
What is a Method Reference?

A **method reference** is a shorthand notation of a **lambda expression** to call a method **directly by its name**.
It enhances code **readability and conciseness**, especially when the lambda expression simply calls an existing method.

Syntax of Method Reference:

ClassName::methodName

Depending on the context, ClassName can also be an object reference or a constructor.

Why Use Method References?

- To make code cleaner and more readable.
- To avoid unnecessary boilerplate when using functional interfaces.
- To reuse existing method definitions instead of rewriting them in lambdas.

Types of Method References:

Type	Syntax	Description
1. Reference to a static method	**ClassName::staticMethod**	Refers to a static method.
2. Reference to an instance method of a particular object	**object::instanceMethod**	Calls a method on a specific object.
3. Reference to an instance method of an arbitrary object of a particular type	**ClassName::instanceMethod**	Used in cases like sorting or mapping where the instance is passed implicitly.
4. Reference to a constructor	**ClassName::new**	Creates a new object using a constructor.

Relationship with Functional Interfaces:

Method references are often used with **functional interfaces** (like Runnable, Function, Supplier, etc.) because they match the interface's abstract method signature.

Behind the Scenes:

When the Java compiler sees a method reference, it **converts it to a lambda expression** internally that matches the target functional interface.