private String name;

//constructor

Main() {

System.out.println("Constructor Called:");

name = "Gochocit.com";

}

public static void main(String[] args) {

//hàm khởi tạo được gọi khi khởi tạo đối tượng

Main obj = new Main();

System.out.println("The name is " + obj.name);

}

}

**Kết quả**

Constructor Called:

The name is Gochocit.com

Lớp **Main** có một hàm khởi tạo là **Main()**. Hàm này khởi tạo giá trị cho thuộc tính **name**. Khởi tạo đối tượng với hàm khởi tạo:

Main obj = new Main();

Khi đó, đối tượng sẽ được phát vùng nhớ trên bộ nhớ **heap** để lưu trữ dữ liệu của đối tượng đó.

Trong Java, hàm khởi tạo có thể được chia thành 3 loại:

**1. Hàm khởi tạo không có tham số (no-argument constructor)**

**2. Hàm khởi tạo có tham số (parameterized constructor)**

**3. Hàm khởi tạo mặc định (default constructor)**

**2. Hàm khởi tạo không có tham số (no-argument constructor)**

Hàm khởi tạo của một lớp có thể không có tham số nào. Cú pháp:

class <Tênlớp>{

accessmodifier <Tênlớp>(){

//thân hàm khởi tạo không có tham số

}

}

Nếu **accessmodifier** là **private** thì hàm khởi tạo chỉ được sử dụng bên trong lớp đó. Nếu **accessmodifier** là **public** thì hàm khởi tạo có thể được sử dụng bên ngoài lớp đó.

**Ví dụ hàm khởi tạo với accessmodifier là private**

class Main {

private String name;

//private constructor

private Main() {

System.out.println("Constructor Called:");

name = "Gochocit.com";

}

public static void main(String[] args) {

//hàm khởi tạo được gọi khi khởi tạo đối tượng

Main obj = new Main();

System.out.println("The name is " + obj.name);

}

}

**Ví dụ hàm khởi tạo với accessmodifier là public**

class Company {

String name;

//public constructor

public Company() {

name = "Gochocit.com";

}

}

class Main {

public static void main(String[] args) {

// object is created in another class

Company obj = new Company();

System.out.println("Company name = " + obj.name);

}

}

**3. Hàm khởi tạo có tham số (parameterized constructor)**

Hàm khởi tạo trong Java có thể có 1 hoặc nhiều tham số.

class Company {

String name;

//public constructor

public Company(String cName) {

name = cName;

System.out.println("Company name is " + name);

}

}

class Main {

public static void main(String[] args) {

Company com1 = new Company("Google");

Company com2 = new Company("Apple");

Company com3 = new Company("Facebook");

}

}

**Kết quả**

Company name is Google

Company name is Apple

Company name is Facebook

Hàm khởi tạo **Company(String cName)** có 1 tham số.

**Nạp chồng hàm khởi tạo (constructor overloading)**

Trong một lớp, có thể có nhiều hàm khởi tạo cùng tên nhưng khác tham số. Ta gọi đó là nạp chồng hàm khởi tạo (constructor overloading).

class Company {

String name;

String country;

public Company(){

name = "Google";

System.out.println("Company name is " + name);

}

//public constructor

public Company(String cName) {

name = cName;

System.out.println("Company name is " + name);

}

//public constructor

public Company(String cName, String cCountry) {

name = cName;

country = cCountry;

System.out.println("Company name is " + name + " in " + country);

}

}

class Main {

public static void main(String[] args) {

Company com1 = new Company();

Company com2 = new Company("Apple");

Company com3 = new Company("Facebook", "USA");

}

}

**Kết quả**

Company name is Google

Company name is Apple

Company name is Facebook in USA

Có các hàm khởi tạo **Company()**, **Company(String cName)**, **Company(String cName, String cCountry)** cùng tên với nhau nhưng khác tham số.

**4. Hàm khởi tạo mặc định (default constructor)**

Nếu trong lớp không có hàm khởi tạo này, trình biên dịch Java sẽ tự động tạo ra một hàm khởi tạo không có tham số. Hàm khởi tạo này gọi là **hàm khởi tạo mặc định (default constructor)**.

class Main {

int a;

boolean b;

public static void main(String[] args) {

// A default constructor is called

Main obj = new Main();

System.out.println("Default Value:");

System.out.println("a = " + obj.a);

System.out.println("b = " + obj.b);

}

}

**Kết quả**

Default Value:

a = 0

b = false

Trong ví dụ trên, lớp Main không có hàm khởi tạo nào được định nghĩa. Trình biên dịch Java sẽ tự động tạo ra hàm default constructor. Hàm này sẽ tự động khởi tạo các thuộc tính với các giá trị mặc định.

|  |  |
| --- | --- |
| **Kiểu dữ liệu** | **Giá trị mặc định** |
| boolean | false |
| byte | 0 |
| short | 0 |
| int | 0 |
| long | 0L |
| char | \u0000 |
| float | 0.0f |
| double | 0.0d |
| object | Tham chiếu đến null |

**5. Một vài tóm tắt về hàm khởi tạo (constructor)**

Hàm khởi tạo được gọi khi khởi tạo đối tượng của lớp. Hàm khởi tạo có 2 đặc điểm là:

* Tên của hàm khởi tạo cùng tên với tên của lớp
* Hàm khởi tạo trong Java không có kiểu trả về (kể cả **void**)

Nếu một lớp không có hàm khởi tạo, trình biên dịch Java sẽ tự động tạo ra một hàm khởi tạo mặc định không có tham số. Hàm khởi tạo mặc định này sẽ tự khởi tạo các thuộc tính của đối tượng với các giá trị mặc định.

Có thể chia hàm khởi tạo thành 3 loại:

**1. Hàm khởi tạo không có tham số (no-argument constructor)**

**2. Hàm khởi tạo có tham số (parameterized constructor)**

**3. Hàm khởi tạo mặc định (default constructor)**

Hàm khởi tạo không thể khai báo với từ khóa abstract, static hoặc final.

Hàm khởi tạo có thể được nạp chồng nhưng không thể được ghi đè trong lớp con khi kế thừa.