Difference Between Inheritance and Interface in Java

www.differencebetween.com

Key Difference - Inheritance vs Interface in Java

Java is a programming language developed by Sun Microsystems. Java can be used to develop various applications. It is a multi-paradigm language which supports object-oriented, structured etc. The main advantage of Java is that it supports Object-Oriented programming(OOP). The programmer can create classes and create objects. There are four pillars in OOP. They are inheritance, abstraction, polymorphism and encapsulation. Inheritance and interfaces are related to OOP. The **key difference** between inheritance and interface is that **inheritance is to derive new classes from existing classes and an interface is to implement abstract classes and multiple inheritance.**

What is Inheritance in Java?

Inheritance can achieve code re usability. Inheritance helps to reuse the attributes and methods of an existing class. The mechanism of deriving new class using old class is called inheritance. The old class is known as parent class or super class. The derived class is called child class or subclass.

The syntax of Java inheritance is as follows.

class subclass_name extends superclass_name {

variable declaration;

method declaration;

}

Inheritance concept can be explained using the following example. Assume that there is a class called A as follows.

public class A{

public void sum(){

System.out.println(“Sum”);

}

}

If we want to add a new method without changing the existing class, we can do it as follows.

```
public class B{

public void sub(){

System.out.println(“Sub”);

}

}
```

Programmer can use inheritance to use class A sum().

```
public class B extends class A{

public void sub(){

System.out.println(“Sub”);

}

}
```

In the main function, it is possible to create an object of B and call sub() which belongs to class B and sum() which belongs to class A using inheritance.

```
public static void main(String[] args){

B obj = new B();

obj.sub();

obj.sum();

}
```

There are different types of inheritance. They are single inheritance, multiple inheritance, multi-level inheritance and hierarchical inheritance. In single inheritance, there is one base class and one derived class. In multi-level inheritance, there are three classes namely, base class, intermediate class and derived class. The intermediate class inherits from the base class, and derived class inherits from intermediate class. In hierarchical inheritance, there is one base class and many derived classes. There is a

special type known as Hybrid inheritance. It is a combination of two or more types of inheritance.

