**Interface trong Java**

**1. Interface trong Java**

### **1.1. Khái niệm**

Trong Java, Interface (giao diện) là một kiểu dữ liệu tham chiếu tương tự như Class (lớp) nhưng chỉ có thể chứa hằng số và tên các phương thức, không có phần thân phương thức (phương thức trừu tượng). Một lớp mô tả các thuộc tính và hành động của đối tượng còn Interface thì mô tả các hành động của lớp đó. Interface không thể được khởi tạo như lớp mà chỉ có thể được mở rộng từ các lớp khác hoặc được kế thừa từ các Interface khác.

Trong Interface, chúng ta không thể khai báo hàm tạo và Interface không thể kế thừa từ một lớp mà chỉ có thể được mở rộng từ lớp và một Interface có thể được kế thừa lại từ nhiều Interface khác.

Ngoại trừ lớp trừu tượng thì tất cả các lớp mở rộng Interface phải định nghĩa lại tất cả các phương thức của Interface.

*Bài viết này được đăng tại [free tuts .net]*

### **Khai báo Interface**

Trong Eclipse, để tạo mới một Interface, chúng ta sẽ kích chuột phải vào tên package chọn File → New → Interface và tiến hành đặt tên Interface. Trong ví dụ dưới đây tôi sẽ tạo mới 1 Interface có tên là InterfaceJava, sau khi tạo chúng ta sẽ có 1 file InterfaceJava.java với nội dung như sau:

**InterfaceJava.java**

|  |  |
| --- | --- |
| 1  2  3  4  5 | package vidu;    public interface InterfaceJava {    } |

Từ khóa public cho biết Interface này có thể sử dụng ở bất kỳ package nào. Nếu chúng ta không khai báo phạm vi truy cập cho Interface thì mặc định Interface này chỉ có thể được sử dụng trong package chứa nó.

Một Interface có thể kế thừa nhiều Interface khác. Khác với lớp là mỗi lớp chỉ có thể kế thừa một lớp thì Interface có thể kế thừa từ nhiều Interface khác, danh sách các Interface được kế thừa cách nhau bởi dấu phẩy.

Trong các bài trước, chúng ta đã tìm hiểu về từ khóa abstract thì đối với một lớp có các phương thức là phương thức trừu tượng thì lớp đó bắt buộc phải là một lớp trừu tượng, vì vậy chúng ta phải sử dụng từ khóa abstract,nhưng đối với Interface chúng ta không cần phải sử dụng từ khóa abstract khi khai báo nó. Tương tự, các phương thức là trừu tượng trong Interface cũng không cần phải sử dụng từ khóa abstract mà chỉ cần khai báo tên phương thức, phạm vi truy cập, kiểu trả về của phương thức và kết thúc bằng dấu chấm phẩy.

Tất cả các phương thức trong Interface đều mặc định là public nên chúng ta không cần khai báo public và tương tự, các thuộc tính trong Interface phải là một hằng số và chúng ta cũng không cần phải khai báo static final và chỉ cần khai báo kiểu dữ liệu, tên biến và giá trị khởi tạo của hằng số đó. Tóm lại, bên trong Interface chỉ có hằng số hoặc tên phương thức, phương thức không chứa phần thân (tức là không có phần nằm trong dấu { } và kết thúc bằng dấu chấm phẩy) hoặc cả hai.

### **Implement Interface (tạm gọi là mở rộng, thực hiện Interface)**

Khi một lớp mở rộng 1 Interface thì lớp đó phải định nghĩa lại tất cả các phương thức của Interface đó. Trong trường hợp lớp đó không định nghĩa lại các phương thức của Interface thì nó phải được khai báo là một lớp trừu tượng.

Nếu một lớp mở rộng 1 Interface thì trong lớp đó chúng ta phải sử dụng từ khóa implements, phía sau là tên của Interface. Nếu lớp này vừa kế thừa từ một lớp khác và vừa mở rộng 1 Interface thì chúng ta sẽ tiến hành kế thừa trước sau đó mở rộng Interface (tức là từ khóa extends sẽ luôn đứng trước từ khóa implements).

