Trừu tượng hóa trong Java

Trừu tượng hóa (Abstraction) trong Java hướng đến khả năng tạo một đối tượng trừu tượng trong lập trình hướng đối tượng. Một lớp trừu tượng là một lớp mà không được khởi tạo. Tất cả các chức năng khác của lớp vẫn tồn tại, và tất cả các trường, phương thức, và hàm khởi tạo đều được truy cập với một cách giống nhau. Bạn không thể tạo một đối tượng với một lớp trừu tượng hóa.

Nếu một lớp là lớp trừu tượng và nó không được thuyết minh, lớp này không được sử dụng trừ khi nó là lớp con.

Lớp Abstract trong Java

Sử dụng từ khóa **abstract** để khai báo một lớp abstract. Từ khóa này xuất hiện trước từ khóa class trong khai báo lớp.

```
/* File name : Employee.java */
public abstract 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 double computePay()
   {
     System.out.println("Inside Employee computePay");
     return 0.0;
   }
   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;
   }
}
```

Bạn chú ý rằng không có gì khác trong lớp Employee này. Lớp này bây giờ là abstract, nhưng nó vẫn có 3 trường, 7 phương thức và một constructor.

Bây giờ, nếu chúng ta thử làm như sau:

```
/* File name : AbstractDemo.java */
public class AbstractDemo
{
   public static void main(String [] args)
   {
```

```
      /* Following is not allowed and would raise error */
      Employee e = new Employee("George W.", "Houston, TX", 43);

      System.out.println("\n Call mailCheck using Employee reference--");
      e.mailCheck();
    }
}
```

Khi bạn biên dịch lớp trên, bạn sẽ nhận một lỗi:

```
Employee.java:46: Employee is abstract; cannot be instantiated
      Employee e = new Employee("George W.", "Houston, TX", 43);
                   ^
1 error
```

Kế thừa lớp Abstract trong Java

Chúng ta có thể kế thừa lớp Employee theo cách thông thường như sau:

```
/* 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;
   }
}
```

Ở đây, chúng ta không thể thuyết minh một Employee mới, nhưng nếu chúng ta thuyết minh một đối tượng Salary mới, đối tượng Salary này sẽ kế thừa 3 trường, 7 phương thức từ Employee.

```
/* File name : AbstractDemo.java */
public class AbstractDemo
{
   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();
```

```
    }
}
```

Nó sẽ cho kết quả sau:

```
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.
```

Phương thức của lớp Abstract trong Java

Nếu bạn muốn một lớp chứa một phương thức cụ thể nhưng bạn muốn triển khai thực sự phương thức đó để được quyết định bởi các lớp con, thì bạn có thể khai báo phương thức đó trong lớp cha ở dạng abstract.

Từ khóa abstract được sử dụng để khai báo một phương thức dạng abstract. Một phương thức gồm một ký số, và không có thân phương thức.

Phương thức abstract sẽ không có định nghĩa, và ký số của nó được theo sau bởi dấu chấm phảy, không có dấu ngoặc móc ôm theo sau:

```
public abstract class Employee
{
   private String name;
   private String address;
   private int number;


   public abstract double computePay();


   //Remainder of class definition
```

```
}
```

Khai báo một phương thức dạng abstract tạo hai kết quả sau:

- Lớp phải được khai báo abstract. Nếu một lớp chứa một phương thức abstract, thì lớp đó cũng phải là abstract.
- Bất kỳ lớp con nào phải hoặc override phương thức abstract hoặc khai bao abstract chính nó.

Một lớp con mà kế thừa một phương thức abstract phải ghi đè nó. Nếu nó không, thì nó phải là abstract và bất kỳ lớp con nào của chúng phải override nó.

Cuối cùng, một lớp con phải triển khai phương thức abstract, nếu không thì bạn sẽ có một cấu trúc phân cấp của các lớp abstract mà không thể được thuyết minh.

Nếu Salary đang kế thừa lớp Employee, thì nó cần triển khai phương thức computePay() như sau:

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

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

   //Remainder of class definition
}
```