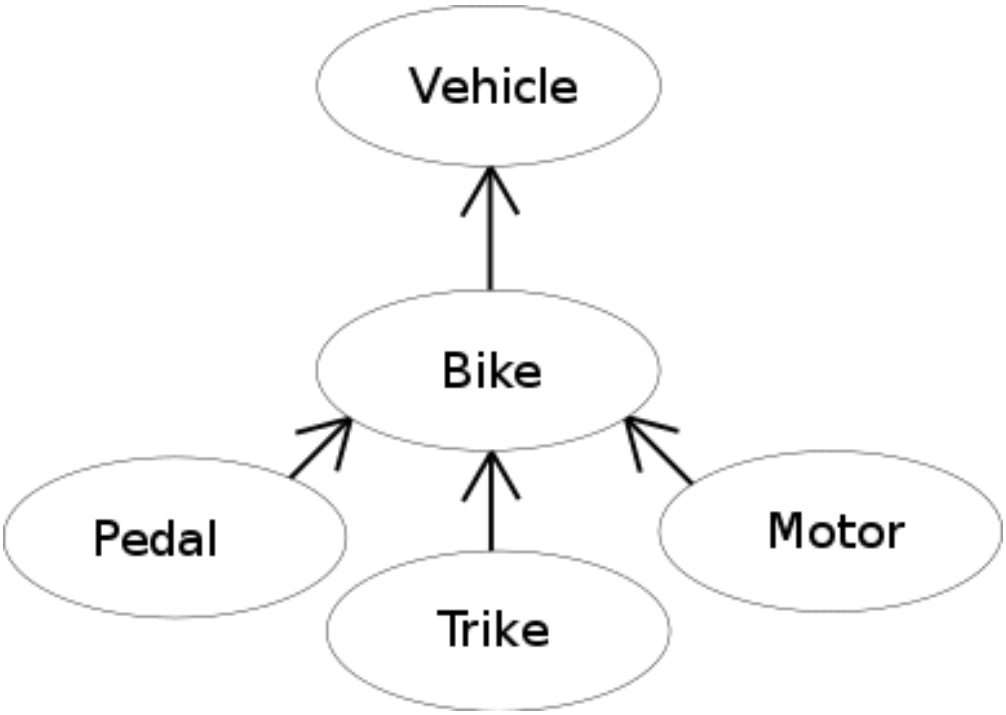

Figure 01: Inheritance

In Multiple inheritance there are many base classes and one derived class. Assume that class A and B are the base classes. Class C is the derived class. If both A and B classes have the same method and the programmer calls that method from the derived class, it will cause an ambiguity problem. Inheriting two classes can cause a compile-time error. Therefore, multiple inheritance is not supported in Java. An interface can be used to overcome that problem.

What is Interface in Java?

Abstraction is a process of hiding the implementation details and displaying only the functionalities to the user. Abstraction can be achieved using Abstract Classes or Interfaces. An abstract method is a method without an implementation. A class with at least one abstract method is an abstract class. Example of an abstract class is as follows.

```
abstract class A{

abstract void sum();

}
```

Assume that there are two abstract classes as A and B. To implement abstract methods of A and B, a new class C is created. Then class C should extend both A and B., But multiple inheritance is not supported in Java. Therefore, should use interfaces. Interfaces can be used to declare methods, but it is not possible to define methods. It is not possible to create an object using interfaces. Class C should implement all methods in interface A and B.

```
interface A{

void sum();

}

interface B{

void sub();

}


class C implements A,B{

public void sum(){

System.out.println(“Summation”);

}

public void sub(){

System.out.println(“Subtraction”);

}

}
```

Now, in the main program it is possible to create an object of C and call both methods.

```
public static void main (String [] args) {

C obj = new C();

obj.sum();

obj.sub();
```

```
}
```

So, interfaces can use to implement multiple inheritance.

Another use of interfaces is that it provides security. Refer the below code.

```
interface A {

void sum ();

}

class B implements A {

public void sum () {

System.out.println(“Summation”);

}

public void multiply () {

System.out.println(“Multiplication”);

}

}
```

When creating an object of B, it is possible to call both methods sum () and multiply (). If the programmer wants to restrict using multiply () function, it is possible as follows.

```
public static void main(String[]  args){

A obj= new B();

obj.sum();

}
```

A obj = new B();will create an object. It is of type A and the memory is allocated as B. It is possible to call sum() but cannot execute multiply(). This restriction is done using interfaces.

What are the Similarities Between Inheritance and Interface in Java?

- Both concepts are related to Object-Oriented
- Both represent IS-A relationship.

What is the Difference Between Inheritance and Interface in Java?

Inheritance vs Interface in Java	
Inheritance is an OOP concept to derive new classes from the existing classes.	Interface is a mechanism in OOP to implement abstraction and multiple inheritance.
Usage	
Inheritance provides code reusability.	Interfaces provide abstraction and multiple inheritance.

Summary - Inheritance vs Interface in Java

Java is a multi-paradigm programming language which supports object-oriented programming. Inheritance and interfaces are related to object-oriented programming. The difference between inheritance and interface is that inheritance is to derive new classes from existing classes and interfaces is to implement abstract classes and multiple inheritance.

Reference:

1.Point, Tutorials. “Java Inheritance.”, Tutorials Point, 31 Oct. 2017. Available here
2.Point, Tutorials. “Java Interfaces.” , Tutorials Point, 31 Oct. 2017. Available here

Image Courtesy:

1.'CPT-OOP-inheritance-bikes' By Pluke - Own work, (Public Domain) via Commons Wikimedia

How to Cite this Article?

APA: Difference Between Inheritance and Interface in Java.(2017 December 29). Retrieved (date), from http://differencebetween.com/difference-between-inheritance-and-vs-interface-in-java/

MLA: "Difference Between Inheritance and Interface in Java" Difference Between.Com. 29 December 2017. Web.

Chicago: “Difference Between Inheritance and Interface in Java.” Difference Between.Com. http://differencebetween.com/difference-between-inheritance-and-vs-interface-in-java/ accessed (accessed [date]).