

Abstract class and interface in java with example pdf

☐

I'm not robot


reCAPTCHA

Next

Abstract class vs Interface (Differences)

Abstract class	Interface
- To declare an abstract class, use abstract keyword. <code>public abstract class B{</code> <code>}</code> - A class can extend only one abstract class. <code>class A extends B{</code> <code>}</code> - In relationship, we say A is B.	- To declare an interface, use interface keyword. <code>public interface B{</code> <code>}</code> - A class can implement n than one interface. <code>class A implements C, D</code> <code>}</code> - In relationship, we say A has C, D, and E.



```
public interface Human {
    int A=20;
    public abstract void walk();
    public abstract void eat();
}

class Engineer implements Human {
    @Override
    public void eat() {
        System.out.println("Engineer can eat.");
    }
    @Override
    public void walk() {
        System.out.println("Engineer can walk.");
    }
    public static void main(String[] args) {
        Engineer E=new Engineer();
        E.eat();
        E.walk();
    }
}
```

Interface

Human

implements

Engineer

class

class Engineer does implement the methods declared in interface Human

OUTPUT:

Engineer can eat

Engineer can walk

	Abstract Class	Interface
Declaration	<code>abstract class Foo {...}</code>	<code>interface Foo {...}</code>
Methods	some or no abstract methods	all methods has no body
Instantiation	—	—
Inheritance	an abstract class can extend only one class (abstract or not)	an interface can extend many interfaces
Implementation	a class can extend only one abstract class (if the class doesn't implement all the abstract methods, it will be also abstract)	a class can implements any number of interfaces
Modifier Types	public, protected, private, static, final	only public, static, final
Method Types	public, protected, private	only public

Interface Vs Abstract Class (Java)	
Interface	Abstract Class
Interface can use abstract and public modifiers.	Abstract Class can use abstract and public modifiers.
abstract modifier is optional for an interface. All Interfaces are inherently public.	The abstract modifier must be used to declare a class as an Abstract Class
Interfaces without public access modifiers are visible within the package.	Abstract Class without public modifiers remains visible with package.
Interface cannot use protected and private modifiers.	Abstract Class cannot use non-public modifiers.

Difference between abstract class and interface

Abstract class and interface both are used to achieve abstraction where we can declare the abstract methods. Both class and interface both can't be instantiated.

But there are many differences between abstract class and interface that are given below.

Abstract class	Interface
1) Abstract class can have abstract and non-abstract methods.	Interface can have only abstract methods.
2) Abstract class doesn't support multiple inheritance.	Interface supports multiple inheritance.
3) Abstract class can have final, non-final, static and non-static variables.	Interface has only static and final variables.
4) Abstract class can have static methods, main method and constructor.	Interface can't have static methods, main method or constructor.
5) Abstract class can provide the implementation of interfaces.	Interface can't provide the implementation of abstract class.
6) The abstract keyword is used to declare abstract class.	The interface keyword is used to declare interface.
7) Example: public class Shape{ public abstract void draw(); }	Example: public interface Drawable{ void draw(); }

Simply, abstract class achieves partial abstraction (0 to 100%) whereas interface achieves fully abstraction (100%)

When to use abstract class and interface in java with example. Difference between abstract class and interface in java with real time example. Abstract class and interface in java with example pdf. Difference between abstract class and interface in java 8 with example. Difference between abstract class and interface in java with example. Abstract class and interface in java with real time example. When to use abstract class and interface in java with real time example. When to use abstract class and interface in java 8 with real time example.

