



Romel Cabiling ▾



Home

Home > My courses > 121 - CC106 > MODULE 11: WHAT ARE THE BASIC CONCEPTS OF POLUMORP... > 02A Lesson Proper for Week 11

02A Lesson Proper for Week 11

What are the Basic Concepts of Polymorphism as Feature of OOP and how to use it?

Polymorphism is the ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object.

Any Java object that can pass more than one IS-A test is considered to be polymorphic. In Java, all Java objects are polymorphic since any object will pass the IS-A test for their own type and for the class Object.

It is important to know that the only possible way to access an object is through a reference variable. A reference variable can be of only one type. Once declared, the type of a reference variable cannot be changed.

The reference variable can be reassigned to other objects provided that it is not declared final. The type of the reference variable would determine the methods that it can invoke on the object.

A reference variable can refer to any object of its declared type or any subtype of its declared type. A reference variable can be declared as a class or interface type.

Example

Let us look at an example.

```
public interface Vegetarian{}  
public class Animal{}  
public class Deer extends Animal implements Vegetarian{}
```

Now, the Deer class is considered to be polymorphic since this has multiple inheritance. Following are true for the above examples –

- A Deer IS-A Animal



- A Deer IS-A Vegetarian
- A Deer IS-A Deer
- A Deer IS-A Object

When we apply the reference variable facts to a Deer object reference, the following declarations are legal.

Example

```
Deer d = new Deer();  
Animal a = d;  
Vegetarian v = d;  
Object o = d;
```

All the reference variables d, a, v, o refers to the same Deer object in the heap.

Virtual Methods

In this section, the behavior of **overridden methods** will be shown in Java, and it allows you to take advantage of polymorphism when designing your classes.

Method overriding is where a child class can override a method in its parent. An overridden method is essentially hidden in the parent class and is not invoked unless the child class uses the super keyword within the overriding method.

Example

```
/* File name : Employee.java */  
public class Employee {  
    private String name;  
    private String address;  
    private int number;  
  
    public Employee(String name, String address, int number) {  
        System.out.println("Constructing an Employee");  
        this.name = name;  
        this.address = address;  
        this.number = number;  
    }  
    public void mailCheck() {  
        System.out.println("Mailing a check to " + this.name + " " + this.address);  
    }  
    public String toString() {  
        return name + " " + address + " " + number;  
    }  
    public String getName() {  
        return name;  
    }  
}
```



```

    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String newAddress) {
        address = newAddress;
    }
    public int getNumber() {
        return number;
    }
}

```

Now suppose we extend Employee class as follows –

```

/* File name : Salary.java */
public class Salary extends Employee {
    private double salary; // Annual salary

    public Salary(String name, String address, int number, double salary) {
        super(name, address, number);
        setSalary(salary);
    }

    public void mailCheck() {
        System.out.println("Within mailCheck of Salary class ");
        System.out.println("Mailing check to " + getName()
            + " with salary " + salary);
    }

    public double getSalary() {
        return salary;
    }

    public void setSalary(double newSalary) {
        if(newSalary >= 0.0) {
            salary = newSalary;
        }
    }

    public double computePay() {
        System.out.println("Computing salary pay for " + getName());
        return salary/52;
    }
}

```

Now, you study the following program carefully and try to determine its output –

```

/* File name : VirtualDemo.java */
public class VirtualDemo {

    public static void main(String [] args) {
        Salary s = new Salary("Mohd Mohtashim", "Ambehta, UP", 3, 3600.00);
        Employee e = new Salary("John Adams", "Boston, MA", 2, 2400.00);
        System.out.println("Call mailCheck using Salary reference --");
        s.mailCheck();
        System.out.println("\n Call mailCheck using Employee reference--");
        e.mailCheck();
    }
}

```



This will produce the following result –

Output

```
Constructing an Employee  
Constructing an Employee
```

```
Call mailCheck using Salary reference --  
Within mailCheck of Salary class  
Mailing check to Mohd Mohtashim with salary 3600.0
```

```
Call mailCheck using Employee reference--  
Within mailCheck of Salary class  
Mailing check to John Adams with salary 2400.0
```

Here, we instantiate two Salary objects. One using a Salary reference **s**, and the other using an Employee reference **e**.

While invoking *s.mailCheck()*, the compiler sees mailCheck() in the Salary class at compile time, and the JVM invokes mailCheck() in the Salary class at run time.

mailCheck() on **e** is quite different because **e** is an Employee reference. When the compiler sees *e.mailCheck()*, the compiler sees the mailCheck() method in the Employee class.

Here, at compile time, the compiler used mailCheck() in Employee to validate this statement. At run time, however, the JVM invokes mailCheck() in the Salary class.

This behavior is referred to as **virtual method invocation**, and these methods are referred to as virtual methods. An **overridden method** is invoked at run time, no matter what data type the reference is that was used in the source code at compile time.

Usages and Advantages of Polymorphism

- Method overloading allows methods that perform similar or closely related functions to be accessed through a common name. For example, a program performs operations on an array of numbers which can be int, float, or double type. Method overloading allows you to define three methods with the same name and different types of parameters to handle the array of operations.
- Method overloading can be implemented on constructors allowing different ways to initialize objects of a class. This enables you to define multiple constructors for handling different types of initializations.
- Method overriding allows a sub class to use all the general definitions that a super class provides and add specialized definitions through overridden methods.
- Method overriding works together with inheritance to enable code reuse of existing classes without the need for re-compilation.

Differences Between Compile-time and Run-time Polymorphism

- Method overloading occurs at compile-time whereas method overriding occurs at run-time.
- Method overloading occurs when the type signatures of two methods are different whereas method overriding occurs only when the type signatures of two methods are the same.
- In method overloading, the compiler determines the correct method to be called by comparing type signatures. In method overriding, the JVM determines the correct method to be called based on the object that the instance variable refers to.
- Method overloading is possible on methods with private, static, and final access modifiers whereas method overriding is not possible on these access modifiers.



EXERCISE 1

A student is to create a simple Java program about different types of persons. The Person class, the Student class, the Employee class and You class. How many classes will be defined based on the problem and identify the parent and children classes.

EXERCISE 2

From the previous exercise (Exercise 1), identify the methods of the Person, Student and Employee classes.

◀ Preliminary Activity for Week 11

Jump to...



Analysis, Application, and Exploration for Week 11 ▶

Navigation

Home

 Dashboard

Site pages

My courses

121 - CC106

Participants

 Grades

General

MODULE 1: WHAT IS APPLICATION DEVELOPMENT?

MODULE 2: WHAT ARE THE TECHNICAL SKILLS REQUIRED I...

MODULE 3: WHAT ARE THE PROGRAMMING LANGUAGES USED ...

MODULE 4: WHAT IS JAVA PROGRAMMING LANGUAGE AS APP...

MODULE 5: HOW TO WRITE JAVA PROGRAMMING LANGUAGE A...

MODULE 6: PRELIMINARY EXAMINATION

MODULE 7: HOW TO WRITE JAVA PROGRAM USING INTEGRAT...

MODULE 8: WHAT ARE THE BUILDING BLOCKS OF OBJECT-O...

MODULE 9: WHAT ARE THE BASIC CONCEPTS OF INHERITAN...


MODULE 10: WHAT ARE THE BASIC CONCEPTS OF ENCAPSUL...




MODULE 11: WHAT ARE THE BASIC CONCEPTS OF POLYMORPHISM...

 Preliminary Activity for Week 11

 **02A Lesson Proper for Week 11**

 Analysis, Application, and Exploration for Week 11

 Generalization for Week 11

 Evaluation for Week 11

 Assignment for Week 11

Week 12: Midterm Examination

MODULE 13: WHAT ARE THE BASIC CONCEPTS OF ABSTRACT...

MODULE 14: HOW TO WRITE JAVA PROGRAM USING ABSTRACT...

MODULE 15: WHAT IS JAVA DATABASE CONNECTIVITY (JDBC)...

MODULE 16: WHAT ARE THE STEPS OF MANIPULATING DATA...

MODULE 17: EMERGING TECHNOLOGIES

121 - BPM101 / DM103

121 - OAELEC2

121 - ITE3

121 - MUL101

121 - ITSP2B

121 - WEB101 / CCS3218

Courses

Fair Warning

NOTICE: Please be reminded that it has come to the attention of the Publishing Team of eLearning Commons that learning materials published and intended for ***free use only by students and faculty members within the eLearning Commons network were UNLAWFULLY uploaded in other sites without due and proper permission.***





PROSECUTION: Under Philippine law (Republic Act No. 8293), copyright infringement is punishable by the following: Imprisonment of between 1 to 3 years and a fine of between 50,000 to 150,000 pesos for the first offense. Imprisonment of 3 years and 1 day to six years plus a fine of between 150,000 to 500,000 pesos for the second offense.

COURSE OF ACTION: Whoever has maliciously uploaded these concerned materials are hereby given an ultimatum to take it down within 24-hours. Beyond the 24-hour grace period, our Legal Department shall initiate the proceedings in coordination with the National Bureau of Investigation for IP Address tracking, account owner identification, and filing of cases for prosecution.





Activities

-  Assignments
-  Forums
-  Quizzes
-  Resources

Bestlink College of the Philippines
College Department

Powered by [eLearning Commons](#)