**Interface.java**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | package vidu;    public interface Interface {        // Đây là phương thức trừu tượng      // phương thức trừu tượng của Interface không cần khai báo từ khóa abstract và public      void createAbstractMethod();    } |

**ClassImplementsInterface.java**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | package vidu;    public class ClassImplementsInterface implements Interface {        // Lớp này sau khi tạo ra      // chỉ có phương thức createAbstractMethod() của Interface      @Override      public void createAbstractMethod() {          System.out.println("Đây là phương thức createAbstractMethod() của Interface");      }        public static void main(String[] args) {          ClassImplementsInterface classImplementsInterface = new ClassImplementsInterface();          classImplementsInterface.createAbstractMethod();      }  } |

**Kết quả sau khi biên dịch chương trình:**

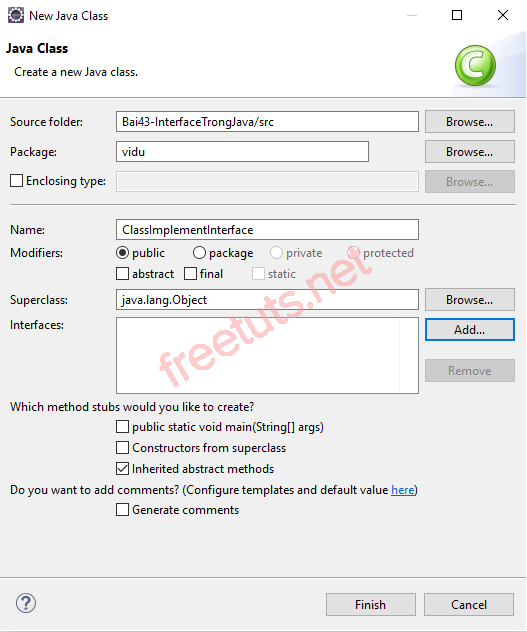


**Giải thích hoạt động của chương trình trên:**

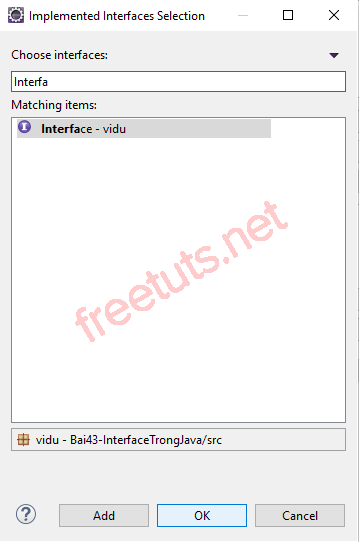
Trong chương trình trên tôi có tạo ra 1 Interface, Interface này có 1 phương thức trừu tượng là createAbstractMethod() (phương thức này không có thân hàm và kết thúc bằng một dấu chấm phẩy). Lớp ClassImplementInterface là một lớp mở rộng Interface trên nên nó sẽ có phương thức createAbstractMethod() (bắt buộc phải có). Sau đó, tôi tiến hành khởi tạo đối tượng của lớp đó và gọi phương thức createAbstractMethod(). Kết quả sẽ hiển thị dòng "Đây là phương thức createAbstractMethod() của Interface" ra màn hình Console.

**Lưu ý:**Để tạo một lớp implements Interface, các bạn làm như sau.

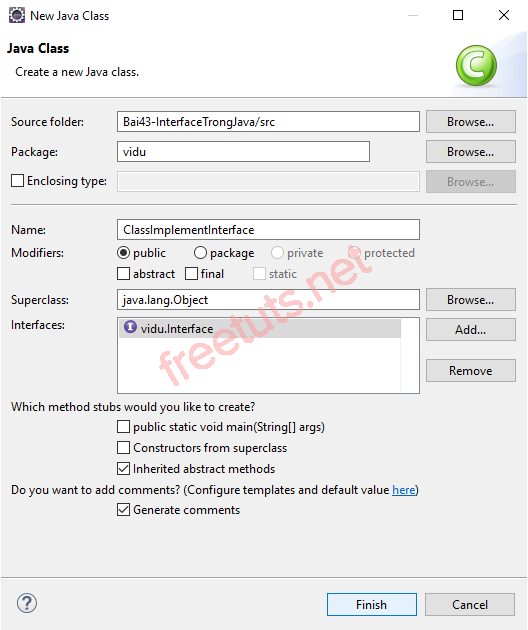
Bước 1: Trong hộp thoại tạo lớp, ngay dòng Interfaces các bạn bấm vào nút Add bên cạnh:



Bước 2: Hộp thoại Implemented Interfaces Selection xuất hiện. Trong ô Choose interfaces các bạn nhập vào tên Interface, sau đó tên của Interface sẽ hiển thị trong ô Matching items, bạn chọn vào đó và nhấn OK để kết thúc.



