Difference between Abstract Class and Interface in Java

APRIL 2, 2018 BY PANKAJ — 36 COMMENTS

Difference between Abstract Class and Interface is one of the popular interview question. Abstract Class and Interface are core part of java programming language. Whether to chose interface or abstract class is a design decision that every architect faces.

In my last articles, I have provided as much as possible details about java interface and **abstract class**.

In this post, we will learn about the difference between abstract class and Interface and when should we use interface over abstract class and vice versa.

Table of Contents [hide]

1 Difference between Abstract Class and Interface
  1.1 Interface or Abstract Class
  1.2 Use Abstract classes and Interface both
  1.3 Java 8 interface changes

Difference between Abstract Class and Interface

1. `abstract` keyword is used to create an abstract class and it can be used with methods also whereas `interface` keyword is used to create interface and it can't be used with methods.
2. Subclasses use `extends` keyword to extend an abstract class and they need to provide implementation of all the declared methods in the abstract class unless the subclass is also an abstract class whereas subclasses use `implements` keyword to implement interfaces and should provide implementation for all the methods declared in the interface.
3. Abstract classes can have methods with implementation whereas interface provides absolute abstraction and can't have any method implementations.
4. Abstract classes can have constructors but interfaces can't have constructors.
5. Abstract class have all the features of a normal java class except that we can't instantiate it. We can use `abstract` keyword to make a class abstract but interfaces are a completely different type and can have only public static final constants and method declarations.
6. Abstract classes methods can have access modifiers as public, private, protected, static but interface methods are implicitly public and abstract, we can't use any other access modifiers with interface methods.
7. A subclass can extend only one abstract class but it can implement multiple interfaces.
8. Abstract classes can extend other class and implement interfaces but interface can only extend other interfaces.
9. We can run an abstract class if it has `main()` method but we can't run an interface because they can't have main method implementation.
10. Interfaces are used to define contract for the subclasses whereas abstract class also define contract but it can provide other methods implementations for subclasses to use.

Thats all for the ***difference between interface and abstract classes***, now we can move on to know when should we use Interface over Abstract class and vice versa.

Interface or Abstract Class

Whether to chose between Interface or abstract class for providing contract for subclasses is a design decision and depends on many factors, lets see when Interfaces are best choice and when can we use abstract classes.

1. Java doesn't support multiple class level inheritance, so every class can extend only one superclass. But a class can implement multiple interfaces. So most of the times Interfaces are a good choice for providing base for class hierarchy and contract. Also coding in terms of interfaces is one of the best practices for coding in java.
2. If there are a lot of methods in the contract, then abstract class is more useful because we can provide default implementation for some of the methods that are common for all the subclasses. Also if subclasses don't need to implement particular method, they can avoid providing the implementation but in case of interface, the subclass will have to provide implementation for all the methods even though it's of no use and implementation is just empty block.
3. If our base contract keeps on changing then interfaces can cause issues because we can't declare additional methods to the interface without changing all the implementation classes, with abstract class we can provide the default implementation and only change the implementation classes that are actually going to use the new methods.

Use Abstract classes and Interface both

Actually most of the times, using Interfaces and abstract classes together is the best approach for designing a system, for example in JDK `java.util.List` is an interface that contains a lot of methods, so there is an abstract class `java.util.AbstractList` that provides skeletal implementation for all the methods of List interface so that any subclass can extend this class and implement only required methods.

We should always start with an interface as base and define methods that every subclasses should implement and then if there are some methods that only certain subclass should implement, we can extend the base interface and create a new interface with those methods. The subclasses will have option to chose between the base interface or the child interface to implement according to its requirements. If the number of methods grows a lot, its not a bad idea to provide a skeletal abstract class implementing the child interface and providing flexibility to the subclasses to chose between interface and abstract class.

Java 8 interface changes

From Java 8 onwards, we can have method implementations in the interfaces. We can create default as well as static methods in the interfaces and provide implementation for them. This has bridge the gap between abstract classes and interfaces and now interfaces are the way to go because we can extend it further by providing default implementations for new methods. For more details, check out Java 8 interface default static methods.

« PREVIOUS
Interface in Java

NEXT »
Java Priority Queue (PriorityQueue) Example

About Pankaj
If you have come this far, it means that you liked what you are reading. Why not reach little more and connect with me directly on **Google Plus**, **Facebook** or **Twitter**. I would love to hear your thoughts and opinions on my articles directly.

Recently I started creating video tutorials too, so do check out my videos on **Youtube**.

FILED UNDER: JAVA

Comments

Vivek Dubey says

JULY 14, 2018 AT 10:45 AM

Dear Pankaj,

Please tell me where we use abstract class and where we use interface in java project.

