

Method Overriding in Oops

PREVIEW / RECAP:

We have seen how to create or call a method, static/non-static methods. Now let's understand the concept of Method Overloading

What is Method Overriding?

If the subclass (child class) has the same method as declared in the parent class, it is known as method overriding. In this case, the method in the parent class is called the overridden method and the method in the child class is called overriding method.

Method overriding is used to provide the specific implementation of a method that is already provided by its superclass.

Method overriding is used for runtime polymorphism

Eg.

Class A extends Class B. Both the classes have a common method void solve(). B class is giving its own implementation to the solve() method or in other words it is overriding the solve() method.

Usage of Java Method Overriding

```
class A{
    //Overridden method
    public void solve(){
        System.out.println("A solved it");
    }
}
class B extends A{
    //Overriding method
    public void solve(){
        System.out.println("B solved it");
    }
    public static void main( String args[]) {
        B obj = new B();
        //This will call the child class version of solve()
        obj.solve();
    }
}
```

OUTPUT


B solved it

Rules for Method Overriding:

- The argument list should be exactly the same as that of the overridden method.
- The return type should be the same or a subtype of the return type declared in the original overridden method in the superclass.
- If the superclass method is declared public then the overriding method in the subclass cannot be either private or protected.
- Instance methods can be overridden only if they are inherited by the subclass.
- A method declared final cannot be overridden.
- A method declared static cannot be overridden but can be re-declared.
- If a method cannot be inherited, then it cannot be overridden.
- A subclass within the same package as the instance's superclass can override any superclass method that is not declared private or final.
- A subclass in a different package can only override the non-final methods declared public or protected.
- Constructors cannot be overridden.

Using the super keyword:

When invoking a superclass version of an overridden method the super keyword is used.



```
class A{
    public void move() {
        System.out.println("A can move");
    }
}

class D extends A{
    public void move() {
        super.move();    // invokes the super class method
        System.out.println("D can walk and run");
    }
}

public class TD {
    public static void main(String args[]) {
        A b = new D();    // Animal reference but Dog object
        b.move();        // runs the method in Dog class
    }
}
```