Bước 3: Tên của Interface vừa chọn sẽ hiển thị trong ô Interfaces:



Bước 4: Nhấn Finish để kết thúc, lúc này ta sẽ có lớp ClassImplementsInterface như sau:

**ClassImplementInterface.java**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | package vidu;    public class ClassImplementInterface implements Interface {        @Override      public void createAbstractMethod() {          // TODO Auto-generated method stub        }    } |

### **Kế thừa từ Interface**

Một Interface có thể kế thừa từ Interface khác tương tự như cách mà một lớp kế thừa từ một lớp khác đó là chúng ta cũng sử dụng từ khóa extends. Một lớp mở rộng từ Interface con sẽ kế thừa tất cả các phương thức có trong Interface đó (tức là phương thức của Interface con và Interface cha của Interface mà lớp đó implement). Trong Java, 1 Interface có thể kế thừa từ nhiều Interface và các Interface này được ngăn cách nhau bởi dấu phẩy.

**Person.java**

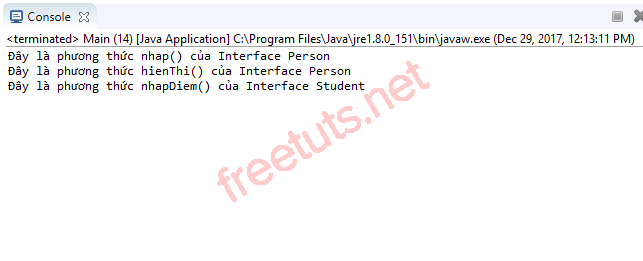
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | package vidu;    public interface Person {        void nhap();      void hienThi();    } |

**Student.java**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | package vidu;    public interface Student extends Person {        void nhapDiem();    } |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | package vidu;    public class Main implements Student {        @Override      public void nhap() {          System.out.println("Đây là phương thức nhap() của Interface Person");      }        @Override      public void hienThi() {          System.out.println("Đây là phương thức hienThi() của Interface Person");      }        @Override      public void nhapDiem() {          System.out.println("Đây là phương thức nhapDiem() của Interface Student");      }        public static void main(String[] args) {          Main demo = new Main();          demo.nhap();          demo.hienThi();          demo.nhapDiem();      }  } |

**Kết  quả sau khi biên dịch chương trình:**



**Giải thích hoạt động của chương trình trên:**

Trong chương trình trên, tôi tạo 1 Interface có tên là Person và 1 Interface con Student kế thừa từ Interface Person và khai báo phương thức nhapDiem() cho Interface này. Sau đó tôi khai báo lớp Main là lớp mở rộng của Interface Student, lúc này trong lớp Main sẽ có 3 phương thức đó là phương thức nhap() và hienThi() của Interface Person, phương thức nhapDiem() của Interface Student. Trong hàm main(), tôi tiến hành tạo đối tượng và gọi đến 3 phương thức trên, lúc đó chúng ta sẽ có kết quả biên dịch như hình trên.

### **Dùng Interface làm kiểu dữ liệu**

Việc định nghĩa một Interface giống như định nghĩa một kiểu dữ liệu tham chiếu mới. Nếu chúng ta khai báo một đối tượng có kiểu dữ liệu là một Interface thì đối tượng đó phải được khởi tạo là một đối tượng của lớp implement Interface. Ví dụ chúng ta sẽ sửa nội dung của lớp Main như sau:

**Main.java**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | package vidu;    public class Main implements Student {        @Override      public void nhap() {          System.out.println("Đây là phương thức nhap() của Interface Person");      }        @Override      public void hienThi() {          System.out.println("Đây là phương thức hienThi() của Interface Person");      }        @Override      public void nhapDiem() {          System.out.println("Đây là phương thức nhapDiem() của Interface Student");      }        public static void main(String[] args) {          Student student = new Main();          student.nhap();          student.hienThi();          student.nhapDiem();      }  } |

Kết quả sau khi biên dịch chương trình:

