🔥 What is Constructor Chaining in Java? Private constructors restrict object creation from outside classes.

- Constructor Chaining means: One constructor calling another constructor inside the same class or calling the parent class constructor (`super()`).
- It helps reuse code between multiple constructors.

📚 Types of Constructor Chaining

There are two types:

1. Within the same class → Using `this()`
2. From child class to parent class → Using `super()`

✅ Important Rules

- `this()` must be the first statement in the constructor.
- `super()` must also be the first statement if calling parent constructor.
- You cannot use both `this()` and `super()` together directly.
- If you don't call `super()` manually, Java automatically inserts `super()` (default no-arg) as the first line.

🚀 Example 1: Constructor Chaining within the same class (using `this()`)

java
CopyEdit

```
public class Student {
    private String name;
    private int age;

    // Constructor 1
    public Student() {
        this("Unknown", 0);   // Calling Constructor 2
        System.out.println("Default Constructor");
    }

    // Constructor 2
    public Student(String name) {
        this(name, 18);        // Calling Constructor 3
        System.out.println("Constructor with Name only");
    }

    // Constructor 3
    public Student(String name, int age) {
        this.name = name;
        this.age = age;
        System.out.println("Constructor with Name and Age");
    }

    public void display() {
        System.out.println("Name: " + name + ", Age: " + age);
    }

    public static void main(String[] args) {
        Student s1 = new Student();
        s1.display();

        Student s2 = new Student("Sasank");
        s2.display();

        Student s3 = new Student("Sasank", 25);
```

```
        s3.display();
    }
}
```

📦 Output

pgsql
CopyEdit

```
Constructor with Name and Age
Constructor with Name only
Constructor with Name and Age
Default Constructor
Name: Unknown, Age: 0
Constructor with Name and Age
Constructor with Name only
Name: Sasank, Age: 18
Constructor with Name and Age
Name: Sasank, Age: 25
```

🚀 Example 2: Constructor Chaining across parent and child classes (using `super()`)

java
CopyEdit

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

class Dog extends Animal {
    public Dog() {
        super(); // Call parent class constructor
        System.out.println("Dog Constructor");
    }
}
```

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

📦 Output

nginx
CopyEdit
`Animal Constructor`
`Dog Constructor`

✨ Summary of Constructor Chaining

Concept	What Happens
`this()`	Calls another constructor in same class
`super()`	Calls parent class constructor
Order	`this()` or `super()` must be first line
Goal	Code reuse, better organization, maintainability

🔥 Real-world thinking

- If an object can be created in multiple ways (with different parameters), constructor chaining reduces duplicate code.
- Also ensures consistent initialization of fields.

✅ **Now you know what is Constructor Chaining with simple examples and full explanation!**

Would you like me to show one real interview coding question where constructor chaining is used cleverly? 🚀