Reply

GST Billing Software says

NOVEMBER 16, 2017 AT 3:16 AM

Nice and very straight forward post. Thanks for posting.

Reply

Chetan says

SEPTEMBER 15, 2017 AT 1:27 PM

We can run interface from Java8 as it can have static method and we can execute so the difference of not running interface is invalid after java8, either u can update or add.

Reply

sanjay pradhan says

AUGUST 29, 2017 AT 8:27 PM

Nice explanation.

Reply

Anurag Singh says

AUGUST 21, 2017 AT 1:53 AM

Thanks for sharing this post.

keep it up.

Reply

Arpit Mandliya says

JULY 6, 2017 AT 3:35 AM

Great Explanation,

Your article provides quality content on difference between Abstract class and Interface. It is a very common question usually asked in interviews. I agree with your all points. We can achieve Abstraction 0-

100% through abstract class and 100% abstraction from Interfaces because Interface can contain only abstract methods but Abstract class may have abstract method as well as methods with implementation. I am also running a blog (java2blog) which is tutorial on Java. It also provides useful interview questions and answers on Java.
Reply

Stan says
JUNE 22, 2017 AT 12:19 PM
Hi Pankaj,
I think "Subclasses use extends keyword to extend an abstract class and they need to provide implementation of all the declared methods in the abstract class" is not correct. It should be "Subclasses use extends keyword to extend an abstract class and they need to provide implementation of all the declared abstract methods in the abstract class "
Reply

Vikas Palakurthi says
MAY 30, 2017 AT 10:50 AM
Can abstract methods in an abstract class be private? No right?
Reply

Pankaj says
MAY 31, 2017 AT 8:48 AM
No it can't, if you ever face these kind of questions, then best way to just write quick code in Eclipse and see if you get errors. □
Reply

Oshin Talreja says
MAY 25, 2017 AT 5:18 PM
You mention that abstract class can be run throw main method while interface can't However abstract class cant be instantiated so how JVM is going to run the abstract class.
You can call any static method with Abstract class implementation, No matter it is main method or not.
Reply

KALADHAR KHAMITHKAR says
APRIL 14, 2017 AT 7:40 PM
Hi Pankaj,
Good Explanation on Abstract and Interfaces.
Can you Post Tutorials on MAVEN, LOG4J and JUnit.
Thanks
Reply

Aliasger Motiwala says
NOVEMBER 7, 2016 AT 11:27 PM
You have written
"The subclasses will have option to chose between the base interface or the child interface to implement according to its requirements"
Hows this possible ?
Reply

Pankaj says
NOVEMBER 9, 2016 AT 6:26 AM
Let's say we have interface A with abstract method foo() and child interface B with method bar(). Now my subclass can either implement A or B, based on whether it just need to provide foo() method or bar() method too.
Reply

Jayavardhana says
NOVEMBER 29, 2016 AT 9:32 PM
HI Pankaj,
Good explanation about abstract and interface, but i have one doubt about abstract class,In future added abstaract method ,it should be override all subclasses. but you said not required to implement.I am confusing this point .please clarify
Reply

Shwetha says
JUNE 5, 2016 AT 9:01 AM
Hi. I have a question.
Assume that there is an abstract class with method String Name() and an Interface with the same

method String Name(). The signatures of both methods in abstract class and interface is same.
Now Class A extends the abstract class and implements the interface.
Which String Name() method is implemented in Class A???
Example :
//AbstractClass
public abstract AbstractClass {
public abstract String Name();
}
//Interface
Public interface SampleInterface{
public String Name();
}
// which method is implemented here???
public class A extends AbstractClass implements SampleInterface{
public String Name(){
//Implementation
// Which method is implemented here. The abstract class method or interface method???
}
I tried executing a small sample program. The program runs fine but i am not sure as to which method is implemented here.
Reply

Pankaj says
JUNE 5, 2016 AT 11:33 AM
Since both super methods are abstract, it's working fine. It doesn't matter which method is implemented.
Reply

Shiva says
DECEMBER 15, 2017 AT 9:19 AM
very good question Shwetha :-). your thinking is awesome
Reply

Waseem Ahmed Kalhoro says
FEBRUARY 20, 2015 AT 10:44 AM
What is the simple definition of abstract??
Give me an example..
Reply

abhishek chanda says
JANUARY 27, 2015 AT 7:03 AM
"… Also if subclasses don't need to implement particular method, they can avoid providing the implementation… " – does that mean we can choose NOT to implement any method of the abstract class?
Reply

suresh says
APRIL 9, 2014 AT 12:19 PM
very good collective information on abstract and interfcae
Reply

