Method Overriding in Java

If subclass (child class) has the same method as declared in the parent class, it is known as **method overriding in java**.

**class** Vehicle{

**void** run()

{

System.out.println("Vehicle is running");

}

}

**class** Bike2 **extends** Vehicle{

**void** run()

{

System.out.println("Bike is running safely");

}

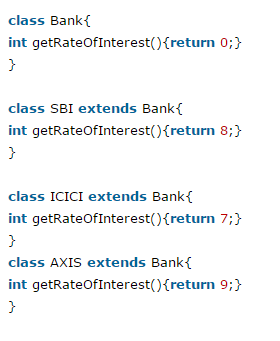
**public** **static** **void** main(String args[])

{

Bike2 obj = **new** Bike2();

obj.run();

}



### Can we override static method?

No, static method cannot be overridden. It can be proved by runtime polymorphism

### Why we cannot override static method?

because static method is bound with class whereas instance method is bound with object. Static belongs to class area and instance belongs to heap area.

### Can we override java main method?

No, because main is a static method.

# Method overloading and method overriding

|  |  |  |
| --- | --- | --- |
| **No.** | **Method Overloading** | **Method Overriding** |
| 1 | Method overloading is used *to increase the readability* of the program. | Method overriding is used *to provide the specific implementation* of the method that is already provided by its super class. |
| 2 | Method overloading is performed *within class*. | Method overriding occurs *in two classes* that have IS-A (inheritance) relationship. |
| 3 | In case of method overloading, *parameter must be different*. | In case of method overriding, *parameter must be same*. |
| 4 | Method overloading is the example of *compile time polymorphism*. | Method overriding is the example of *run time polymorphism*. |
| 5 | In java, method overloading can't be performed by changing return type of the method only. *Return type can be same or different* in method overloading. But you must have to change the parameter. | *Return type must be same or covariant* in method overriding. |
| 6 |  |  |