In the previous tutorial, you must have noticed that abstract classes and interfaces have some similar. For example, both are used to get the Abstraction in Java. Both define abstracts that do not require an implementation. It is also similar in the sense that we can not create an instance of both abstract class and interface. But there are also many differences between them. Thus, Let's discuss the difference between the abstract class and the interface in Java. Abstract Class VS Interface in Java 1. Keyword (s) Used: A à two abstract keywords and class are used to define an abstract class. B. à only one interface keyword is used to set an interface. 2. Keyword used by class implementation: à a. à to inherit the abstract class, we use the keyword extends. B. à to implement an interface, we can use the keyword implements. 3. Variables: A. The abstract class may have final, non-final, static and non-static variables. B. The interface can not have any instance variable. It may have only static variables. 4. Initialization: a. The abstract class variable does not require the initialization accomplishment at the time of declaration. For example: Abstract Public Class A {à int x; // à, no error. } B. The interface variable must be initialized at the time of Declaration, if contrary to obtain compilation time error. For example: Interface A {INT X; // Compile Time Error Because The Final Blank Field X May Not Have Been Initialized. } 5. METHOD: A. à EVERYTHING ALL GIFT IN AN INTERFACE IS ALL Public and Abstract Whether we are declaring or not. That is why the interface is also known as pure abstract class (100%). B. à an abstract class can have all abstract and not abstract (concrete). 6. Builders: a. Within an interface we can not declare / define a constructor because the objective of the builder is to carry out the From the variable of instance, but within an interface each variable is always static. Therefore, within the interface, the constructor concept is not applicable and and Not required. B. As an abstract class can have variables à à à à of instance. Therefore, we can define constructors within the abstract class to initialize the variables à à à à instance. 7. Static and instance blocks: A. We can not declare static instance and blocks within an interface. If you declare them, you will have compiled time error. B. We can declare instance and static blocks within the abstract class. 8. Access modifiers: A. We can not define any private or protected members in an interface. All members are popular by standard. B. There is no restriction on declaring private or protected members within an abstract class. 9. Single vs. Multiple Inheritance: A. A class can extend only one class (which can be an abstract or concrete class). B. A class can implement any number of interfaces. 10. Standard implementation: a. An abstract class can provide a standard implementation of a method. Thus, the subclasses of an abstract class can only use this definition, but the subclasses can not define this method. B. An interface can only declare a method. All interface deployment classes should define this method. 11. Difficulty in making changes: A. It is not difficult to make changes in the implementation of the abstract class. For example, we can add a method with the default implementation and the existing subclass can not define it. B. It is not easy to make changes to an interface if many classes have already implemented this interface. For example, suppose you declare a new method in the interface, all the classes that implement that the interface will stop compiling because it does not define this method. 12. USA: A. If you do not know anything about the implementation. You have only requirement specification, so you should go to use the interface. B. If you know about implementation, but not completely (this is, partial implementation), then you should go for abstract class use. Program Abstract Java Class Example Let's create a sample program where we deal with all the points above the abstract abstract class Java. Programs Càdigo source 1: com.abstractProgram package; pàblica abstract class abstractclass { // Two keywords used: à Abstract class. // Declaraà à variàels the end, the final à nà, and urges estàticas à INSTANCE. int a; // In the à à requires the startup. final int b = 20; // Variable end. Static int c = 30; // Variable Static. // à Declaraà the mà à all abstract and the abstract à. abstract void m1 (); static void m2 () {System.out.println ("Static method in abstract class"); } // the default Aplicaà à mà à à INSTANCE at all calls. void m3 () { // MEDIUM à all concrete. System.out.println ("Mà à all Insta à INSTANCE in abstract class"); } // the Declaraà à à builders for startup of the Variable calls à INSTANCE. Abstractclass () {int a = 10; System.out.println ("Value of the " + à); } // Declaration of static and non-static blocks. static {System.out.println ("Static block in abstract class"); } {System.out.println ("Instance block in abstract class"); } // Declaration of private à protected members. private void m4 () {System.out.println ("Private method"); Protected void m5 () {System.out.println ("Protected method"); } } Public class A {void extends abstractclass m1 () {System.out.println ("Implementation of abstract method"); } } Public class AbstractTest {public static void main (String [] args) {A a = new A (); System.out.println ("Value of b:" a.b +); System.out.println ("Value of c" + AbstractClass.c); a.m1 (); AbstractClass.m2 (); a.m3 (); a.m5 (); } } Output: Static block in abstract class Instance block in abstract class Value of the 10 Value of B: 20 Value of c: 30 Implementation of abstract Method Static method in abstract class Instance method in abstract class Protected method Java Interface Example Program WTS à Càdigo program source 2: interfacePrograma package; AA pàblica interface { // A keyword: interface. int x = 20; // Variable The interface must be initialized at the time of the à declaraà. By default, the Variable à à interface and final. empty m1 (); By standard, the à interface system is à and sane interface. Here, we can't declare variables of instinct, information à, builders, sts and not static block. } bb public interface {int y = 20; empty m2 (); } Public class cc implements AA, BB { // Màtipla Heranà. public void m1 () {system.out.println ("x:" + x value); System.out.println ("Mà@mtodo M1"); } public void m2 () {system.out.println ("y:" + y value); System.out.println ("Mà@todo M2"); } } Public Class MyClass {Public Static Void Main (String [] Args) {CC C = NEW CC (); c.m1 (); c.m2 (); } } Come out: Value of X: 20 M1 Mà@todo value of Y: 20 m2 Mà@todo I hope that this tutorial has covered almost all the important points related to the difference between abstract class and interface in Java with sample programs. I hope you enjoyed this tutorial. Thanks for reading !!! Close 'Difference between class and interface in Java IVExt à'