Swati Aher says
MARCH 19, 2014 AT 9:37 AM
thank You …for giving very good explanation…
Reply

prabhu says
DECEMBER 28, 2013 AT 3:15 PM
best explanation,Thank you.
Reply

rishi says
DECEMBER 21, 2013 AT 7:05 PM
the best explanation i came across. going to copy this page forever.
Reply

sha says
NOVEMBER 29, 2013 AT 11:38 AM

sir
8. interface can only extend other interfaces.
but interface only implements other interface
correct
Reply

Pankaj says
NOVEMBER 29, 2013 AT 7:51 PM
Abstract class can implement interfaces because it's like normal class with some abstract methods. Interfaces don't have implementation, so they can ONLY extend other interfaces.
Reply

lakshmi says
SEPTEMBER 18, 2013 AT 2:58 PM
thank u sir
Reply

lakshmi says
SEPTEMBER 18, 2013 AT 2:56 PM
sir can u give me reply for the question i have asked
on last point
Reply

lakshmi says
SEPTEMBER 17, 2013 AT 6:33 PM
you mentioned in last point"Interfaces are used to define contract for the subclasses" like what is ' contract here' klindly explain that point
Reply

Pankaj says
SEPTEMBER 17, 2013 AT 10:02 PM
Contract are the behavior methods that are declared in the interface that all the implementation classes must define.
Reply

Tuntun kumar says
OCTOBER 9, 2013 AT 5:23 PM
very good collective information on abstract and interfcae
Reply

Madiraju Krishna Chaitanya says
AUGUST 15, 2013 AT 3:27 PM
Hi Pankaj Sir,
Thank You for sharing this wonderful site with us.
Would be Thankful to you,if you can shed light on ANT,JUnit,TestNG topics too.
I am an Automation tester in Selenium and am looking forward for the topics mentioned above,since,your experience and understanding helps us understand the topics better.
Thanks Again.
~Chaitanya
Reply

Pankaj says
AUGUST 15, 2013 AT 8:39 PM
ANT is an outdated technology now when it comes to build and deployment tools, I am planning to post some tutorials for Maven.
JUnit and TestNG tutorials will also arrive in coming months.
Thanks for your kind words, it pushes me to keep going.
Reply

Lucas says
SEPTEMBER 23, 2013 AT 1:50 PM
You have given
8.Abstract classes can extend other class and implement interfaces but interface can only extend other interfaces.
I have a doubt can abstract class “extend” another abstract class or “implement” another abstract class
reply to my email please……
Reply

Pankaj says
SEPTEMBER 23, 2013 AT 10:28 PM

Yes, an abstract class can extend other classes or other abstract classes and implement interfaces. You can check this easily with sample code in Eclipse.

```
public abstract class Ab1 {
public abstract void ss();
}
public interface It1 {
public void xx();
}
public abstract class Ab2 extends Ab1 implements It1{
public abstract void tt();
}
```

Ab2 is extending abstract class Ab1 and implementing interface It1.

Reply

sunil patil says

JULY 16, 2013 AT 1:07 PM

Hello Pankaj,
I like this post much. I got clear idea about when to use interfaces and when to use abstract class.
But i didnt understand last paragraph i.e extending interface.
could u elobarate more. So that it will understandable.
I will be clear if u provide some bit of code. Please
Thanks.

Reply

Pankaj says

JULY 16, 2013 AT 8:58 PM

It feels great that you liked the post, as I mentioned that once you have created an interface, its hard to change its behaviors because that would result in compilation error for all the implementation classes, so we can use interfaces extend functionality to create a new interface rather than changing the base interface.
To learn more about this, read https://www.journaldev.com/1601/what-is-interface-in-java-example-tutorial

Reply

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Save my name, email, and website in this browser for the next time I comment.

POST COMMENT

Search for tutorials...

DOWNLOAD ANDROID APP

GET IT ON Google Play

CORE JAVA TUTORIAL

Java 10 Tutorials Java 9 Tutorials Java 8 Tutorials Java 7 Tutorials Core Java Basics OOPS Concepts Data Types and Operators String Manipulation Java Arrays Annotation and Enum

Java Collections Java IO Operations Java Exception Handling MultiThreading and Concurrency Regular Expressions Advanced Java Concepts

RECOMMENDED TUTORIALS

Java Tutorials

- Java IO
- Java Regular Expressions
- Multithreading in Java
- Java Logging
- Java Annotations
- Java XML
- Collections in Java
- Java Generics
- Exception Handling in Java
- Java Reflection
- Java Design Patterns
- JDBC Tutorial

Java EE Tutorials

- Servlet JSP Tutorial
- Struts2 Tutorial
- Spring Tutorial
- Hibernate Tutorial
- Primefaces Tutorial
- Apache Axis 2
- JAX-RS
- Memcached Tutorial