

Polymorphism

Means "many forms",

**Polymorphism in Java** is a concept by which we can perform a single action in different ways. Polymorphism is derived from 2 Greek words: poly and morphs. The word "poly" means many and "morphs" means forms. So polymorphism means many forms.

There are two types of polymorphism in Java: compile-time polymorphism and runtime polymorphism. We can perform polymorphism in java by method overloading and method overriding.

Types:

> Compile time polymorphism

> Run-time polymorphism

Compile time polymorphism (Static polymorphism)

> Method overloading:

Multiple functions with same name, but different number/type of arguments.

> Operator overloading

In JAVA, only '+' can be overloaded.

+ used to add two numbers.

Also used to concatenate two Strings.

Run-time polymorphism (Dynamic Method Dispatch/Dynamic polymorphisma)

It is a process in which a function call to the overridden method is resolved at Runtime.

This is achieved by Method overriding

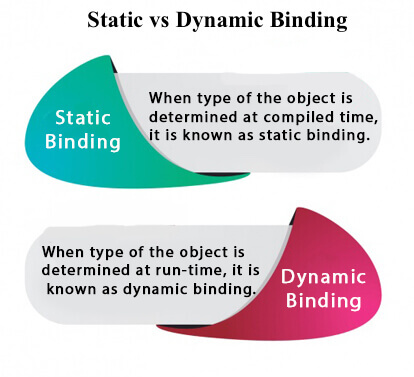
Method overriding

Occurs when a derived class has a definition for one of the member functions of the base class. That base function is said to be overridden.

Connecting a method call to the method body is known as binding.

There are two types of binding

1. Static Binding (also known as Early Binding).
2. Dynamic Binding (also known as Late Binding).



static binding

When type of the object is determined at compiled time(by the compiler), it is known as static binding.

If there is any private, final or static method in a class, there is static binding.

Example of static binding

1. **class** Redmi{
2. **private** **void** switchOn(){System.out.println("phone booting in 20 sec");}
4. **public** **static** **void** main(String args[]){
5. Redmi mi=**new** Redmi();
6. mi.switchOn();
7. }
8. }

Dynamic binding

When type of the object is determined at run-time, it is known as dynamic binding.

Example of dynamic binding

1. **class** Phone{
2. **void** switchOn(){System.out.println("phone takes 40 sec to boot");}
3. }
5. **class** Redmi **extends** Phone{
6. **void** switchoOn){System.out.println("Boots in 20 sec");}
8. **public** **static** **void** main(String args[]){
9. Phone a=**new** Redmi();
10. a.switchOn();
11. }
12. }